

UTFT_Geometry

Add-on Library for UTFT: Geometric functions

Manual

The logo for Rinky-Dink Electronics features the company name in a stylized, glowing cyan font with a 3D effect. The text is set against a dark background that includes a close-up of a green printed circuit board (PCB) with various electronic components and traces visible.

Rinky-Dink Electronics

Introduction:

This library is an add-on to UTFT and will not work on its own.

This library adds some geometric functions to UTFT which are not included in the main library.

You can always find the latest version of the library at <http://www.RinkyDinkElectronics.com/>

For version information, please refer to **version.txt**.

This library is licensed under a **CC BY-NC-SA 3.0** (Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported) License.

For more information see: <http://creativecommons.org/licenses/by-nc-sa/3.0/>

FUNCTIONS:

UTFT_Geometry(UTFT);	
The main class constructor.	
Parameters:	UTFT: A reference to an already created UTFT object (<i>remember the '&' in front of the object name</i>)
Usage:	<code>UTFT_Geometry geo(&myGLCD);</code> // Start an instance of UTFT_Geometry pointing to the UTFT object myGLCD
Notes:	Remember that all functions in UTFT_Geometry must be called through the UTFT_Geometry object (geo in this example), while all UTFT functions still must be called through the UTFT object (myGLCD in this example)

drawTriangle(x1, y1, x2, y2, x3, y3);	
Draw a triangle with the specified corners.	
Parameters:	x1: x-coordinate of the first corner y1: y-coordinate of the first corner x2: x-coordinate of the second corner y2: y-coordinate of the second corner x3: x-coordinate of the third corner y3: y-coordinate of the third corner
Usage:	<code>geo. drawTriangle(159,119,319,119,319,239);</code> // Draw a triangle

fillTriangle(x1, y1, x2, y2, x3, y3);	
Draw a filled triangle with the specified corners.	
Parameters:	x1: x-coordinate of the first corner y1: y-coordinate of the first corner x2: x-coordinate of the second corner y2: y-coordinate of the second corner x3: x-coordinate of the third corner y3: y-coordinate of the third corner
Usage:	<code>geo. fillTriangle(159,119,319,119,319,239);</code> // Draw a filled triangle

drawArc(x, y, r, startAngle, endAngle[, thickness]);	
Draw an arc.	
Parameters:	x : x-coordinate for the center of the imaginary circle y : y-coordinate for the center of the imaginary circle r : Radius of the imaginary circle startAngle: Angle from the center of the imaginary circle to start drawing the arc endAngle : Angle from the center of the imaginary circle to stop drawing the arc thickness : <optional> Thickness of the arc in pixels. Default is 1 pixel.
Usage:	<code>geo. drawArc(159,239,100,-75,75);</code> // Draw an arc
Notes:	0° is at the top of the imaginary circle

drawPie(x, y, r, startAngle, endAngle);	
Draw an arc with lines to the center of the imaginary circle.	
Parameters:	x : x-coordinate for the center of the imaginary circle y : y-coordinate for the center of the imaginary circle r : Radius of the imaginary circle startAngle: Angle from the center of the imaginary circle to start drawing the arc endAngle : Angle from the center of the imaginary circle to stop drawing the arc
Usage:	<code>geo. drawPie(159,50,100,135,225);</code> // Draw a pie shape
Notes:	0° is at the top of the imaginary circle