1. **GDI是什么的英文缩写？**

Graphics Device Interface

**2、 什么是设备的无关性？**

设备的无关性是指应用程序或操作系统能够不受硬件设备类型、品牌或型号的限制而运行。它是一种能够在不同的设备上运行的能力，而不需要针对每个设备编写独立的代码。在计算机图形学中，设备的无关性通常是指图形设备无关性，即应用程序能够使用相同的代码生成可在多种图形设备上渲染的图像。这样，开发人员就可以编写只需一次编译和调试，就能在不同的平台和设备上运行的程序。

**3、 GDI的主要任务是什么？起到什么样的作用？**

GDI负责系统与用户或绘图程序之间的信息交换，并控制在输出设备上显示图形或文字， 是Windows系统的重要组成部分。

开发人员只要建立与输出设备的关联，让系统加载相应的设备驱动程序即可。

**4、 什么是设备描述表？它的作用是什么？**

设备描述表是一种数据结构，用于描述硬件设备的属性和特性。在Windows系统中，设备描述表通常包含在驱动程序中，可以通过操作系统提供的API来访问。

设备描述表的作用是告诉操作系统如何与硬件设备进行交互。它包含了硬件设备的各种信息，例如设备类型、厂商ID、设备序列号、支持的功能和性能参数等。根据这些信息，操作系统可以选择适当的驱动程序来管理硬件设备，并向应用程序提供正确的接口。在Windows系统中，设备描述表通常使用一种名为“设备信息集”（Device Information Set）的结构来组织，每个设备都有一个唯一的标识符。应用程序可以使用基于设备描述表的API来遍历设备信息集，查找并枚举系统中可用的设备，并获取有关这些设备的详细信息。

**5、 图形刷新包括什么？**

1. 刷新请求
2. 对刷新请求的响应

(3) 有效的刷新方法

**6、 刷新请求有哪些？**

对象穿越

窗口移动

被覆盖后的恢复

**7、 应用程序什么情况下会发生刷新请求？同时发出什么消息？**

当用户区的内容需要刷新时，系统向应用程序消息队列发送WM\_PAINT消息，系统在应用程序的消息队列中加入该消息，以通知窗口函数执行刷新处理

**8、 什么是无效区域？**

窗口被另一个窗口覆盖的区域称为无效区域。

Windows系统为每个窗口建立了一个PAINTSTRUCT结构，该结构中包含了包围无效区域的一个最小矩形的结构RECT，应用程序可以根据这个无效矩形执行刷新操作。

**9、 有效的刷新方法有哪些？大家觉得哪种比较好？在代码中若主动要求刷新窗口，应该怎么办？**

记录事件。刷新时重新执行这个曾经发生的事件

保存副本。刷新时将副本拷贝到相应的窗口中

重新绘制。将图形绘制处理程序放在消息WM\_PAINT响应模块中，刷新时重绘图形

重新绘制比较好。将图形绘制处理程序放在消息WM\_PAINT响应模块中，刷新时重绘图形。

**10、 获取设备环境常用的两种方法是什么？**

调用函数BeginPaint或GetDC

**11、 调用BeginPaint函数和GetDc有什么联系与区别？在实际编程中怎样选择使用？**

(1) 调用BeginPaint函数

应用程序响应WM\_PAINT消息进行图形刷新时，主要通过调用BeginPaint函数获取设备环境

hdc=BeginPaint(hwnd,&ps)； //ps为PAINTSTRUCT类型结构

定义方式为：PAINTSTRUCT ps;

由BeginPaint函数获取的设备环境要用EndPaint函数释放

BOOL EndPaint(HWND hwnd, PAINTSTRUCT &ps)

(2) 调用GetDC函数

如果绘图工作并非由WM\_PAINT消息驱动，则调用GetDC函数获取设备环境。

hdc=GetDC(hwnd);

由GetDC函数获取的设备环境必须用ReleaseDC函数释放

void ReleaseDC(HWND hwnd);



**12、 什么是映像模式？为什么要用映像模式**

映像模式定义了将逻辑单位转化为设备的度量单位以及设备的x方向和y方向，程序员可在一个统一的逻辑坐标系中操作而不必考虑输出设备的坐标系情况

**13、 了解几种常用的映像模式，默认是哪种映像模式。**



默认是MM\_TEXT。

**14、 了解映像模式中函数的使用（不需要书面写），可以写部分重点函数的理解。**

设置设备环境的映像模式

SetMapMode(hdc,nMapMode); nMapMode为映像模式的整型标识符

获取当前设备环境的映像模式

nMapMode=GetMapMode(hdc);

窗口区域的定义由SetWindowExtEx函数完成，其函数原型形为：BOOL SetWindowExtEx

( HDC hdc,

int nHeight,nWidth， //以逻辑单位表示的窗口区域高宽度

LPSIZE lpSize， //函数调用前窗口区域尺寸的SIZE结构地址

);

视口区域的定义由SetViewportExtEx函数完成，函数原型为：

BOOL SetViewportExtEx

(

HDC hdc,

int nHeight,nWidth,//以物理设备单位表示的新视口区域高宽度

LPSIZE lpSize

);

**15、 画笔的操作有哪些？**

创建画笔

将画笔选入设备环境

删除画笔

**16、 了解画笔中使用的函数名字的英文意思，便于大家记忆函数，（不需要书面写出）**

**17、 了解常用的绘图函数，尤其是饼图函数，椭圆弧线函数。想一想怎么画圆，有没有画圆的函数？**

1．设置画笔当前位置的函数MoveToEx，

BOOL MoveToEx

(HDC hdc,

int X,Y, // X、Y分别为新位置的逻辑坐标

LPPOINT lpPoint //存放原画笔位置的POINT结构地址

)

2． 从当前位置向指定坐标点画直线的函数LineToEx，

BOOL LineToEx(HDC hdc,int X,int Y) //X和Y为线段的终点坐标

3． 从当前位置开始，依次用线段连接lpPoints中指定的各点

BOOL Polyline

( HDC hdc,

LPPOINT lpPoints, //指向包含各点坐标的POINT结构数组的指针

int nCount // nCount为POINT数组中点的个数

)

4． 绘制椭圆弧线的函数Arc，

BOOL Arc

(

HDC hdc,

int X1,intY1, //边框矩形左上角的逻辑坐标

int X2,int Y2, //边框矩形右下角的逻辑坐标

int X3,int Y3, //椭圆弧起始点坐标

int X4,int Y4 //椭圆弧终止点坐标

)

5． 绘制饼图，并用当前画刷进行填充

BOOL Pie

(

HDC hdc,

int X1,intY1, //边框矩形左上角的逻辑坐标

int X2,int Y2, //边框矩形右下角的逻辑坐标

int X3,int Y3, //椭圆弧起始经线的确定点坐标

int X4,int Y4 //椭圆弧终止经线的确定点坐标

)

6． 绘制矩形，并用当前画刷进行填充

BOOL Rectangle(HDC hdc,int X1,int Y1,int X2,int Y2)

7． 绘制圆角矩形，并用当前画刷填充

BOOL RoundRect (HDC hdc,int X1,int Y1,int X2,int Y2,

int nHeight, int nWidth)

8． 绘制椭圆，并用当前画刷填充

BOOL Ellipse(HDC hdc,intX1,intY1,intX2,intY2)

9．绘制多边形，并用当前画刷填充

BOOL Polygon(HDC hdc,LPPOINT lpPoints,int nCount)