



2D Shaper is a **full featured, high performance** 2D drawing library.

It provides essential 2D drawing capabilities for ImGui (the “Immediate Mode” GUI system) or for any other script that can leverage OpenGL 2D drawing (UI, charts, HUDs, editor customization, etc).

#### Features:

- Simple shapes drawing: points, lines, rects, quads circles, arcs, triangles, polygons, arrows...
- Fully configurable line style, including: line thickness, color and dashed line pattern
- Full support for sprite rendering with configurable size, orientation and overlay color
- Full transparency support
- Optional fill texture for different patterns
- Easy text drawing with configurable size and color, or with fully customizable text style

#### Other utilities:

- Get world position and size of a UI element
- Coordinates conversion between GUI, screen and OpenGL spaces

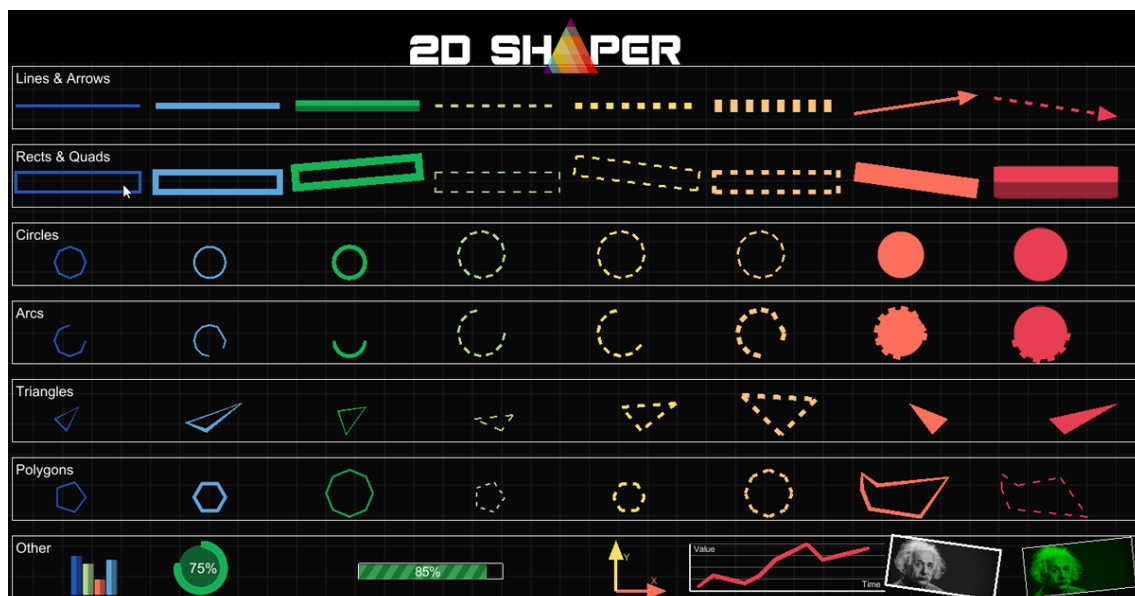
**Please note:** This package is not a component that can be added to a game object in the inspector. It's a code library you can use from a script (typically from methods like OnGui, OnRenderObject or OnPostRender). It does include though a sample scene showing the most frequent usage cases.

## Support

If you need any support with the package, please contact: [iaiyucar@simax.es](mailto:iaiyucar@simax.es)

## Examples

2D Shaper can draw and fill most common 2D shapes:



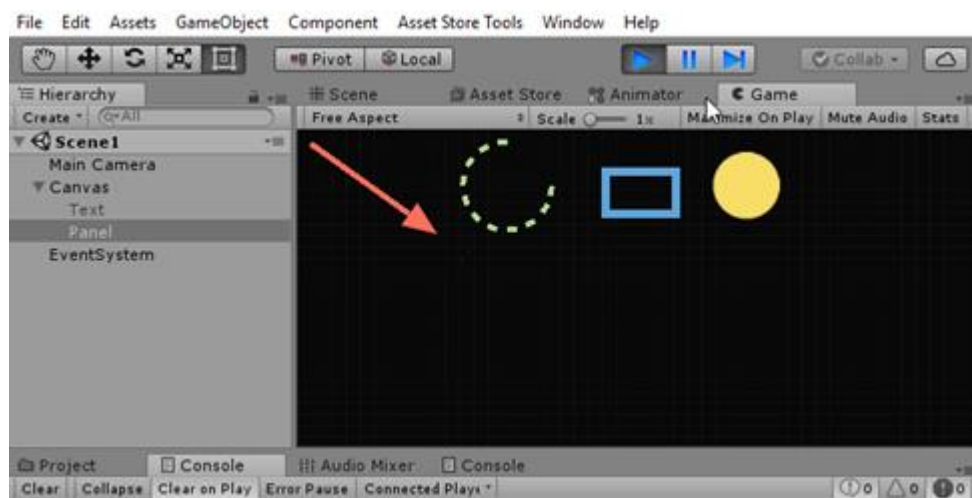
## Sample Code

The following code shows a basic use case in the OnGUI method of a panel, inside a Canvas.

It first sets the bounds of the parent panel where the draw will be done, and then calls several drawing methods:

```
6 public class DrawTestScript : MonoBehaviour
7 {
8
9     private void OnGUI()
10    {
11        Rect parentRect = Drawing2D.GetWorldRect(this.transform as RectTransform);
12        Drawing2D.SetParentBounds(parentRect, false);
13
14        Drawing2D.DrawArrow(new Vector2(10, 10), new Vector2(80, 60), ToColor(255, 112, 92), 4, 20, 50);
15        Drawing2D.DrawDashedArc(new Vector2(150, 40), 32, 14, 0, 270, ToColor(175, 217, 141), 4f, 4f);
16        Drawing2D.DrawRect(new Rect(220, 30, 50, 30), ToColor(93, 166, 221), 6f);
17        Drawing2D.FillCircle(new Vector2(320, 40), 24, ToColor(248, 222, 104));
18
19        Drawing2D.ClearParentBounds();
20    }
```

That code produces the following result:



## Appendix 1 - Reference

# Drawing2D Class

**Namespace:** [GraphicDNA](#)


**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

`public class` Drawing2D

[Copy](#)













The [Drawing2D](#) type exposes the following members.
















## Constructors














	Name	Description
	<a href="#">Drawing2D</a>	Initializes a new instance of the <a href="#">Drawing2D</a> class












[Top](#)














## Methods

	Name	Description
 	<a href="#">ClearFrameBuffer</a>	Clears the frame buffer window to the specified color
 	<a href="#">ClearParentBounds</a>	Clears any parent bounds or coordinate system. After a call to this method, clipping will be deactivated and all coordinates will be relative to screen's origin.
 	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single)</a>	Draws a Arc. GUI.ContentColor is assumed to draw the circle
 	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single, Color)</a>	Draws a Arc.
 	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single, Color, Single)</a>	Draws a Arc.
 	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single, Color, Single, Texture2D)</a>	Draws a Arc.



























	<a href="#"><u>DrawArrow</u></a>	Draws an arrow
	<a href="#"><u>DrawCircle(Vector2, Single, Int32)</u></a>	Draws a circle. GUI.ContentColor is assumed to draw the circle
	<a href="#"><u>DrawCircle(Vector2, Single, Int32, Color)</u></a>	Draws a circle.
	<a href="#"><u>DrawCircle(Vector2, Single, Int32, Color, Single)</u></a>	Draws a circle.
	<a href="#"><u>DrawCircle(Vector2, Single, Int32, Color, Single, Texture2D)</u></a>	Draws a circle.
	<a href="#"><u>DrawDashedArc</u></a>	Draws a dashed Arc.
	<a href="#"><u>DrawDashedArrow</u></a>	Draws an arrow
	<a href="#"><u>DrawDashedCircle</u></a>	Draws a circle.
	<a href="#"><u>DrawDashedHexagon</u></a>	Draws a dashed Hexagon
	<a href="#"><u>DrawDashedLine(Vector2, Vector2)</u></a>	Draws a dashed line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#"><u>DrawDashedLine(Vector2, Vector2, Single)</u></a>	Draws a dashed line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#"><u>DrawDashedLine(Vector2, Vector2, Color)</u></a>	Draws a dashed line between two points.
	<a href="#"><u>DrawDashedLine(Vector2, Vector2, Color, Single, Single)</u></a>	Draws a dashed line between two points.
	<a href="#"><u>DrawDashedOctagon</u></a>	Draws a dashed Octagon
	<a href="#"><u>DrawDashedPentagon</u></a>	Draws a dashed pentagon
	<a href="#"><u>DrawDashedPolygon</u></a>	Draws a closed or open, dashed polygon (draws a dashed line between each pair of vertices, in order, and if pLeaveOpen is false, draws a closing line between the last and the first)

	<a href="#"><u>DrawDashedQuad</u></a>	Fills a quad with an arbitrary orientation
	<a href="#"><u>DrawDashedRect</u></a>	Draws a rectangle (draws the four sides of it)
	<a href="#"><u>DrawDashedTriangle</u></a>	Draws a Triangle, defined by 3 points
	<a href="#"><u>DrawHexagon(Vector2, Single, Color, Single)</u></a>	Draws a Hexagon
	<a href="#"><u>DrawHexagon(Vector2, Single, Color, Single, Texture2D)</u></a>	Draws a Hexagon
	<a href="#"><u>DrawLine(Vector2, Vector2)</u></a>	Draws a line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#"><u>DrawLine(Vector2, Vector2, Single)</u></a>	Draws a line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#"><u>DrawLine(Vector2, Vector2, Color)</u></a>	Draws a line between two points.
	<a href="#"><u>DrawLine(Vector2, Vector2, Color, Single)</u></a>	Draws a line between two points.
	<a href="#"><u>DrawLine(Vector2, Vector2, Color, Single, Texture2D, NullableVector2)</u></a>	Draws a line between two points, using a customized base texture to fill the segment
	<a href="#"><u>DrawLines(Vector2, Color)</u></a>	Draws lines with thickness == 1, using GUI 2D points
	<a href="#"><u>DrawLines(Vector3, Color)</u></a>	Draws lines with thickness == 1, using OpenGL Vector3 points in homogeneous coords. Please use the BuildGLVertexBuffer method to convert from GUI coordinates to OpenGL homogeneous coordinates
	<a href="#"><u>DrawLines(Vector2, Color, Single, Texture2D, NullableVector2, Boolean)</u></a>	Draws lines, using OpenGL Vector3 points in homogeneous coords. Please use the

		BuildGLVertexBuffer method to convert from GUI coordinates to OpenGL homogeneous coordinates
	<a href="#"><u>DrawOctagon(Vector2, Single, Color, Single)</u></a>	Draws a Octagon
	<a href="#"><u>DrawOctagon(Vector2, Single, Color, Single, Texture2D)</u></a>	Draws a Octagon
	<a href="#"><u>DrawPentagon(Vector2, Single, Color, Single)</u></a>	Draws a pentagon
	<a href="#"><u>DrawPentagon(Vector2, Single, Color, Single, Texture2D)</u></a>	Draws a pentagon
	<a href="#"><u>DrawPoint(Vector2, Color)</u></a>	Draws a point
	<a href="#"><u>DrawPoint(Vector2, Color, Single)</u></a>	Draws a point
	<a href="#"><u>DrawPolygon(Vector2)</u></a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first) Assumes GUI.ContentColor
	<a href="#"><u>DrawPolygon(Vector2, Color)</u></a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)
	<a href="#"><u>DrawPolygon(Vector2, Color, Single)</u></a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)
	<a href="#"><u>DrawPolygon(Vector2, Color, Single, Texture2D)</u></a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)
	<a href="#"><u>DrawPolygon(Vector2, Color, Single, Texture2D, Boolean, Boolean)</u></a>	Draws a closed or open polygon (draws a line between each pair of vertices, in order, and if

		pLeaveOpen is false, draws a closing line between the last and the first)
	<a href="#">DrawQuad</a>	Fills a quad with an arbitrary orientation
	<a href="#">DrawRect(Rect)</a>	Draws a rectangle (draws the four sides of it). GUI.ContentColor is assumed
	<a href="#">DrawRect(Rect, Color)</a>	Draws a rectangle (draws the four sides of it)
	<a href="#">DrawRect(Rect, Color, Single)</a>	Draws a rectangle (draws the four sides of it)
	<a href="#">DrawRect(Rect, Color, Single, Texture2D)</a>	Draws a rectangle (draws the four sides of it)
	<a href="#">DrawRects</a>	Draws Rects using OpenGL, ideal for performance critical situations. Please note: Rects drawn using OpenGL are not affected by clipping
	<a href="#">DrawText(String, Single, Single, GUIStyle)</a>	Draws text with a customizable style
	<a href="#">DrawText(String, Vector2, Int32, Color)</a>	Draws text
	<a href="#">DrawText(String, Single, Single, Int32, Color)</a>	Draws text
	<a href="#">DrawTexture</a>	Draws a texture
	<a href="#">DrawTriangle(Vector2, Vector2, Vector2, Color, Single)</a>	Draws a Triangle, defined by 3 points
	<a href="#">DrawTriangle(Vector2, Vector2, Vector2, Color, Single, Texture2D)</a>	Draws a Triangle, defined by 3 points
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)



 	<a href="#">FillCircle(Vector2, Single)</a>	Fills a circle, using GUI.Content color
 	<a href="#">FillCircle(Vector2, Single, Color)</a>	Fills a circle
 	<a href="#">FillCircle(Vector2, Single, Color, Texture2D)</a>	Fills a circle
 	<a href="#">FillQuad</a>	Fills a quad with an arbitrary orientation
 	<a href="#">FillQuads</a>	Fills quads that can have any orientation. Each quad is defined by its 4 vertices in the following order: BL, TL, TR, BR
 	<a href="#">FillRect(Rect)</a>	Fills a rectangle. GUI.ContentColor is assumed
 	<a href="#">FillRect(Rect, Color)</a>	Fills a rectangle.
 	<a href="#">FillRect(Rect, Color, Texture2D, NullableVector2)</a>	Fills a rectangle.
 	<a href="#">FillTriangle</a>	
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 	<a href="#">GetWorldRect</a>	Returns the world position and size of a UI element (provided its RectTransform) The returned position corresponds to the Top-Left corner of the element.
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 	<a href="#">SetParentBounds</a>	Sets the coordinate system for drawing operations so the (0,0) is the top-left corner of the group. If pClip is enabled, all controls are clipped to the group. Groups CANNOT be nested Please note: Do not use GUI.BeginGroup to

---

achieve this, as that might result in a malfunction of the drawing features

---









[ToString](#)

(Inherited from [Object](#).)

---

[Top](#)

## ▲ Properties

	Name	Description
 <b>S</b>	<a href="#">DefaultCircleTexture</a>	Default circle texture used for filling round shapes
 <b>S</b>	<a href="#">DefaultDashedLineTexture</a>	Default pixel texture used for filling dashed shapes
 <b>S</b>	<a href="#">DefaultTextStyle</a>	Default text style to render text
 <b>S</b>	<a href="#">DefaultTriangleTexture</a>	Default Triangle texture used for filling triangle shapes
 <b>S</b>	<a href="#">MaterialColorAndTexture</a>	Returns the default material to fill shapes
 <b>S</b>	<a href="#">MaterialColorOnly</a>	Returns the default material to fill shapes

---

[Top](#)

## ▲ See Also

Reference
















[GraphicDNA Namespace](#)














---












# Drawing2D Methods












The [Drawing2D](#) type exposes the following members.






























## ▲ Methods














	Name	Description
 	<a href="#">ClearFrameBuffer</a>	Clears the frame buffer window to the specified color
 	<a href="#">ClearParentBounds</a>	Clears any parent bounds or coordinate system. After a call to this method, clipping will be deactivated and all coordinates will be relative to screen's origin.
 	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single)</a>	Draws a Arc. GUI.ContentColor is assumed to draw the circle
 	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single, Color)</a>	Draws a Arc.
 	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single, Color, Single)</a>	Draws a Arc.
 	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single, Color, Single, Texture2D)</a>	Draws a Arc.
 	<a href="#">DrawArrow</a>	Draws an arrow
 	<a href="#">DrawCircle(Vector2, Single, Int32)</a>	Draws a circle. GUI.ContentColor is assumed to draw the circle
 	<a href="#">DrawCircle(Vector2, Single, Int32, Color)</a>	Draws a circle.
 	<a href="#">DrawCircle(Vector2, Single, Int32, Color, Single)</a>	Draws a circle.

	<a href="#"><u>DrawCircle(Vector2, Single, Int32, Color, Single, Texture2D)</u></a>	Draws a circle.
	<a href="#"><u>DrawDashedArc</u></a>	Draws a dashed Arc.
	<a href="#"><u>DrawDashedArrow</u></a>	Draws an arrow
	<a href="#"><u>DrawDashedCircle</u></a>	Draws a circle.
	<a href="#"><u>DrawDashedHexagon</u></a>	Draws a dashed Hexagon
	<a href="#"><u>DrawDashedLine(Vector2, Vector2)</u></a>	Draws a dashed line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#"><u>DrawDashedLine(Vector2, Vector2, Single)</u></a>	Draws a dashed line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#"><u>DrawDashedLine(Vector2, Vector2, Color)</u></a>	Draws a dashed line between two points.
	<a href="#"><u>DrawDashedLine(Vector2, Vector2, Color, Single, Single)</u></a>	Draws a dashed line between two points.
	<a href="#"><u>DrawDashedOctagon</u></a>	Draws a dashed Octagon
	<a href="#"><u>DrawDashedPentagon</u></a>	Draws a dashed pentagon
	<a href="#"><u>DrawDashedPolygon</u></a>	Draws a closed or open, dashed polygon (draws a dashed line between each pair of vertices, in order, and if pLeaveOpen is false, draws a closing line between the last and the first)
	<a href="#"><u>DrawDashedQuad</u></a>	Fills a quad with an arbitrary orientation
	<a href="#"><u>DrawDashedRect</u></a>	Draws a rectangle (draws the four sides of it)
	<a href="#"><u>DrawDashedTriangle</u></a>	Draws a Triangle, defined by 3 points

	<a href="#"><u>DrawHexagon(Vector2, Single, Color, Single)</u></a>	Draws a Hexagon
	<a href="#"><u>DrawHexagon(Vector2, Single, Color, Single, Texture2D)</u></a>	Draws a Hexagon
	<a href="#"><u>DrawLine(Vector2, Vector2)</u></a>	Draws a line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#"><u>DrawLine(Vector2, Vector2, Single)</u></a>	Draws a line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#"><u>DrawLine(Vector2, Vector2, Color)</u></a>	Draws a line between two points.
	<a href="#"><u>DrawLine(Vector2, Vector2, Color, Single)</u></a>	Draws a line between two points.
	<a href="#"><u>DrawLine(Vector2, Vector2, Color, Single, Texture2D, NullableVector2)</u></a>	Draws a line between two points, using a customized base texture to fill the segment
	<a href="#"><u>DrawLines(Vector2, Color)</u></a>	Draws lines with thickness == 1, using GUI 2D points
	<a href="#"><u>DrawLines(Vector3, Color)</u></a>	Draws lines with thickness == 1, using OpenGL Vector3 points in homogeneous coords. Please use the BuildGLVertexBuffer method to convert from GUI coordinates to OpenGL homogeneous coordinates
	<a href="#"><u>DrawLines(Vector2, Color, Single, Texture2D, NullableVector2, Boolean)</u></a>	Draws lines, using OpenGL Vector3 points in homogeneous coords. Please use the BuildGLVertexBuffer method to convert from GUI coordinates to OpenGL homogeneous coordinates
	<a href="#"><u>DrawOctagon(Vector2, Single, Color, Single)</u></a>	Draws a Octagon

	<a href="#"><u>DrawOctagon(Vector2, Single, Color, Single, Texture2D)</u></a>	Draws a Octagon
	<a href="#"><u>DrawPentagon(Vector2, Single, Color, Single)</u></a>	Draws a pentagon
	<a href="#"><u>DrawPentagon(Vector2, Single, Color, Single, Texture2D)</u></a>	Draws a pentagon
	<a href="#"><u>DrawPoint(Vector2, Color)</u></a>	Draws a point
	<a href="#"><u>DrawPoint(Vector2, Color, Single)</u></a>	Draws a point
	<a href="#"><u>DrawPolygon(Vector2)</u></a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first) Assumes GUI.ContentColor
	<a href="#"><u>DrawPolygon(Vector2, Color)</u></a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)
	<a href="#"><u>DrawPolygon(Vector2, Color, Single)</u></a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)
	<a href="#"><u>DrawPolygon(Vector2, Color, Single, Texture2D)</u></a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)
	<a href="#"><u>DrawPolygon(Vector2, Color, Single, Texture2D, Boolean, Boolean)</u></a>	Draws a closed or open polygon (draws a line between each pair of vertices, in order, and if pLeaveOpen is false, draws a closing line between the last and the first)
	<a href="#"><u>DrawQuad</u></a>	Fills a quad with an arbitrary orientation

 	<a href="#"><u>DrawRect(Rect)</u></a>	Draws a rectangle (draws the four sides of it). GUI.ContentColor is assumed
 	<a href="#"><u>DrawRect(Rect, Color)</u></a>	Draws a rectangle (draws the four sides of it)
 	<a href="#"><u>DrawRect(Rect, Color, Single)</u></a>	Draws a rectangle (draws the four sides of it)
 	<a href="#"><u>DrawRect(Rect, Color, Single, Texture2D)</u></a>	Draws a rectangle (draws the four sides of it)
 	<a href="#"><u>DrawRects</u></a>	Draws Rects using OpenGL, ideal for performance critical situations. Please note: Rects drawn using OpenGL are not affected by clipping
 	<a href="#"><u>DrawText(String, Single, Single, GUIStyle)</u></a>	Draws text with a customizable style
 	<a href="#"><u>DrawText(String, Vector2, Int32, Color)</u></a>	Draws text
 	<a href="#"><u>DrawText(String, Single, Single, Int32, Color)</u></a>	Draws text
 	<a href="#"><u>DrawTexture</u></a>	Draws a texture
 	<a href="#"><u>DrawTriangle(Vector2, Vector2, Vector2, Color, Single)</u></a>	Draws a Triangle, defined by 3 points
 	<a href="#"><u>DrawTriangle(Vector2, Vector2, Vector2, Color, Single, Texture2D)</u></a>	Draws a Triangle, defined by 3 points
	<a href="#"><u>Equals</u></a>	(Inherited from <a href="#"><u>Object</u></a> .)
 	<a href="#"><u>FillCircle(Vector2, Single)</u></a>	Fills a circle, using GUI.Content color
 	<a href="#"><u>FillCircle(Vector2, Single, Color)</u></a>	Fills a circle
 	<a href="#"><u>FillCircle(Vector2, Single, Color, Texture2D)</u></a>	Fills a circle

	<a href="#">FillQuad</a>	Fills a quad with an arbitrary orientation
	<a href="#">FillQuads</a>	Fills quads that can have any orientation. Each quad is defined by its 4 vertices in the following order: BL, TL, TR, BR
	<a href="#">FillRect(Rect)</a>	Fills a rectangle. GUI.ContentColor is assumed
	<a href="#">FillRect(Rect, Color)</a>	Fills a rectangle.
	<a href="#">FillRect(Rect, Color, Texture2D, NullableVector2)</a>	Fills a rectangle.
	<a href="#">FillTriangle</a>	
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetWorldRect</a>	Returns the world position and size of a UI element (provided its RectTransform) The returned position corresponds to the Top-Left corner of the element.
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">SetParentBounds</a>	Sets the coordinate system for drawing operations so the (0,0) is the top-left corner of the group. If pClip is enabled, all controls are clipped to the group. Groups CANNOT be nested Please note: Do not use GUI.BeginGroup to achieve this, as that might result in a malfunction of the drawing features
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

[Top](#)

▲ See Also



Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DClearFrameBuffer Method

Clears the frame buffer window to the specified color

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void ClearFrameBuffer( Color pClearColor )
```

### Parameters

*pClearColor*

Type: **Color**

[Missing <param name="pClearColor"/> documentation for  
"M:GraphicDNA.Drawing2D.ClearFrameBuffer(UnityEngine.Color)"]

## ▲ See Also

### Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DClearParentBounds Method

Clears any parent bounds or coordinate system. After a call to this method, clipping will be deactivated and all coordinates will be relative to screen's origin.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void ClearParentBounds()
```

## ▲ See Also

[Reference](#)

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawArc Method

## ▲ Overload List

	Name	Description
	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single)</a>	Draws a Arc. GUI.ContentColor is assumed to draw the circle
	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single, Color)</a>	Draws a Arc.
	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single, Color, Single)</a>	Draws a Arc.
	<a href="#">DrawArc(Vector2, Single, Int32, Single, Single, Color, Single, Texture2D)</a>	Draws a Arc.

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawArc Method (Vector2, Single, Int32, Single, Single)

Draws a Arc. GUI.ContentColor is assumed to draw the circle

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawArc(      Vector2 center, float radius,
                               int sides,      float pStartAngleDeg,      float pDegrees )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

*pStartAngleDeg*

Type: [SystemSingle](#)

Degrees where arc starts (being 0° == 15h in a clock and 270° == 12h)

*pDegrees*

Type: [SystemSingle](#)

Number of degrees to rotate, starting from pStartAngleDeg

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawArc Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawArc Method (Vector2, Single, Int32, Single, Single, Color)

Draws a Arc.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawArc(      Vector2 center, float radius,
                               int sides,      float pStartAngleDeg,      float pDegrees,
                               Color color )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

*pStartAngleDeg*

Type: [SystemSingle](#)

Degrees where arc starts (being 0° == 12h in a clock and 270° == 12h)

*pDegrees*

Type: [SystemSingle](#)

Number of degrees to rotate, starting from pStartAngleDeg

*color*

Type: **Color**

Color of the circle

## ▴ See Also

Reference

[Drawing2D Class](#)

[DrawArc Overload](#)

[GraphicDNA Namespace](#)

---



# Drawing2DDrawArc Method (Vector2, Single, Int32, Single, Single, Color, Single)

Draws a Arc.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawArc(      Vector2 center, float radius,
                               int sides,      float pStartAngleDeg,      float pDegrees,
                               Color color,      float lineWidth )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

*pStartAngleDeg*

Type: [SystemSingle](#)

Degrees where to start the arc, being 0° = 15h on a clock, 90° = 18h, 180° = 21h, etc

*pDegrees*

Type: [SystemSingle](#)

Number of degrees to rotate, starting from pStartAngleDeg

*color*

Type: **Color**

Color of the circle

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawArc Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawArc Method (Vector2, Single, Int32, Single, Single, Color, Single, Texture2D)

Draws a Arc.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawArc(      Vector2 center, float radius,
                                int sides,      float pStartAngleDeg,      float pDegrees,
                                Color color,      float lineWidth,      Texture2D
pLineTexture )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

*pStartAngleDeg*

Type: [SystemSingle](#)

Degrees where to start the arc, being 0° = 12h on a clock, 90° = 3h, 180° = 6h, etc

*pDegrees*

Type: [SystemSingle](#)

Number of degrees to rotate, starting from pStartAngleDeg

*color*

Type: **Color**

Color of the circle

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pLineTexture*

Type: **Texture2D**

Custom base texture to fill the segments (null to use default fill)

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawArc Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawArrow Method

Draws an arrow

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawArrow(    Vector2 p1,    Vector2 p2,
    Color color,    float lineWidth,    float pTipWidth,
    float pTipLength )
```

## Parameters

*p1*

Type: **Vector2**

First point of the arrow (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*p2*

Type: **Vector2**

Second point of the arrow (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the arrow

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pTipWidth*

Type: [SystemSingle](#)

Width of the tip, in pixels

*pTipLength*

Type: [SystemSingle](#)

Length of the tip, in pixels

## ▲ See Also

Reference



[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawCircle Method

## ▲ Overload List

	Name	Description
	<a href="#">DrawCircle(Vector2, Single, Int32)</a>	Draws a circle. GUI.ContentColor is assumed to draw the circle
	<a href="#">DrawCircle(Vector2, Single, Int32, Color)</a>	Draws a circle.
	<a href="#">DrawCircle(Vector2, Single, Int32, Color, Single)</a>	Draws a circle.
	<a href="#">DrawCircle(Vector2, Single, Int32, Color, Single, Texture2D)</a>	Draws a circle.

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawCircle Method (Vector2, Single, Int32)

Draws a circle. GUI.ContentColor is assumed to draw the circle

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawCircle( Vector2 center, float radius,  
    int sides )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawCircle Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DDrawCircle Method (Vector2, Single, Int32, Color)

Draws a circle.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawCircle( Vector2 center, float radius,
                             int sides,      Color color )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

*color*

Type: **Color**

Color of the circle

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawCircle Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DDrawCircle Method (Vector2, Single, Int32, Color, Single)

Draws a circle.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawCircle( Vector2 center, float radius,  
                             int sides,      Color color,      float lineWidth )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

*color*

Type: **Color**

Color of the circle

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawCircle Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawCircle Method (Vector2, Single, Int32, Color, Single, Texture2D)

Draws a circle.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawCircle( Vector2 center, float radius,
                               int sides,      Color color,    float lineWidth,
                               Texture2D pLineTexture )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

*color*

Type: **Color**

Color of the circle

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pLineTexture*

Type: **Texture2D**

Custom base texture to fill the segments (null to use default fill)

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawCircle Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawDashedArc Method

Draws a dashed Arc.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedArc(    Vector2 center, float
radius,    int sides,    float pStartAngleDeg,    float
pDegrees, Color color,    float linewidth,    float
pDashMultiplier = 3f )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if

SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

*pStartAngleDeg*

Type: [SystemSingle](#)

Degrees where to start the arc, being 0° = 12h on a clock, 90° = 3h, 180° = 6h, etc

*pDegrees*

Type: [SystemSingle](#)

Number of degrees to rotate, starting from pStartAngleDeg

*color*

Type: **Color**

Color of the circle

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pDashMultiplier* **(Optional)**

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---



# Drawing2DDrawDashedArrow Method

Draws an arrow

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedArrow( Vector2 p1, Vector2  
p2, Color color, float lineWidth, float pTipWidth,  
float pTipLength, float pDashMultiplier = 3f )
```

## Parameters

*p1*

Type: **Vector2**

First point of the arrow (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*p2*

Type: **Vector2**

Second point of the arrow (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the arrow

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pTipWidth*

Type: [SystemSingle](#)

Width of the tip, in pixels

*pTipLength*

Type: [SystemSingle](#)

Length of the tip, in pixels

*pDashMultiplier* (**Optional**)

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawDashedCircle Method

Draws a circle.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedCircle( Vector2 center, float
radius,      int sides,      Color color,      float lineWidth,
      float pDashMultiplier = 3f )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*sides*

Type: [SystemInt32](#)

Number of sides (more sides, more detail and less performance)

*color*

Type: **Color**

Color of the circle

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pDashMultiplier* (**Optional**)

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawDashedHexagon Method

Draws a dashed Hexagon

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedHexagon(      Vector2 center, float
radius,      Color color,      float lineWidth,      float
pDashMultiplier = 3f )
```

## Parameters

*center*

Type: **Vector2**

Center of the Hexagon (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius, in pixels

*color*

Type: **Color**

Color of the pentagon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pDashMultiplier* (**Optional**)

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

Reference





[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawDashedLine Method

## ▲ Overload List

	Name	Description
	<a href="#">DrawDashedLine(Vector2, Vector2)</a>	Draws a dashed line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#">DrawDashedLine(Vector2, Vector2, Single)</a>	Draws a dashed line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#">DrawDashedLine(Vector2, Vector2, Color)</a>	Draws a dashed line between two points.
	<a href="#">DrawDashedLine(Vector2, Vector2, Color, Single, Single)</a>	Draws a dashed line between two points.

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawDashedLine Method (Vector2, Vector2)

Draws a dashed line between two points. GUI.ContentColor is assumed to draw the line

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedLine(    Vector2 pointA, Vector2  
pointB )
```

## Parameters

*pointA*

Type: **Vector2**

First point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pointB*

Type: **Vector2**

Second point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawDashedLine Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DDrawDashedLine Method (Vector2, Vector2, Single)

Draws a dashed line between two points. GUI.ContentColor is assumed to draw the line

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedLine(    Vector2 pointA, Vector2  
pointB,    float width )
```

## Parameters

*pointA*

Type: **Vector2**

First point of the line (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*pointB*

Type: **Vector2**

Second point of the line (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*width*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawDashedLine Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DDrawDashedLine Method (Vector2, Vector2, Color)

Draws a dashed line between two points.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedLine(    Vector2 pointA, Vector2  
pointB,    Color color )
```

## Parameters

*pointA*

Type: **Vector2**

First point of the line (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*pointB*

Type: **Vector2**

Second point of the line (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the line

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawDashedLine Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawDashedLine Method (Vector2, Vector2, Color, Single, Single)

Draws a dashed line between two points.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedLine(    Vector2 p1,    Vector2  
p2,    Color color,    float lineWidth,    float pDashMultiplier  
= 3f )
```

## Parameters

*p1*

Type: **Vector2**

First point of the line (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*p2*

Type: **Vector2**

Second point of the line (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the line

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pDashMultiplier* (**Optional**)

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawDashedLine Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawDashedOctagon Method

Draws a dashed Octagon

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedOctagon(      Vector2 center, float
radius,      Color color,      float lineWidth,      float
pDashMultiplier = 3f )
```

## Parameters

*center*

Type: **Vector2**

Center of the Octagon (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius, in pixels

*color*

Type: **Color**

Color of the pentagon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pDashMultiplier* (**Optional**)

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawDashedPentagon Method

Draws a dashed pentagon

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedPentagon(      Vector2 center, float
radius,      Color color,      float lineWidth,      float
pDashMultiplier = 3f )
```

## Parameters

*center*

Type: **Vector2**

Center of the pentagon (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius, in pixels

*color*

Type: **Color**

Color of the pentagon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pDashMultiplier* (**Optional**)

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also



Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawDashedPolygon Method

Draws a closed or open, dashed polygon (draws a dashed line between each pair of vertices, in order, and if `pLeaveOpen` is false, draws a closing line between the last and the first)

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedPolygon(      Vector2[] pVertices,  
    Color color,      float lineWidth,      bool pLeaveOpen,  
    float pDashMultiplier = 3f )
```

## Parameters

*pVertices*

Type: **Vector2**

Array of vertices of the polygon (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the polygon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pLeaveOpen*

Type: [SystemBoolean](#)

True to leave the polygon open. False to close it with a final line between the last and first vertices

*pDashMultiplier* (**Optional**)

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawDashedQuad Method

Fills a quad with an arbitrary orientation

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedQuad(    Vector2 pCenter,        float
pWidth,        float pHeight,    Color pColor,    float pLineWidth,
        Nullable<float> pOrientationDeg = null,        float
pDashMultiplier = 3f )
```

## Parameters

*pCenter*

Type: **Vector2**

Center of the quad (use relative coordinates to parent only if

SetParentBounds has been used, use absolute screen coords otherwise)

*pWidth*

Type: [SystemSingle](#)

Width of the quad, in pixels

*pHeight*

Type: [SystemSingle](#)

Height of the quad, in pixels

*pColor*

Type: **Color**

Color of the quad

*pLineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pOrientationDeg* (**Optional**)

Type: [SystemNullableSingle](#)

[Missing <param name="pOrientationDeg"/> documentation for  
"M:GraphicDNA.Drawing2D.DrawDashedQuad(UnityEngine.Vector2,System.Single,System.Single,UnityEngine.Color,System.Single,System.Nullable{System.Single},System.Single)"]

*pDashMultiplier* **(Optional)**

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawDashedRect Method

Draws a rectangle (draws the four sides of it)

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedRect(    Rect pRect,    Color  
color,    float lineWidth,    float pDashMultiplier = 3f )
```

## Parameters

*pRect*

Type: **Rect**

Rectangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the rectangle.

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pDashMultiplier* (**Optional**)

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawDashedTriangle Method

Draws a Triangle, defined by 3 points

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawDashedTriangle(    Vector2 pA,  
    Vector2 pB,    Vector2 pC,    Color color,    float  
lineWidth,    float pDashMultiplier = 3f )
```

## Parameters

*pA*

Type: **Vector2**

First point of the triangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pB*

Type: **Vector2**

Second point of the triangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pC*

Type: **Vector2**

Third point of the triangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the triangle

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pDashMultiplier* (**Optional**)

Type: [SystemSingle](#)

Dash frequency multiplier (min = 0.1, max = 10, default = 3)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)


---



# Drawing2DDrawHexagon Method

---

## ▲ Overload List

	Name	Description
	<a href="#">DrawHexagon(Vector2, Single, Color, Single)</a>	Draws a Hexagon
	<a href="#">DrawHexagon(Vector2, Single, Color, Single, Texture2D)</a>	Draws a Hexagon

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawHexagon Method (Vector2, Single, Color, Single)

Draws a Hexagon

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawHexagon( Vector2 center, float radius,
                                Color color,      float lineWidth )
```

## Parameters

*center*

Type: **Vector2**

Center of the Hexagon (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius, in pixels

*color*

Type: **Color**

Color of the pentagon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawHexagon Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DDrawHexagon Method (Vector2, Single, Color, Single, Texture2D)

Draws a Hexagon

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawHexagon( Vector2 center, float radius,  
                                Color color,    float lineWidth,    Texture2D  
pLineTexture )
```

## Parameters

*center*

Type: **Vector2**

Center of the Hexagon (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius, in pixels

*color*

Type: **Color**

Color of the pentagon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pLineTexture*

Type: **Texture2D**

Custom texture to use as base fill (null to use default fill)

## ▴ See Also

Reference

[Drawing2D Class](#)






[DrawHexagon Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawLine Method

## ▲ Overload List

	Name	Description
	<a href="#">DrawLine(Vector2, Vector2)</a>	Draws a line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#">DrawLine(Vector2, Vector2, Single)</a>	Draws a line between two points. GUI.ContentColor is assumed to draw the line
	<a href="#">DrawLine(Vector2, Vector2, Color)</a>	Draws a line between two points.
	<a href="#">DrawLine(Vector2, Vector2, Color, Single)</a>	Draws a line between two points.
	<a href="#">DrawLine(Vector2, Vector2, Color, Single, Texture2D, NullableVector2)</a>	Draws a line between two points, using a customized base texture to fill the segment

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawLine Method (Vector2, Vector2)

Draws a line between two points. GUI.ContentColor is assumed to draw the line

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawLine(    Vector2 pointA, Vector2 pointB )
```

### Parameters

*pointA*

Type: **Vector2**

First point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pointB*

Type: **Vector2**

Second point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawLine Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawLine Method (Vector2, Vector2, Single)

Draws a line between two points. GUI.ContentColor is assumed to draw the line

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawLine(    Vector2 pointA, Vector2 pointB,  
    float width )
```

## Parameters

*pointA*

Type: **Vector2**

First point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pointB*

Type: **Vector2**

Second point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*width*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawLine Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DDrawLine Method (Vector2, Vector2, Color)

Draws a line between two points.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawLine(    Vector2 pointA, Vector2 pointB,  
    Color color )
```

## Parameters

*pointA*

Type: **Vector2**

First point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pointB*

Type: **Vector2**

Second point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the line

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawLine Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawLine Method (Vector2, Vector2, Color, Single)

Draws a line between two points.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawLine(    Vector2 pointA, Vector2 pointB,  
                             Color color,    float width )
```

## Parameters

*pointA*

Type: **Vector2**

First point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pointB*

Type: **Vector2**

Second point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the line

*width*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawLine Overload](#)



# Drawing2DDrawLine Method (Vector2, Vector2, Color, Single, Texture2D, NullableVector2)

Draws a line between two points, using a customized base texture to fill the segment

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawLine(    Vector2 p1,    Vector2 p2,
                             Color color,    float lineWidth,    Texture2D
pOverrideTexture = null,    Nullable<Vector2> pTilingMultiplier =
null )
```

## Parameters

*p1*

Type: **Vector2**

First point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*p2*

Type: **Vector2**

Second point of the line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the line

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pOverrideTexture* (**Optional**)

Type: **Texture2D**

Custom texture to use as base fill (null to use default fill)

*pTilingMultiplier* (**Optional**)

Type: [SystemNullable](#)**Vector2**

Number of repetitions of the texture in U,V (null to use default, with 1 repetition in each direction)

## ▲ See Also

Reference

[Drawing2D Class](#)




[DrawLine Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawLines Method

## ▲ Overload List

	Name	Description
	<a href="#">DrawLines(Vector2, Color)</a>	Draws lines with thickness == 1, using GUI 2D points
	<a href="#">DrawLines(Vector3, Color)</a>	Draws lines with thickness == 1, using OpenGL Vector3 points in homogeneous coords. Please use the BuildGLVertexBuffer method to convert from GUI coordinates to OpenGL homogeneous coordinates
	<a href="#">DrawLines(Vector2, Color, Single, Texture2D, NullableVector2, Boolean)</a>	Draws lines, using OpenGL Vector3 points in homogeneous coords. Please use the BuildGLVertexBuffer method to convert from GUI coordinates to OpenGL homogeneous coordinates

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawLines Method (Vector2, Color)

Draws lines with thickness == 1, using GUI 2D points

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawLines(    Vector2[] pPointPairs,    Color pColor )
```

## Parameters

*pPointPairs*

Type: **Vector2**

Pairs of points, with the start-end points for each line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pColor*

Type: **Color**

Color of all lines

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawLines Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawLines Method (Vector3, Color)

Draws lines with thickness == 1, using OpenGL Vector3 points in homogeneous coords. Please use the BuildGLVertexBuffer method to convert from GUI coordinates to OpenGL homogeneous coordinates

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawLines(    Vector3[] vertexBuffer,    Color pColor )
```

## Parameters

*vertexBuffer*

Type: **Vector3**

Pairs of points in homogeneous coordinates, valid for OpenGL rendering with an Ortho projection matrix

*pColor*

Type: **Color**

Color of all lines

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawLines Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DDrawLines Method (Vector2, Color, Single, Texture2D, NullableVector2, Boolean)

Draws lines, using OpenGL Vector3 points in homogeneous coords. Please use the BuildGLVertexBuffer method to convert from GUI coordinates to OpenGL homogeneous coordinates

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawLines(    Vector2[] pPointPairs,    Color
pColor,    float lineWidth,    Texture2D pOverrideTexture =
null,    Nullable<Vector2> pTilingMultiplier = null,    bool
pFixLineCorners = true )
```

## Parameters

*pPointPairs*

Type: **Vector2**

Pairs of points, with the start-end points for each line (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pColor*

Type: **Color**

Color of all lines

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pOverrideTexture* (**Optional**)

Type: **Texture2D**

Texture to overlay (null to use only color)

*pTilingMultiplier* **(Optional)**

Type: [SystemNullableVector2](#)

Tiling multiplier or null for default tiling (1, 1)

*pFixLineCorners* **(Optional)**

Type: [SystemBoolean](#)

If true, fixes line corners considering the line to be continuous. Looks better but it's significantly slower (default = true)

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawLines Overload](#)


[GraphicDNA Namespace](#)

---

# Drawing2DDrawOctagon Method

---

## ▲ Overload List

	Name	Description
	<a href="#">DrawOctagon(Vector2, Single, Color, Single)</a>	Draws a Octagon
	<a href="#">DrawOctagon(Vector2, Single, Color, Single, Texture2D)</a>	Draws a Octagon

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawOctagon Method (Vector2, Single, Color, Single)

Draws a Octagon

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawOctagon( Vector2 center, float radius,
                                Color color,      float lineWidth )
```

## Parameters

*center*

Type: **Vector2**

Center of the Octagon (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius, in pixels

*color*

Type: **Color**

Color of the pentagon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawOctagon Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DDrawOctagon Method (Vector2, Single, Color, Single, Texture2D)

Draws a Octagon

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawOctagon( Vector2 center, float radius,  
                                Color color,    float lineWidth,    Texture2D  
pLineTexture )
```

## Parameters

*center*

Type: **Vector2**

Center of the Octagon (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius, in pixels

*color*

Type: **Color**

Color of the pentagon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pLineTexture*

Type: **Texture2D**

Custom texture to use as base fill (null to use default fill)

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawOctagon Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawPentagon Method

---

## ▲ Overload List

	Name	Description
	<a href="#">DrawPentagon(Vector2, Single, Color, Single)</a>	Draws a pentagon
	<a href="#">DrawPentagon(Vector2, Single, Color, Single, Texture2D)</a>	Draws a pentagon

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---



# Drawing2DDrawPentagon Method (Vector2, Single, Color, Single)

Draws a pentagon

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawPentagon(      Vector2 center, float  
radius,      Color color,      float lineWidth )
```

## Parameters

*center*

Type: **Vector2**

Center of the pentagon (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius, in pixels

*color*

Type: **Color**

Color of the pentagon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawPentagon Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DDrawPentagon Method (Vector2, Single, Color, Single, Texture2D)

Draws a pentagon

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawPentagon(      Vector2 center, float
radius,      Color color,      float lineWidth,      Texture2D
pLineTexture )
```

## Parameters

*center*

Type: **Vector2**

Center of the pentagon (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius, in pixels

*color*

Type: **Color**

Color of the pentagon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pLineTexture*

Type: **Texture2D**

Custom texture to use as base fill (null to use default fill)

## ▴ See Also

Reference

[Drawing2D Class](#)

[DrawPentagon Overload](#)


[GraphicDNA Namespace](#)

---

# Drawing2DDrawPoint Method

---

## ▲ Overload List

	Name	Description
 	<a href="#">DrawPoint(Vector2, Color)</a>	Draws a point
 	<a href="#">DrawPoint(Vector2, Color, Single)</a>	Draws a point

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawPoint Method (Vector2, Color)

Draws a point

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawPoint(    Vector2 pPoint, Color color )
```

## Parameters

*pPoint*

Type: **Vector2**

Coordinates of the point (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the point

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawPoint Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawPoint Method (Vector2, Color, Single)

Draws a point

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawPoint(    Vector2 pPoint, Color color,
                                float pSize )
```

## Parameters

*pPoint*

Type: **Vector2**

Coordinates of the point (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the point

*pSize*

Type: [SystemSingle](#)

Size of the point, in pixels

## ▲ See Also

### Reference






[Drawing2D Class](#)

[DrawPoint Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawPolygon Method

## ▲ Overload List

Name	Description
 <a href="#">DrawPolygon(Vector2)</a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first) Assumes GUI.ContentColor
 <a href="#">DrawPolygon(Vector2, Color)</a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)
 <a href="#">DrawPolygon(Vector2, Color, Single)</a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)
 <a href="#">DrawPolygon(Vector2, Color, Single, Texture2D)</a>	Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)
 <a href="#">DrawPolygon(Vector2, Color, Single, Texture2D, Boolean, Boolean)</a>	Draws a closed or open polygon (draws a line between each pair of vertices, in order, and if pLeaveOpen is false, draws a closing line between the last and the first)

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)





# Drawing2DDrawPolygon Method (Vector2)

Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first) Assumes GUI.ContentColor

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawPolygon( Vector2[] pVertices )
```

## Parameters

*pVertices*

Type: **Vector2**

Array of vertices of the polygon (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawPolygon Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawPolygon Method (Vector2, Color)

Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawPolygon( Vector2[] pVertices, Color  
color )
```

## Parameters

*pVertices*

Type: **Vector2**

Array of vertices of the polygon (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the polygon

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawPolygon Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawPolygon Method (Vector2, Color, Single)

Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawPolygon( Vector2[] pVertices, Color  
color, float lineWidth )
```

## Parameters

*pVertices*

Type: **Vector2**

Array of vertices of the polygon (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the polygon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawPolygon Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawPolygon Method (Vector2, Color, Single, Texture2D)

Draws a closed polygon (draws a line between each pair of vertices, in order, and one closing line between the last and the first)

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawPolygon( Vector2[] pVertices, Color  
color, float lineWidth, Texture2D pLineTexture )
```

## Parameters

*pVertices*

Type: **Vector2**

Array of vertices of the polygon (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the polygon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pLineTexture*

Type: **Texture2D**

Custom texture to use as base fill (null to use default fill)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawPolygon Overload](#)



# Drawing2DDrawPolygon Method (Vector2, Color, Single, Texture2D, Boolean, Boolean)

Draws a closed or open polygon (draws a line between each pair of vertices, in order, and if pLeaveOpen is false, draws a closing line between the last and the first)

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawPolygon( Vector2[] pVertices, Color  
color, float lineWidth, Texture2D pLineTexture, bool  
pLeaveOpen, bool pFixLineCorners = true )
```

## Parameters

*pVertices*

Type: **Vector2**

Array of vertices of the polygon, plain list, not point pairs (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the polygon

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pLineTexture*

Type: **Texture2D**

Custom texture to use as base fill (null to use default fill)

*pLeaveOpen*

Type: [SystemBoolean](#)

True to leave the polygon open. False to close it with a final line between the last and first vertices

*pFixLineCorners* **(Optional)**

Type: [SystemBoolean](#)

If true, fixes line corners considering the line to be continuous. Looks better but it's significantly slower (default = true)

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawPolygon Overload](#)

[GraphicDNA Namespace](#)

---



# Drawing2DDrawQuad Method

Fills a quad with an arbitrary orientation

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawQuad(    Vector2 pCenter,    float
pWidth,    float pHeight,    Color pColor,    float pLineWidth,
    Nullable<float> pOrientationDeg = null,    Texture2D
pOverrideTexture = null )
```

## Parameters

*pCenter*

Type: **Vector2**

Center of the quad (use relative coordinates to parent only if

SetParentBounds has been used, use absolute screen coords otherwise)

*pWidth*

Type: [SystemSingle](#)

Width of the quad, in pixels

*pHeight*

Type: [SystemSingle](#)

Height of the quad, in pixels

*pColor*

Type: **Color**

Color of the quad

*pLineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pOrientationDeg* (**Optional**)

Type: [SystemNullableSingle](#)

[Missing <param name="pOrientationDeg"/> documentation for  
"M:GraphicDNA.Drawing2D.DrawQuad(UnityEngine.Vector2,System.Single,System.Single,UnityEngine.Color,System.Single,System.Nullable{System.Single},UnityEngine.Texture2D)"]

*pOverrideTexture* **(Optional)**

Type: **Texture2D**

Fill texture to overlay, or null to use color only

## ▲ See Also

Reference





[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawRect Method

## ▲ Overload List

	Name	Description
	<a href="#">DrawRect(Rect)</a>	Draws a rectangle (draws the four sides of it). GUI.ContentColor is assumed
	<a href="#">DrawRect(Rect, Color)</a>	Draws a rectangle (draws the four sides of it)
	<a href="#">DrawRect(Rect, Color, Single)</a>	Draws a rectangle (draws the four sides of it)
	<a href="#">DrawRect(Rect, Color, Single, Texture2D)</a>	Draws a rectangle (draws the four sides of it)

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawRect Method (Rect)

Draws a rectangle (draws the four sides of it). GUI.ContentColor is assumed

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawRect(    Rect pRect )
```

## Parameters

*pRect*

Type: **Rect**

Rectangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawRect Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawRect Method (Rect, Color)

Draws a rectangle (draws the four sides of it)

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawRect(    Rect pRect,    Color color )
```

## Parameters

*pRect*

Type: **Rect**

Rectangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the rectangle.

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawRect Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawRect Method (Rect, Color, Single)

Draws a rectangle (draws the four sides of it)

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawRect(    Rect pRect,    Color color,
    float lineWidth )
```

## Parameters

*pRect*

Type: **Rect**

Rectangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the rectangle.

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawRect Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawRect Method (Rect, Color, Single, Texture2D)

Draws a rectangle (draws the four sides of it)

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawRect(    Rect pRect,    Color color,
                             float lineWidth,    Texture2D pTexture )
```

## Parameters

*pRect*

Type: **Rect**

Rectangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the rectangle.

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pTexture*

Type: **Texture2D**

Custom texture to use as base fill (null to use default fill)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawRect Overload](#)

[GraphicDNA Namespace](#)





# Drawing2DDrawRects Method

Draws Rects using OpenGL, ideal for performance critical situations. Please note: Rects drawn using OpenGL are not affected by clipping

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawRects(    IList<Rect> pRects,    Color pColor,    bool pClearFrameBufferColor = false )
```

## Parameters

*pRects*

Type: [System.Collections.Generic.IList<Rect>](#)

List of rectangles to draw (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pColor*

Type: **Color**

Color of all rects

*pClearFrameBufferColor* (**Optional**)

Type: [System.Boolean](#)

True to clear the whole frame buffer, false otherwise

## ▲ See Also

### Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawText Method

---

## ▲ Overload List

	Name	Description
	<a href="#">DrawText(String, Single, Single, GUIStyle)</a>	Draws text with a customizable style
	<a href="#">DrawText(String, Vector2, Int32, Color)</a>	Draws text
	<a href="#">DrawText(String, Single, Single, Int32, Color)</a>	Draws text

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawText Method (String, Single, Single, GUIStyle)

Draws text with a customizable style

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawText(    string pText,    float pX,    float pY,    GUIStyle pTextStyle )
```

## Parameters

*pText*

Type: [SystemString](#)

Text to draw

*pX*

Type: [SystemSingle](#)

X coord of the TopLeft corner of the text (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pY*

Type: [SystemSingle](#)

Y coord of the TopLeft corner of the text (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pTextStyle*

Type: **GUIStyle**

Custom Text Style

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawText Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawText Method (String, Vector2, Int32, Color)

Draws text

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawText(    string pText,    Vector2 pPos,
    int pFontSize,    Color pColor )
```

## Parameters

*pText*

Type: [SystemString](#)

Text to draw

*pPos*

Type: **Vector2**

Coords of the TopLeft corner of the text (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pFontSize*

Type: [SystemInt32](#)

Font size

*pColor*

Type: **Color**

Color of the text

## ▲ See Also

### Reference

[Drawing2D Class](#)

[DrawText Overload](#)



# Drawing2DDrawText Method (String, Single, Single, Int32, Color)

Draws text

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawText(    string pText,    float pX, float  
pY, int pFontSize, Color pColor )
```

### Parameters

*pText*

Type: [SystemString](#)

Text to draw

*pX*

Type: [SystemSingle](#)

X coord of the TopLeft corner of the text (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pY*

Type: [SystemSingle](#)

Y coord of the TopLeft corner of the text (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pFontSize*

Type: [SystemInt32](#)

Font size

*pColor*

Type: **Color**

Color of the text

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawText Overload](#)

[GraphicDNA Namespace](#)

---



# Drawing2DDrawTexture Method

Draws a texture

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawTexture( Rect pRect,      Color color,
                               Texture2D pTexture )
```

## Parameters

*pRect*

Type: **Rect**

Rectangle in the screen to draw the texture (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color that will be multiplied to the texture (white for no effect)

*pTexture*

Type: **Texture2D**

Texture to draw

## ▲ See Also

### Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDrawTriangle Method

---

## ▲ Overload List

	Name	Description
 	<a href="#">DrawTriangle(Vector2, Vector2, Vector2, Color, Single)</a>	Draws a Triangle, defined by 3 points
 	<a href="#">DrawTriangle(Vector2, Vector2, Vector2, Color, Single, Texture2D)</a>	Draws a Triangle, defined by 3 points

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawTriangle Method (Vector2, Vector2, Vector2, Color, Single)

Draws a Triangle, defined by 3 points

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawTriangle(    Vector2 pA,    Vector2  
pB,    Vector2 pC,    Color color,    float lineWidth )
```

## Parameters

*pA*

Type: **Vector2**

First point of the triangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pB*

Type: **Vector2**

Second point of the triangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pC*

Type: **Vector2**

Third point of the triangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the triangle

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawTriangle Overload](#)

[GraphicDNA Namespace](#)

---

# Drawing2DDrawTriangle Method (Vector2, Vector2, Vector2, Color, Single, Texture2D)

Draws a Triangle, defined by 3 points

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void DrawTriangle(      Vector2 pA,      Vector2  
pB,  Vector2 pC,      Color color,      float lineWidth,  
      Texture2D pLineTexture )
```

## Parameters

*pA*

Type: **Vector2**

First point of the triangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pB*

Type: **Vector2**

Second point of the triangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pC*

Type: **Vector2**

Third point of the triangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the triangle

*lineWidth*

Type: [SystemSingle](#)

Line thickness, in pixels

*pLineTexture*

Type: **Texture2D**

Custom texture to use as base fill (null to use default fill)

## ▲ See Also

Reference

[Drawing2D Class](#)

[DrawTriangle Overload](#)




[GraphicDNA Namespace](#)

---

# Drawing2DFillCircle Method

---

## ▲ Overload List

	Name	Description
	<a href="#">FillCircle(Vector2, Single)</a>	Fills a circle, using GUI.Content color
	<a href="#">FillCircle(Vector2, Single, Color)</a>	Fills a circle
	<a href="#">FillCircle(Vector2, Single, Color, Texture2D)</a>	Fills a circle

---

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DFillColor Method (Vector2, Single)

Fills a circle, using GUI.Content color

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void FillCircle( Vector2 center, float radius )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

## ▲ See Also

### Reference

[Drawing2D Class](#)

[FillColor Overload](#)

[GraphicDNA Namespace](#)



# Drawing2DFillColor Method (Vector2, Single, Color)

Fills a circle

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void FillCircle( Vector2 center, float radius,
                             Color color )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*color*

Type: **Color**

Color of the circle

## ▲ See Also

### Reference

[Drawing2D Class](#)

[FillColor Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DFillColor Method (Vector2, Single, Color, Texture2D)

Fills a circle

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void FillCircle( Vector2 center, float radius,
                             Color color, Texture2D pCircleTexture )
```

## Parameters

*center*

Type: **Vector2**

Center of the circle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*radius*

Type: [SystemSingle](#)

Radius of the circle, in Pixels

*color*

Type: **Color**

Color of the circle

*pCircleTexture*

Type: **Texture2D**

Custom base texture to perform the fill (must be round texture to keep the circle shape)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[FillColor Overload](#)



# Drawing2DFillQuad Method

Fills a quad with an arbitrary orientation

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void FillQuad(    Vector2 pCenter,    float
pWidth,    float pHeight,    Color pColor,    Nullable<float>
pOrientationDeg = null,    Texture2D pOverrideTexture = null,
    Nullable<Vector2> pTilingMultiplier = null )
```

## Parameters

*pCenter*

Type: **Vector2**

Center of the quad (use relative coordinates to parent only if

SetParentBounds has been used, use absolute screen coords otherwise)

*pWidth*

Type: [SystemSingle](#)

Width of the quad, in pixels

*pHeight*

Type: [SystemSingle](#)

Height of the quad, in pixels

*pColor*

Type: **Color**

Color of the quad

*pOrientationDeg* (**Optional**)

Type: [SystemNullableSingle](#)

[Missing <param name="pOrientationDeg"/> documentation for

"M:GraphicDNA.Drawing2D.FillQuad(UnityEngine.Vector2,System.Single,System.Single,UnityEngine.Color,System.Nullable{System.Single},UnityEngine.Texture2D,System.Nullable{UnityEngine.Vector2})"]

*pOverrideTexture* **(Optional)**

Type: **Texture2D**

Fill texture to overlay, or null to use color only

*pTilingMultiplier* **(Optional)**

Type: [SystemNullable](#)**Vector2**

Tiling multiplier or null for default tiling (1, 1)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DFillQuads Method

Fills quads that can have any orientation. Each quad is defined by its 4 vertices in the following order: BL, TL, TR, BR

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void FillQuads(    Vector2[] pVertices, Color
color,        Texture2D pOverrideTexture = null,
        Nullable<Vector2> pTilingMultiplier = null )
```

## Parameters

*pVertices*

Type: **Vector2**

Vertices of the Quad in the following order: BL, TL, TR, BR (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the quad

*pOverrideTexture* (**Optional**)

Type: **Texture2D**

Texture to overlay, if any

*pTilingMultiplier* (**Optional**)

Type: [SystemNullableVector2](#)

Tiling multiplier or null for default tiling (1, 1)

## ▲ See Also

### Reference

[Drawing2D Class](#)




[GraphicDNA Namespace](#)



# Drawing2DFillRect Method

---

## ▲ Overload List

	Name	Description
	<a href="#">FillRect(Rect)</a>	Fills a rectangle. GUI.ContentColor is assumed
	<a href="#">FillRect(Rect, Color)</a>	Fills a rectangle.
	<a href="#">FillRect(Rect, Color, Texture2D, NullableVector2)</a>	Fills a rectangle.

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---



# Drawing2DFillRect Method (Rect)

Fills a rectangle. GUI.ContentColor is assumed

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void FillRect(    Rect pRect )
```

## Parameters

*pRect*

Type: **Rect**

Rectangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

## ▲ See Also

### Reference

[Drawing2D Class](#)

[FillRect Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DFillRect Method (Rect, Color)

Fills a rectangle.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void FillRect(    Rect pRect,    Color color )
```

## Parameters

*pRect*

Type: **Rect**

Rectangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the rectangle

## ▲ See Also

### Reference

[Drawing2D Class](#)

[FillRect Overload](#)

[GraphicDNA Namespace](#)

# Drawing2DFillRect Method (Rect, Color, Texture2D, NullableVector2)

Fills a rectangle.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void FillRect(    Rect pRectangle,    Color  
pColor,    Texture2D pTexture = null,    Nullable<Vector2>  
pTilingMultiplier = null )
```

## Parameters

*pRectangle*

Type: **Rect**

Rectangle (use relative coordinates to parent only if SetParentBounds has been used, use absolute screen coords otherwise)

*pColor*

Type: **Color**

Color of the rectangle

*pTexture* (**Optional**)

Type: **Texture2D**

Custom texture to use as base fill (null to use default fill)

*pTilingMultiplier* (**Optional**)

Type: [SystemNullableVector2](#)

Number of repetitions of the texture in U,V (null to use default (1, 1))

## ▲ See Also

### Reference

[Drawing2D Class](#)

[FillRect Overload](#)



# Drawing2DFillTriangle Method

[Missing <summary> documentation for

"M:GraphicDNA.Drawing2D.FillTriangle(UnityEngine.Vector2,UnityEngine.Vector2,UnityEngine.Vector2,UnityEngine.Color,UnityEngine.Texture2D,System.Nullable{UnityEngine.Vector2})"]

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void FillTriangle(      Vector2 pA,      Vector2  
pB,  Vector2 pC,      Color color,      Texture2D pOverrideTexture  
= null,      Nullable<Vector2> pTilingMultiplier = null )
```

## Parameters

*pA*

Type: **Vector2**

First point of the triangle (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*pB*

Type: **Vector2**

Second point of the triangle (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*pC*

Type: **Vector2**

Third point of the triangle (use relative coordinates to parent only if  
SetParentBounds has been used, use absolute screen coords otherwise)

*color*

Type: **Color**

Color of the Triangle

*pOverrideTexture* (**Optional**)

Type: **Texture2D**

Texture to overlay (null to use color only)

*pTilingMultiplier* (**Optional**)

Type: [SystemNullable](#)**Vector2**

Number of repetitions of the texture in U,V (null to use default (1, 1))

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DGetWorldRect Method

Returns the world position and size of a UI element (provided its RectTransform)  
The returned position corresponds to the Top-Left corner of the element.

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static Rect GetWorldRect(          RectTransform pTransform )
```

### Parameters

*pTransform*

Type: **RectTransform**

RectTransform of the UI Element

### Return Value

Type: **Rect**

[Missing <returns> documentation for  
"M:GraphicDNA.Drawing2D.GetWorldRect(UnityEngine.RectTransform)"]

## ▲ See Also

### Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

---

# Drawing2DSetParentBounds Method

Sets the coordinate system for drawing operations so the (0,0) is the top-left corner of the group. If pClip is enabled, all controls are clipped to the group. Groups CANNOT be nested Please note: Do not use GUI.BeginGroup to achieve this, as that might result in a malfunction of the drawing features

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static void SetParentBounds( Rect pRect )
```

## Parameters

*pRect*

Type: **Rect**

Parent rectangle to use as a reference

## ▲ See Also

### Reference

[Drawing2D Class](#)







[GraphicDNA Namespace](#)



# Drawing2D Properties

The [Drawing2D](#) type exposes the following members.

## ▲ Properties

	Name	Description
 <b>S</b>	<a href="#">DefaultCircleTexture</a>	Default circle texture used for filling round shapes
 <b>S</b>	<a href="#">DefaultDashedLineTexture</a>	Default pixel texture used for filling dashed shapes
 <b>S</b>	<a href="#">DefaultTextStyle</a>	Default text style to render text
 <b>S</b>	<a href="#">DefaultTriangleTexture</a>	Default Triangle texture used for filling triangle shapes
 <b>S</b>	<a href="#">MaterialColorAndTexture</a>	Returns the default material to fill shapes
 <b>S</b>	<a href="#">MaterialColorOnly</a>	Returns the default material to fill shapes

[Top](#)

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDefaultCircleTexture Property

Default circle texture used for filling round shapes

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static Texture2D DefaultCircleTexture { get; }
```

Property Value

Type: **Texture2D**

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDefaultDashedLineTexture Property

Default pixel texture used for filling dashed shapes

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static Texture2D DefaultDashedLineTexture { get; }
```

Property Value

Type: **Texture2D**

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDefaultTextStyle Property

Default text style to render text

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static GUIStyle DefaultTextStyle { get; }
```

Property Value

Type: **GUIStyle**

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DDefaultTriangleTexture Property

Default Triangle texture used for filling triangle shapes

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static Texture2D DefaultTriangleTexture { get; }
```

Property Value

Type: **Texture2D**

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DMaterialColorAndTexture Property

Returns the default material to fill shapes

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static Material MaterialColorAndTexture { get; }
```

Property Value

Type: **Material**

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)

# Drawing2DMaterialColorOnly Property

Returns the default material to fill shapes

**Namespace:** [GraphicDNA](#)

**Assembly:** 2DShaper (in 2DShaper.dll) Version: 1.0.0.0 (1.0.0.0)

## ▲ Syntax

C#

[Copy](#)

VB

C++

F#

```
public static Material MaterialColorOnly { get; }
```

Property Value

Type: **Material**

## ▲ See Also

Reference

[Drawing2D Class](#)

[GraphicDNA Namespace](#)