**GDI+**

Eines der interessantesten Leistungsmerkmale von Windows Forms sind die Grafikfähigke­iten. Mit dem Graphical Device Interface (GDI+) können Sie alle Arten von Schnittstellen ­für das Zeichnen und Malen erstellen, von Bildeditoren bis zu Textmodellierungsanwendungen, oder sogar etwas so Simples wie Microsoft Paint.

Das Thema Grafikfähigkeiten in Windows umfasst eine breite Palette von Anwendungen. Fast alles, was man in Windows macht, betrifft einen grafischen Aspekt. Viele haben schon wenigstens einmal von Grafik- oder Videokarten in ihrem Computer gehört. Typ, Größe. Farbtiefe und andere Eigenschaften der Anzeige hängen alle von der verwendeten Grafik­karte ab. Das uralte Problem mit den Grafikfähigkeiten eines Betriebssystems besteht darin, dass man als Entwickler nie vorher weiß, welche Leistung die jeweilige Grafikkarte erbringt. In modernen PCs kann man ohne weiteres Komponenten austauschen, ohne das übrige System zu verändern, so dass möglicherweise keine zwei Computer ident sind. Die Verarbeitung von Grafiken wird dadurch etwas kompliziert, denn die Anwendungen müssen mit allen möglichen Grafikkarten zurechtkommen - leider erledigen nur wenige Grafikkarten ihre Arbeit auf ähnliche Weise.

Man braucht also eine einfache Schnittstelle, die der Entwickler nutzen kann, um Bildpro­gramme unabhängig von der verwendeten Hardware zu erstellen. Ursprünglich entwi­ckelte Microsoft das GDI (Graphical Device Interface, grafische Geräteschnittstelle), um dieses Problem zu lösen. GDI dient als eine Art Übersetzer für grafische Hardware. Man schickt Befehle ans GDI, um Grafikfunktionen auszuführen, und GDI wiederum spricht zur jeweiligen Hardware. Zum Glück ist GDI vielsprachig; es wurde entworfen, um mit praktisch jeder Art von Grafikkarte auf dem Markt zu kommunizieren. Daher müs­sen sich Entwickler nicht mit den Eigenheiten jeder einzelnen Grafikkarte befassen. Viel­mehr kann man sich nun auf den Umgang mit dem GDI konzentrieren, das sich wiederum mit den jeweiligen Grafikkarten auseinandersetzt.

GDI und GDI+ betreffen nur zweidimensionale Grafiken unter Windows. Für dreidimensionale Grafik sind DirectX und OpenGL zuständig, die die gleiche Ubersetzerfunktion wie GDI(+) haben.

Bei GDI+ handelt es sich um eine neu entwickelte, erweiterte Version von GDI. Diese unterstützt nicht nur die neuesten Grafiktechniken, sondern ist vollständig konform zu .NET. Gerätekontext und Handles braucht man nicht mehr, denn man hat nun ledig­lich mit Grafikklassen und ihren Mitgliedern zu tun, genau wie in jeder anderen Klasse in Windows Forms.

GDI+ lässt sich in drei Bereiche teilen.

1. Vektorgrafik: Der erste Bereich dreht sich um Grunde­lemente - Dinge wie Linien, Kurven und einfache Figuren. Vektorgrafik basiert auf einem Koordinatensystem; ändern sich die Koordinaten, so ändert sich auch die Grafik.
2. Bitmap Grafiken: Der zweite Bereich besteht aus (Bitmap-) Grafiken, dreht sich also um Dinge, die sich nicht mit einfachen Linien darstellen lassen (wie etwa die Hintergrundgrafik für Ihren Desktop). Für Grafiken ist ein Koordinatensystem ohne Bedeutung, denn Bilder werden ohne Rücksicht auf x- und y-Koordinaten gemalt. Weil Vektorgrafiken auf Koordinaten basieren, lassen sie sich einfacher und effizienter bearbeiten als Bilder. Wenn man etwa eine Linie verzerren will, ändert man einfach die entsprechenden Koordinaten. Will man eine Bitmap-Grafik verzerren, muss man die Bits bearbeiten, aus denen die Grafik besteht. Vektorgrafiken lassen sich beliebig ohne Qualitätsverlust verzerren, denn eine Koordinate ist eine Koordinate. Im Gegensatz dazu verliert eine Grafik beim Skalieren an Qualität, weil es nur eine bestimmte Datenmenge pro Bild gibt; man kann einem Bild keine neuen Details hinzufügen, ohne es neu zu erfassen.
3. Typografie: Die dritte Komponente besteht aus Typografie bzw. Text. Hier geht es um unterschiedli­che Schriftarten, Schriftgrößen, -stile und -farben.

Alle drei Bereiche drehen sich um ein paar zentrale GDI+-Klassen und eine Reihe von grundlegenden Konzepten.

**Pinsel und Stifte:**

In GDI+ stoßt man laufend auf zwei gängige Begriffe, nämlich Stifte *(pens)* und Pinsel *(brushes). S*ie ähneln ­ihren Gegenstücken aus der Realität.

Ein Pen ist im Wesentlichen für das Zeichnen einer Linie zuständig. Man muss sich einen echten Stift vorstellen. Da gibt es Stifte mit verschiedenen Farbtinten, verschiedenen Dicken (ein Füller zieht einen dickeren Strich als ein Kugelschreiber) und sogar verschiedenen Texturen. In GDI+ verhält sich ein Pen genauso. Man kann einen Pen mit allen gewünschten Charakteristika erzeugen und damit Linien oder Text zeichnen.

Ein Brush ist dafür zuständig, die von einem Pen gezeichneten Figuren mit einer vorgegebenen Farbe und Textur auszufüllen. So würde man etwa mit dem Pen einen Kreis zeich­nen, ihn aber mit einem Brush mit der Farbe Orange ausfüllen. Brushes lassen sich auf Dinge anwenden, die man nicht unbedingt als Figuren ansehen würde, beispielsweise Schriften. In GDI+ können Brushes sogar Farbverläufe malen, etwa um Objekte inter­essanter zu machen.

Das Graphics-Objekt bildet das Kernstück von GDI+. Es enthält alle Methoden und Eigen­schaften, die man zum Zeichnen von Linien und Text sowie zum Drucken benötigt.

**Rechtecke und Bereiche**

Zwei weitere wichtige Begriffe sind Rechtecke *(Rectangles)* und Bereiche *(Regions).* Ein Rectangle ist in NET einfach ein rechtwinkliges grafisches Objekt und eine Region stellt die Fläche dieses Rechtecks dar. Diese zwei Objekte werden oft benutzt, um bestimmte Bereiche zu vereinbaren oder festzulegen, in die andere Objekte (etwa Linien oder Bilder) gezeichnet werden sollen. Man sieht diese Objekte praktisch nie ohne Referenz auf eine *Graphics*-Methode.

Ein Rechteck lässt sich auf zweierlei Weise erstellen: durch das Angeben eines Startpunk­tes links oben und eines *Si*ze-Werts oder durch das Angeben der x- und y-Koordinaten für die obere linke Ecke sowie einer Breite und Höhe:

Rectangle objRect = new Rectangle(new Point(0,0), new Size(100,100));

Im zweiten Konstruktor bestehen die ersten beiden Parameter aus den x- und y-Koordina­ten des Startpunktes und die letzten beiden Koordinaten aus der Breite und Höhe des Rechtecks.

Rectangle objRect = new Rectangle(0,0,100,100);

Ein Region-Objekt wird erzeugt, indem man ihm ein Rectangle-Objekt übergibt:

Region objRegion = new Region(objRect);

Alternativ kann man eine beliebige Form mit einem GraphicsPath-Objekt angeben.

Diese beiden Objekte verfügen selbst über nicht besonders viele Methoden, doch sie erwei­sen sich als hilfreich bei der Bestimmung von Grafikumrissen, weshalb man sie stets im Zusammenhang mit anderen Grafikobjekten antrifft.

***Das Ereignis Paint:***

GDI+ ist dafür zuständig die Fenster jeder verwendeten Anwendung zu zeichnen. Jedes Mal wenn ein Fenster gezeichnet wird (etwa beim Start, beim Maximieren usw.), wird das Paint-Ereignis ausgelöst. Wann immer man steuert, wie ein Windows-Formular gezeichnet wird, muss man für Paint einen Ereignishandler bereitstellen.