Лекция 6.

**Идеология графического вывода**

Особенностью операционной системы Windows является графическое представление информации.

Естественно, что практически все приложения используют экран, чтобы отобразить данные, которыми они управляют. Даже если это не так, то все равно визуальные элементы приложений отображаются на экране самой операционной системой. Windows обеспечивает универсальность представления информации , как на экране, так и на других устройствах вывода, например на принтере. Причем, что чрезвычайно важно, используются для этого одни и те же примитивы отображения. Система сама распознает целевое устройство и активизирует соответствующий ему модуль. В связи с тем, что Windows - многозадачная система, к приложениям предъявляется ряд требований, исключающих конфликты при использовании функций вывода. Однако это не значит, что Windows обеспечивает приложения только набором функций вывода на экран или печать — система управляет всем выводом. Правильнее будет сказать, что в качестве первичного устройства вывода приложения используют скорее окно, чем непосредственно экран.

Каждое устройство вывода в Windows имеет набор текущих параметров, с использованием которых и происходит собственно вывод. Причем в каждый конкретный момент времени только одному приложению соответствует некоторое логическое устройство вывода, что исключает одновременный доступ к последнему - перед началом процесса вывода приложение изменяет параметры вывода, подстраивая их под себя.

**Аппаратно-независимый графический вывод**

Как вы уже поняли, одна из главных особенностей Windows независимость графического вывода от устройства. Программное обеспечение, которое поддерживает независимость, содержится в двух библиотеках динамической компоновки (DLL, Dynamic Link Library). Первая, GDI.DLL, обеспечивает графический интерфейс устройства (Graphics Device Interface), а вторая, зависящая от конкретного устройства, является драйвером этого устройства. Например, если приложение осуществляет вывод в область пользовательского окна на дисплее VGA, это библиотека VGA.DLL, а если на принтер Epson FX-80, то EPSON9.DLL.

GDI позволяет программировать графический вывод независимо от оборудования, на котором он отображается, а это означает, что ваша программа может работать на разных машинах с разными дисплейными устройствами. В дополнение к экранам дис­плеев Windows интерфейс GDI также поддерживает принтеры и плоттеры, так что вывод данных на принтер или плоттер включает, по сути, те же механизмы, что и отображение информации на экране.

Перед операцией вывода на некоторое устройство приложение должно запросить GDI о загрузке соответствующего драйвера (обычно это происходит автоматически и не требует от программиста дополнительных действий). Как только драйвер загружен, приложение может настроить ряд параметров вывода, таких как цвет линии и ее ширина, тип кисти и ее цвет, шрифт, область отсечения и т. д.

MFC избавляет программиста от необходимости непосредственной работы с дескрипторами контекстов устройств. Дескриптор контекста устройства и функции рисования GDI инкапсулированы в класс "Контекст устройства" – CDC. От него унаследованы классы для представления различных контекстов устройств.

**Контекст устройства**

Контекст устройства - структура, определяющая набор графических объектов и связанных с ними атрибутов и графических режимов, которые, собственно, и воздействуют на вывод. Графические объекты включают карандаши для рисования линий, кисти для закрашивания и заполнения, битовые образы для копирования или прокрутки частей экрана, цветовые палитры для определения набора доступных цветов, области для отсечения и других операций, а также контуры для операций рисования и закрашивания.

Настройка параметров осуществляется посредством вызова специальных функций.

Используя контекст устройства, приложение может осуществлять следующий набор операций:

* перечисление графических объектов;
* установка новых графических объектов;
* удаление графических объектов;
* сохранение графических объектов, их атрибутов и параметров графических режимов;
* восстановление графических объектов, их атрибутов и параметров графических режимов.

Кроме того, приложение может использовать контекст устройства для определения процесса графического вывода, прерывания длительных графических операций, начатых другим потоком многопоточного приложения, а также инициализировать принтер.

Контекст устройства- объект класса СDC , содержащий все необходимые методы для построения изображения в окне. Для осуществления вывода создается контекст устройства, а далее к созданному объекту можно применять все методы класса СDC

**Класс СDC**

Все рисование в программе выполняется с использованием методов класса СDC. Все объекты этого класса и его производных классов содержат контекст устройства и методы, необходимые для того, чтобы посылать графику и текст на дисплей и принтер. В нем есть также методы, предназначенные для полу­чения информации об используемом физическом устройстве вывода.

Поскольку объекты класса СDC могут представлять почти все, что вам, вероятно, понадобится для графического вывода, в классе определено множество методов- фактически, более ста. Поэтому здесь мы рассмотрим только некоторые из них, а остальные — позднее, по мере необходимости.

Обратите внимание, что MFC включает некоторые более специализированные классы для графического вывода, которые унаследованы от CDC.

Классы контекстов устройств:

CDC - Базовый класс для контекстов устройств; используется непосредственно для доступа ко всем экранным и не экранным контекстам устройств, таких как принтеры.

CWindowDC - Контекст экрана для всего окна, включая клиентскую и не клиентскую области.

CСlientDC - Контекст экрана для клиентской области окна; используется, например, при рисовании в ответ на события от мыши.

CPaintDC - Контекст экрана, используемый в функции OnPaint() оконных классов; в конструкторе автоматически вызывается функция BeginPaint, а в деструкторе – EndPaint.

CMetaFileDC - Контекст устройства для метафайла Windows, который содержит последовательность команд GDI, необходимых для создания образа.

Например, вы будете использовать объекты класса CClientDC, поскольку он унаследован от CDC и содержит все члены, которые мы будем обсуждать далее. Преимущество CClientDC перед CDC состоит в том, что он всегда содержит контекст устройства, представляющий только клиентскую область окна, а это как раз то, что необходимо в большинстве случаев.

Объекты этих классов можно создавать как автоматически, так и динамически. В конструкторе и деструкторе каждого класса вызываются функции GDI для получения и освобождения дескрипторов контекста устройства.

**Атрибуты контекста устройства**

В контексте устройства хранится ряд атрибутов, влияющих на работу функций рисования. В классе CDC есть функции-члены для чтения текущих значений и для изменения этих атрибутов.

Основные атрибуты контекста устройства

|  |  |  |  |
| --- | --- | --- | --- |
| **Атрибут** | **Значение по умолчанию** | **Метод CDC для задания значения** | **Метод СDC для получения значения** |
| Цвет текста | Черный | SetTextColor | GetTextColor |
| Цвет фона | Белый | SetBkColor | GetBkColor |
| Режим фона | OPAQUE | SetBkMode | GetBkMode |
| Режим преобразования координат | MM\_TEXT | SetMapMode | GetMapMode |
| Режим рисования | R2\_COPYPEN | SetROP2 | GetROP2 |
| Текущая позиция | (0,0) | MoveTo | GetCurrentPosition |
| Текущее перо | BLACK\_PEN | SelectObject | SelectObject |
| Текущая кисть | WHITE\_BRUSH | SelectObject | SelectObject |
| Текущий шрифт | SYSTEM\_FONT | SelectObject | SelectObject |

Различные функции рисования CDC пользуются атрибутами по-разному. Например, цвет, ширина и стиль (сплошная, штриховая и т.п.) линии для рисования отрезка функцией LineTo определяются текущим пером. При рисовании прямоугольника функцией Rectangle модуль GDI рисует контур текущим пером, а внутреннюю область заполняет текущей кистью. Цвет текста, фона и шрифт используются всеми функциями отображения текста. Фоновый цвет применяется также при заполнении промежутков в несплошных линиях. Если фоновый цвет не нужен, его можно отключить (сделать "прозрачным"):

dc.SetBkMode( TRANSPARENT );

Атрибуты CDC чаще всего изменяются с помощью функции SelectObject. Она предназначена для "выбора" в контексте устройства объектов GDI 6-ти типов:

**Графические объекты**

Система Windows обеспечивает приложения набором графических объектов, которые позволяют управлять выводом.

Для этого применяются:

* битовые массивы (bitmaps) — прямоугольные массивы точек, формирующие растровые изображения;
* карандаши (pens) — для задания параметров рисования линий, таких как толщина, цвет и стиль (сплошная, прерывистая и т. п.);
* кисти (brushes) — для задания параметров заливки замкнутых контуров, таких как цвет и стиль;
* регионы (regions) — области окна, которые могут быть ограничены прямоугольником, многоугольником, эллипсом или их комбинацией для выполнения операций заполнения, заливки, инверсии и т. п.; кроме того, служат для определения местоположения курсора;
* логические палитры (logical palettes) — для осуществления интерфейса между приложением и цветным устройством вывода, таким как дисплей; содержат список цветов, необходимых приложению;
* контуры (paths) — для целей заполнения или выделения контура различных фигур;
* шрифты (fonts) — для задания параметров вывода текста, включая имя шрифта, размер символов и т. п.

В MFC перья, кисти и шрифты представлены классами CPen, CBrush и CFont. Свойства пера "по умолчанию": сплошная черная линия толщиной 1 пиксель; кисть "по умолчанию": сплошная белая; шрифт "по умолчанию": пропорциональный высотой примерно 12 пт. Вы можете создавать объекты-перья, кисти и шрифты с нужными вам свойствами и выбирать их в любом контексте устройства.

**Клиентская область окна**

Окно не имеет фиксированного положения на экране и даже не имеет фиксированной видимой области, потому что может быть передвинуто с помощью мыши, и его размер может быть изменен перетаскиванием его границ. Как же вы узнаете, где именно на экране следует рисовать?

К счастью, вам этого знать и не нужно. Поскольку Windows предоставляет согла­сованный способ рисования в окне, вам не приходится беспокоиться о точном ме­сте на экране; если бы это было не так, рисование в окне чересчур бы усложнилось. Windows обеспечивает это, поддерживая координатную систему для клиентской обла­сти окна, локальную по отношению к окну. Она всегда использует левый верхний угол клиентской области в качестве точки отсчета. Все точки внутри клиентской области определены относительно этой начальной точки, как показано на рис. .



Горизонтальное и вертикальное расстояние точки от левого верхнего угла клиентской области всегда одно и то же, независимо от местоположения окна на экране и его текущего размера. Конечно, Windows необходимо отслеживать местоположение окна, и когда вы рисуете что-то в точке клиентской области, она должна определять, где именно эта точка находится на экране.

**Режимы отображения**

Каждый режим отображения в контексте устройства идентифицируется иден­тификатором (ID), подобно тому, как это делается с сообщениями Windows. Каждый символ имеет префикс ММ\_, указывающий на то, что речь идет о режиме отображе­ния (mapping mode). Windows поддерживает режимы отображения, перечисленныемв таблице.

Режимы отображения Windows

|  |  |  |
| --- | --- | --- |
| **Константа для обозначения режима** | **Расстояние, соответствующее логической единице** | **Ориентация координатных осей** |
| MM\_TEXT | 1 пиксел | x вправо, у вниз |
| MM\_LOMETRIC | 0.1 мм | x вправо, у вверх |
| MM\_HIMETRIC | 0.01 мм | x вправо, у вверх |
| MM\_LOENGLISH | 0.01 дюйма | x вправо, у вверх |
| MM\_HIENGLISH | 0.001 дюйма | x вправо, у вверх |
| MM\_TWIPS | 1/1440 дюйма (0.0007 дюйма) | x вправо, у вверх |
| MM\_ISOTROPIC | Определяется пользователем (масштаб по осям x и y одинаков) | Определяется пользователем |
| MM\_ANISOTROPIC | Определяется пользователем (масштаб по осям x и y задается независимо) | Определяется пользователем |

ММ\_ТЕХТ — режим отображения по умолчанию для контекста устройства. Если вам нужно использовать другой режим отображения, вы должны предпринять шаги по его замене. Обратите внимание, что направление положительной оси у в режиме ММ\_ТЕХТ противоположно тому, к которому вы привыкли в школьном курсе геоме­трии, что показано на рис. ниже.

По умолчанию точка в верхнем левом углу клиентской области имеет координаты (0, 0) во всех режимах отображения, хотя и возможно переместить начало координат из верхнего левого угла клиентской области, если вы этого хотите. Например, неко­торые приложения, представляющие данные в графическом виде, перемещают нача­ло координат в центр клиентской области, чтобы упростить рисование графиков по числовым данным.



При начале координат в левом верхнем углу в режиме ММ\_ТЕХТ точка, отстоящая на 50 пикселей от левой границы и на 100 пикселей вниз от верши­ны клиентской области, будет иметь координаты (50, 100). Конечно, поскольку еди­ницами измерения являются пиксели, эта точка окажется ближе к верхнему левому углу клиентской области, если ваш монитор имеет разрешение 1280x1024, чем при разрешении 1024x768. Обратите внимание, что установка DPI (dot per inch — точек на дюйм) вашего дисплея влияет на представление во всех режимах отображения.

Установка по умолчанию предполагает 96 DPI, поэтому если DPI для вашего дисплея установлено в другое значение, это повлияет на внешний вид. Координаты всегда измеряются 32-битными целыми числами. Максимальный физический размер всего рисунка варьируется с физической длиной единицы координат, что определяется режимом отображения.

Направления координатных осей х и у в режиме MM\_LOENGLISH и всех остальных режимах отображения одинаковы, но отличаются от ММ\_ТЕХТ. Координатные оси для MM\_LOENGLISH показаны на рис..

Хотя, положительные значения у согласуются с тем, что вы учили в школе (значение у увеличивается по мере движения по экрану снизу вверх), MM\_LOENGLISH - все же несколько необычный режим, поскольку точка начала координат находится в левом верхнем углу клиентской области, так что для точек внутри видимой клиентской области у всегда отрицательно.



В режиме отображения MM\_LOENGLISH единицы измерения вдоль осей составляют 0,01 дюйма, так что точка с координатами (50, -100) находится на полдюйма от левой границы и на один дюйм ниже вершины клиентской области. Объект имеет всегда один и тот же размер на экране, независимо от разрешения монитора, на котором он отображается. Если вы рисуете что-либо в режиме MM\_LOENGLISH с отрицатель­ной координатой х или положительной у, это оказывается за пределами клиентской области и потому невидимо, поскольку точка отсчета (0, 0) по умолчанию располо­жена в левом верхнем углу. Однако точку начала координат можно сдвинуть, вы­звав необходимый метод класса MFC CDC SetViewportOrg .