GDI

Базови типове DC:
- **Display DC**
- **Printer DC**
- **Memory DC**
- **Информационен DC**

MFC класова йерархия за поддръжка на DC в NT
- **CDC**
- **CPaintDC**
- **CMetaFileDC**
- **Windowed**

```
Void CMyView::OnLButtonDown(..)
{           CRect rect;
            CClientDC dc(this) //иска указател към прозорец
            dc.GetClipBox(rect);          // връща областта за рисуване
}                                         //dc авт. се унищожава
```

CPaintDC обикновено се използва ако предефинирате ф-ията OnPaint().

```
CPaintDC dc(this);
```

Класове на MFC за DC

За да изработите DC в приложението можете:

```
CDC* pDC = GetDC();          //всъщност CWnd::GetDC()
/* използване на DC за рисуване */ ReleaseDC(pDC);
```

същото може и с : **PAINTSTRUCT ps;**

CDC* pDC = BeginPaint(&ps); **…..**

извън кл. област трябва друг DC: **CWnd::GetWindowDC();**
За да получим инф. за възможностите на у-вото преди рисуване:
CDC::GetDeviceCaps(); **// напр. допустими цветове, пиксели…**
пр: **CClientDC dc(this);**
int cx = dc.GetDeviceCaps(HORZRES);

int cy = dc.GetDeviceCaps(VERTRES); **// получаваме напр. 1024 , 768**

Режим на чертане

CDC::SetROP2(); **// Set Raster Operation to**
напр. можете да установите изхода да инвертира цветовете на пикселите :
dc.SetROP2(R2_NOT); **dc.MoveTo(0,0);** **dc.lineTo(100, 200);**

Относно Mapping modes

Избраната mapping mode се помни в текущия DC и може да се прочете GetMapMode() – SetMapMode()).
Има следните ММ:
= ММ позоляващ мащабитане анизотропна, както и различна ориентация на осите.
= ММ, както горната , но с еднакъв мащаб по двете оси (изотропна).
= ММ лог. единица →във физическа 0.001 инч или 0.01 инч. Осите са : надясно и нагоре
= ММ, както горната но с 0.01 мм или 0.1 мм.
= ММ, лог. единица е = 1 пиксел (текстова). Осите са надясно и надолу.

= ММ лог. единица се преобразува в 1/20 от точката (1/1440 от инч). Осите са надясно и нагоре.

Превръщане на координати

Това става с CDC::LPtoDP() или CDC::DPtoLP();
за да получим в цифров вид централната точка на екрана:

```
CRect rect;          GetClientRect(&rect);
CPoint point( rect.width() / 2, rect.Height() / 2);
```

Но ако искате същата инф. в MM_LOENGLISH единици:

```
CRect rect;
GetClientRect(&rect);
CPoint point( rect.width()/ 2, rect.Height()/ 2);
CClientDC(this);   dc.SetMapMode(MM_LOENGLISH);   dc.DPtoLP(&point);
```

Местене началото на коорд.с-ма

```
CDC::SetWindowOrg()                  // за прозореца
CDC::SetViewportOrg()                // за видимата област
```

клиентски координати
екранни координати
преобразуваме:

```
CWnd::ClientToScreen();
CWnd::ScreenToClient();
```

GDI обекти

Pen **Font (подр. Solid black)**
Brush **Bitmap**
Palette **Region**

Всеки от тях може да се подмени с SelectObject():
CBitmap* pbmOld = dcMem.SelectObject(&bmpHello);
или : CBitmap* pbmOld = dcMem.SelectObject(&bmpHello);

Pen (писалки)

Има следните атрибути:
ширина; стил; цвят.
Cosmetic pen
Geometric pen
*** Patern**
*** шаблон за края на ред: заобленост, с правоъгълен ръб или полусглобка).**
*** шаблон за пресичане на 2 линии: просто, със заобляне или с отсичане.**

Цвят на писалката

Подава се чрез структурата COLORREF – 32 битова стойност от 4 полета (байт) за интензитета на червено, зелено, синьо и неизползваем.

COLORREF clrwhite = RGB(255,255,255);

COLORREF clrblack = RGB (0,0,0);

пр: CPen agreenpen;

agreenpen.Create(PS_SOLID, 1, RGB(0,255,0));

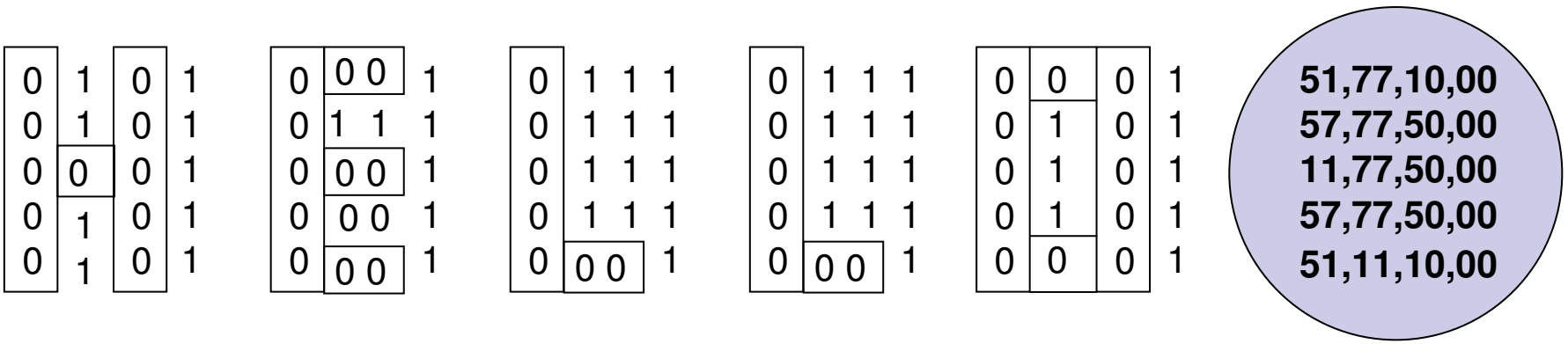
Растери и Bitmaps

hBitmap = LoadBitmap(hInstance, lpszBitmap);

hBitmap = CreateBitmap(…., указател масив изобр.);

тип BITMAP:

тип; ширина; височина; брой байтове на ред; брой цветови плоскости; брой битове/цвят; указател към масив с шаблона.



Brush

```
CBrush brRed ( RGB 255,0,0));
CBrush* pOldBrush = pDC→SelectObject( &brBlue);      // нов brush
CRect rcCircle(0,0,50,50);                           // подготвя структура
pDC→Ellipse( rcCircle);                              //  рисува кръг
pDC→SelectObject( pOldBrush);                        // възстановява старата
```

Шрифт

- font pitch – широчината на буквата. Им fixed pitch, variable pitch.
- Serif –завършека на буквата :Serif sans serif
- raster fonts
- vector fonts
- TrueType fonts

Фамилии шрифтове:

- decorative;
- modern;
- roman;
- script;
- swiss

Атрибути на шрифт

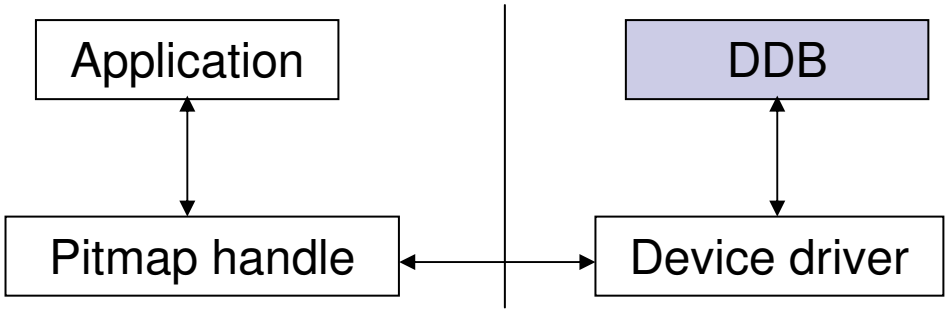
височина в лог. единици;
character set: ANSI_CHARSET;OEM_CHARSET; SYMBOL_CHARSET .
- output precision ; clipping region ; output quality
- фамилия :FF_DECORATIVE; FF_DONTCARE; FF_MODERN; FF_ROMAN; FF_SCRIPT; FF_SWISS.
плътност (font weight); escapment ; orientation;
Italic, underline, strikethrough ; Typeface

Избор на шрифт в програма

```
Void CDCTestView::OnDraw(CDC* pDC)
{           CRect rcClient;
GetClientRect( rcClient; pDC→DPtoLP( rcClient);
COLORREF clrold  = pDC→SetTextColer( RGB(0,255,0));
int oldMode = pDC→SetBkMode(TRANSPARENT);
CFont fntArial; fntArial.CreateFont(0,0,0,    ,     “Arial”);
CString szMsg  = “I’m writing with Arial”;
CFont* pOldFont = pDC→SelectObject( &fntArial);
pDC→TextOut( ……., szMsg);
pDC→SelectObject( pOldFont);   ...
```

Bitmaps

Device Independent Bitmap (DIB); Device Dependent Bitmap (DDB).



Ето пример за изобразяване на битмап с ресурсен ID – IDB_HELLO в view (в OnDraw()):

```
void  CBitmapView::OnDraw( CDC* pDC)
{  CBitmap bmp;
   Bmp.LoadBitmap( IDB_HELLO );
// определяне размера на битмап
   BITMAP bm;
   Bmp.GetObject( sizeof(BITMAP), &bm);
// създаваме memory DC, селектираме битмап в него и извеждаме към екран
   CDC dcMem;
   DcMem.CreateCompatibleDC(pDC);
CBitmap* pbmOld = dcMem.SelectObject( &bmp);
PDc→BitBlt(10,10,bm.bmWidth, bm.bmHeight, &dcMem, 0,0SRCCOPY);
```