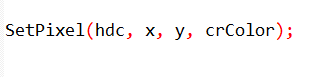
DRAWING LINES AND DOTS

Chapter 5 Part 2

SetPixel Function

The SetPixel function is a fundamental building block for creating graphics in Windows GDI. It allows you to set the color of a specific pixel at a specified x and y coordinate. The syntax is as follows:



hdc: A handle to the device context (DC) that represents the drawing surface.

x: The x-coordinate of the pixel to be set.

y: The y-coordinate of the pixel to be set.

crColor: A COLORREF value representing the desired color of the pixel.

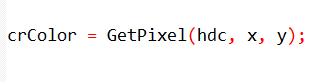
When called, SetPixel sets the specified pixel to the given color.

If the specified color cannot be represented on the video display, the function sets the pixel to the nearest pure non-dithered color and returns that value.

GetPixel Function

The GetPixel function is another essential tool for working with pixels in Windows GDI.

It retrieves the color of a specific pixel at a specified x and y coordinate. The syntax is as follows:



hdc: A handle to the DC that represents the drawing surface.

x: The x-coordinate of the pixel to retrieve the color from.

y: The y-coordinate of the pixel to retrieve the color from.

The GetPixel function returns a COLORREF value representing the color of the specified pixel.

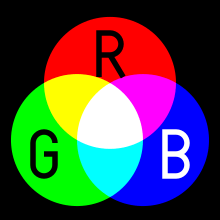
Limitations of SetPixel and GetPixel

While SetPixel and GetPixel provide direct access to individual pixels, they are not commonly used for complex graphics operations. This is primarily due to performance considerations.

Performance Overhead: Drawing complex shapes using SetPixel involves calling the function repeatedly for each pixel, which can be inefficient. Higher-level GDI functions, such as LineTo and Polyline, are optimized for efficient line drawing and utilize specialized hardware acceleration when available.



Device-Dependent Colors: COLORREF values represent colors in a device-dependent manner. Using SetPixel and GetPixel directly can lead to color discrepancies between different display devices.



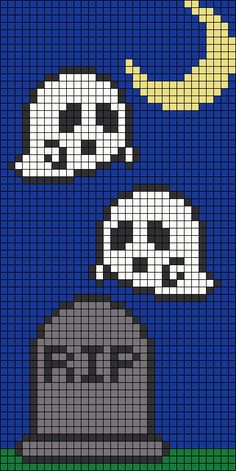
Alternative Graphics Approaches

In general, it is recommended to use higher-level GDI functions whenever possible.

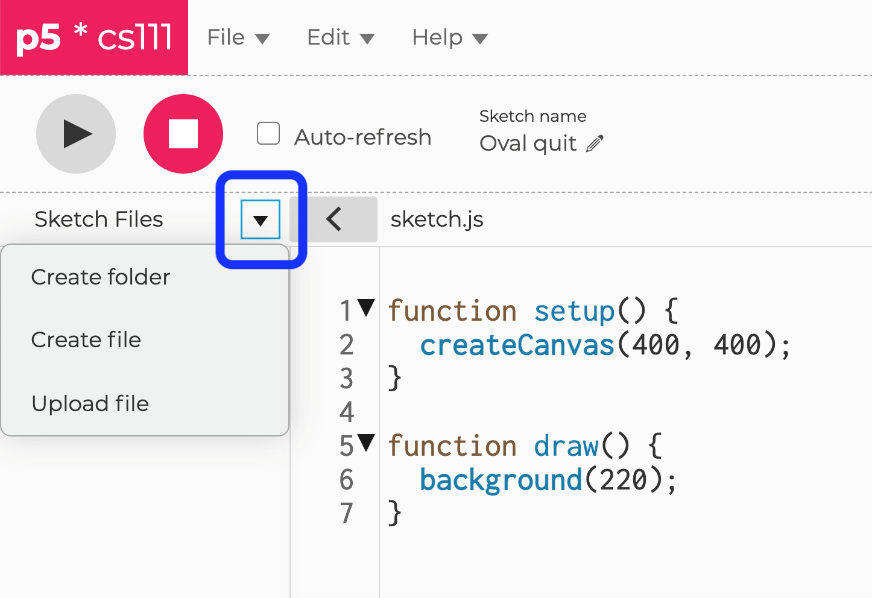
These functions provide better performance, hardware acceleration, and device-independent color handling.

For specialized cases where direct pixel manipulation is required, there are alternative approaches that offer more efficiency and flexibility.

Raster Operations: GDI provides raster operations (Rops) that allow for efficient manipulation of pixel patterns. These operations can be combined with SetPixel and GetPixel to achieve more complex graphics effects.



Custom Drawing Functions: Developers can create their own drawing functions that employ optimized algorithms and utilize hardware acceleration when available. This approach can be particularly beneficial for specialized graphics tasks.

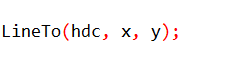


Line Drawing Functions in Windows GDI

Windows GDI provides a variety of functions for drawing straight lines. These functions offer different levels of flexibility and control over the line drawing process.

LineTo Function

The LineTo function draws a single straight line from the current pen position to the specified endpoint. The syntax is as follows:



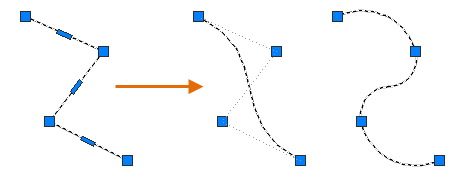
hdc: A handle to the device context (DC) that represents the drawing surface.

x: The x-coordinate of the endpoint of the line.

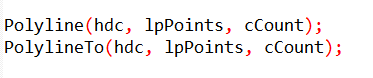
y: The y-coordinate of the endpoint of the line.

After calling LineTo, the current pen position is updated to the endpoint of the drawn line.

Polyline and PolylineTo Functions



The Polyline and PolylineTo functions draw a series of connected straight lines. Polyline defines an open polyline, while PolylineTo defines a closed polyline. The syntax for both functions is as follows:

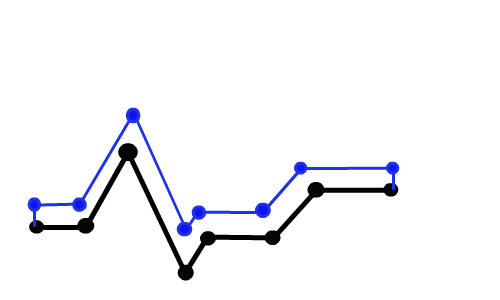


hdc: A handle to the DC that represents the drawing surface.

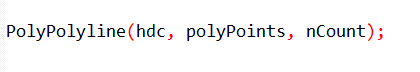
lpPoints: A pointer to an array of POINT structures, where each POINT structure specifies an x-coordinate and a y-coordinate for a vertex of the polyline.

cCount: The number of vertices in the polyline.

PolyPolyline Function



The PolyPolyline function draws multiple polylines. The syntax is as follows:



hdc: A handle to the DC that represents the drawing surface.

polyPoints: A pointer to an array of POLYLINE structures, where each POLYLINE structure specifies a polyline using its lpPoints member and its cCount member.

nCount: The number of polylines in the array.

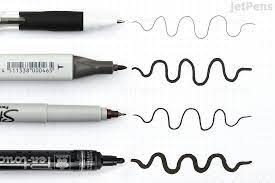
Factors Affecting Line Appearance

Five device context attributes influence the appearance of lines drawn using these functions:

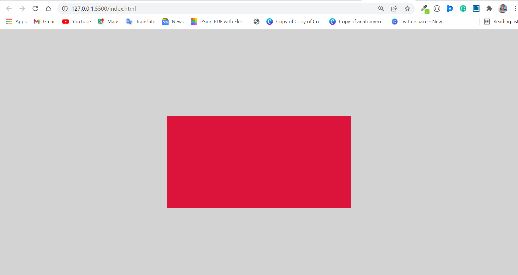
Current pen position: This attribute applies to LineTo, PolylineTo, PolyBezierTo, and ArcTo functions. It determines the starting point of the line.



Pen: The pen defines the style and attributes of the line, including its width, color, and pattern.



Background mode: This attribute determines how the background color is handled when drawing lines. OPAQUE mode fills the background with the specified color, while TRANSPARENT mode allows the underlying background to show through.



Background color: This attribute specifies the color used to fill the background in OPAQUE mode.



Drawing mode: This attribute determines how source pixels are combined with destination pixels when drawing lines. It affects how the line color is blended with the background color.