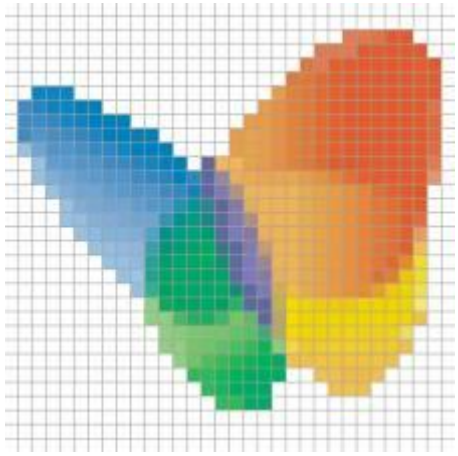


Jeden obraz jest wart tysiąca słów...

Tomasz Dzieniak

Toruń, 21 grudnia 2011 r.

Rodzaje grafiki

Według tradycyjnego podziału grafiki wyróżnia się dwa rodzaje:

- **grafikę rastrową**
- **grafikę wektorową**.

Różnice pomiędzy nimi polegają na

- **interpretacji obrazu**
- **zapisie obrazu** (zarówno w pamięci jak i na dysku).

Rodzaje grafiki - Grafika rastrowa

Tryb rastrowy jest naturalnym (i czasem jedynym) **trybem pracy większości urządzeń** graficznych, między innymi takich jak: monitor czy drukarka.

Obraz przedstawiany jest **w postaci dwuwymiarowej tablicy**. Każdy fragment obrazu określany jest przez pewną liczbę informacji - bitów. Stąd nazwa - **bitmapa**.

Najmniejszym elementem obrazka rastrowego jest **piksel** - najczęściej kwadratowy i zawsze wypełniony tylko jednolitym kolorem.

Jednym z podstawowych cech określających obraz jest jego **rozdzielczość**, czyli liczba pikseli w pionie i poziomie (wymiar wcześniej wspomnianej siatki). Wyższa rozdzielczość oznacza większą liczbę danych opisujących obraz i większy rozmiar pliku.

Rodzaje grafiki - Grafika rastrowa - Kolor piksela i głębia koloru

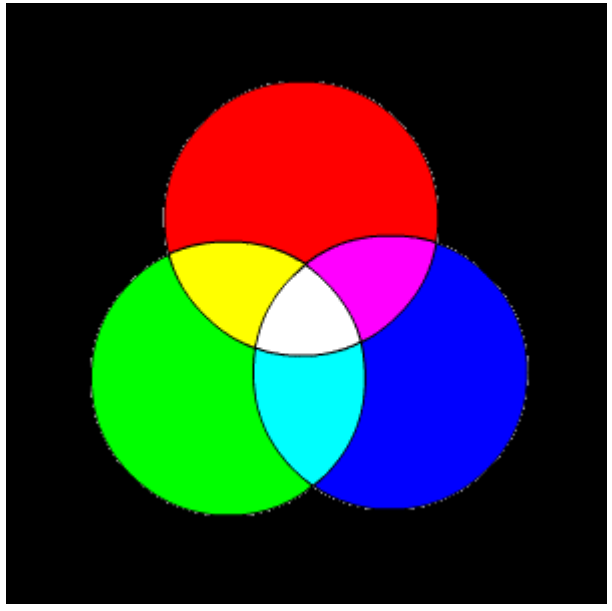
Na subiektywną jakość bitmapy rastrowej wpływa także **dokładność odwzorowania rzeczywistych kolorów**, ich **odcieni** i **natężenia**.

Większa jakość odwzorowania jest wprost proporcjonalna do wielkości używanej bitmapy.

Najpopularniejsze modele kodowania barw to **RGB** i **CMYK**.

Rodzaje grafiki - Grafika rastrowa - Kolor piksela i głębia koloru - RGB

Akronim **RGB** pochodzi od nazw kolorów podstawowych w systemie: **czerwonego**, **zielonego** i **niebieskiego**. Pozostałe barwy powstają poprzez mieszanie powyższych kolorów.

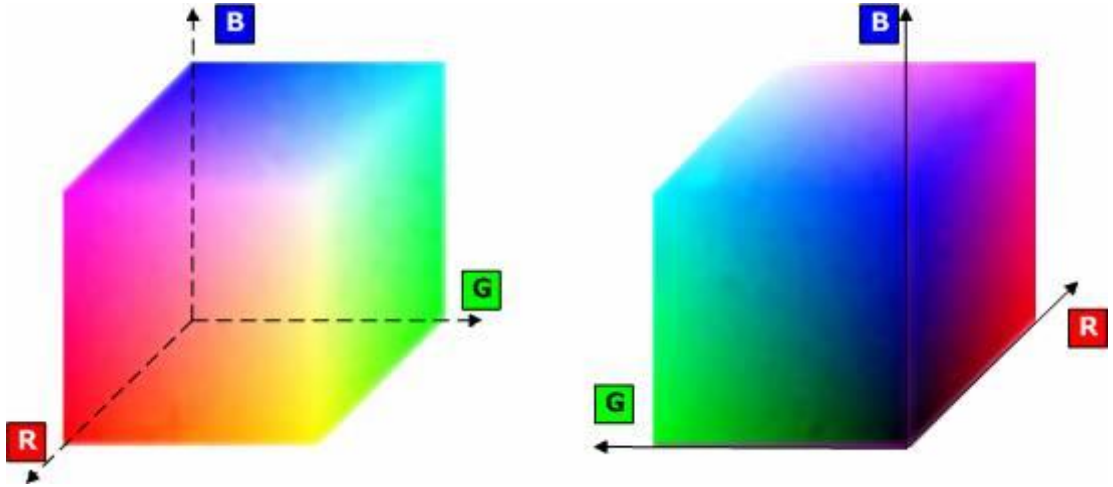


Składowe barw podstawowych w kolorze wynikowym nazywa się **kanałami** - w systemie RGB są więc trzy kanały.

Liczba wyświetlanych kolorów zależy od zastosowanej głębi kolorów:

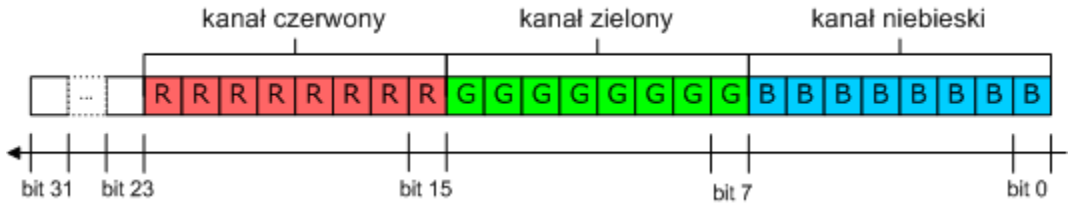
8 - bitowe = 2^8 kolorów, 16 - bitowe = 2^{16} kolorów itd.

Rodzaje grafiki - Grafika rastrowa - Kolor piksela i głębia koloru - RGB

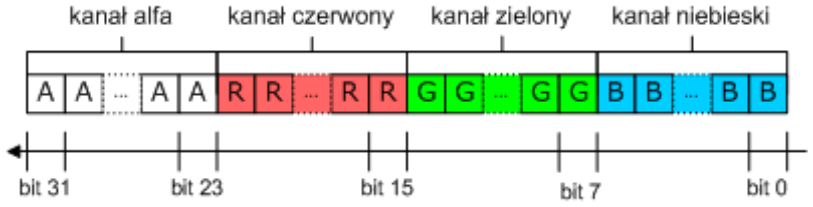


Przestrzeń barw RGB. Osie współrzędnych odpowiadają wartościom kolorów w poszczególnych kanałach.

Rodzaje grafiki - Grafika rastrowa - Kolor piksela i głębia koloru - RGB



Reprezentacja w pamięci komputera - COLORREF (w rzeczywistości - DWORD)

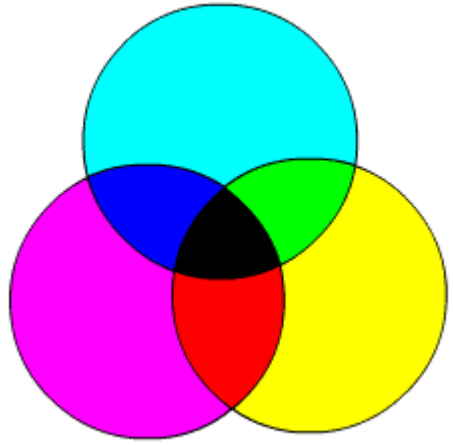


Rodzaje grafiki - Grafika rastrowa - Kolor piksela i głębia koloru - CMY(K)

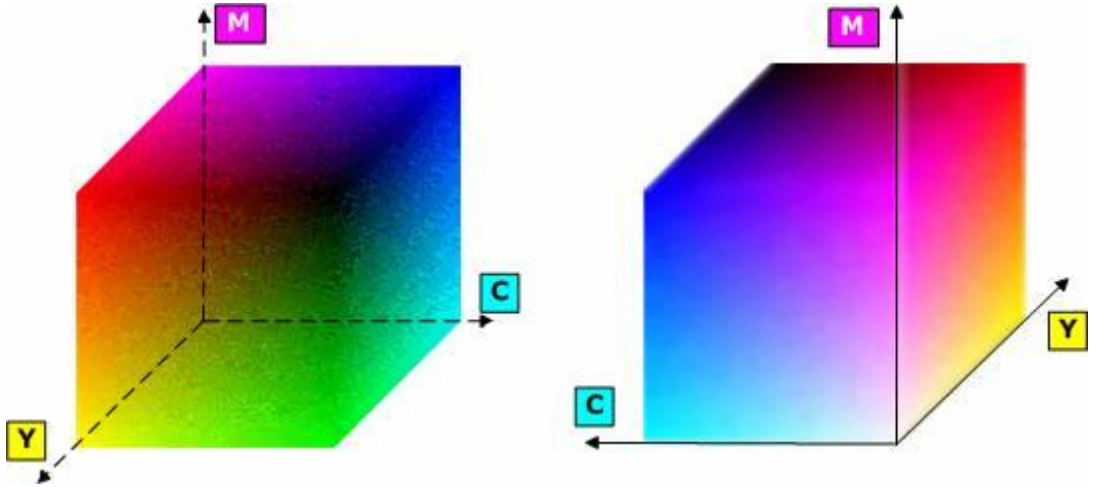
Akronim ponownie pochodzi od barw podstawowych - **morskiego**, **karmazynowego** i **żółtego** (i czarnego).

Stworzony głównie z myślą o drukarkach i innych urządzeniach, które swoją twórczość obrazują na papierze.

Łączenie barw opiera się nie na łączeniu fal świetlnych (RGB) a na mieszaniu barwników atramentu.



Rodzaje grafiki - Grafika rastrowa - Kolor piksela i głębia koloru - CMY(K)



Przestrzeń barw CMY(K). Osie współrzędnych odpowiadają wartościom kolorów w poszczególnych kanałach.

Rodzaje grafiki - Grafika wektorowa

W zamian punktowego opisu obrazu (tablicy dwuwymiarowej) stosowany jest **opis geometryczny**.

Ogromną zaletą takiego traktowania opisu obrazu jest **możliwość dowolnego skalowania**.

Najczęstsze zastosowanie:

- **szkice techniczne**
- **modelowanie graficzne**

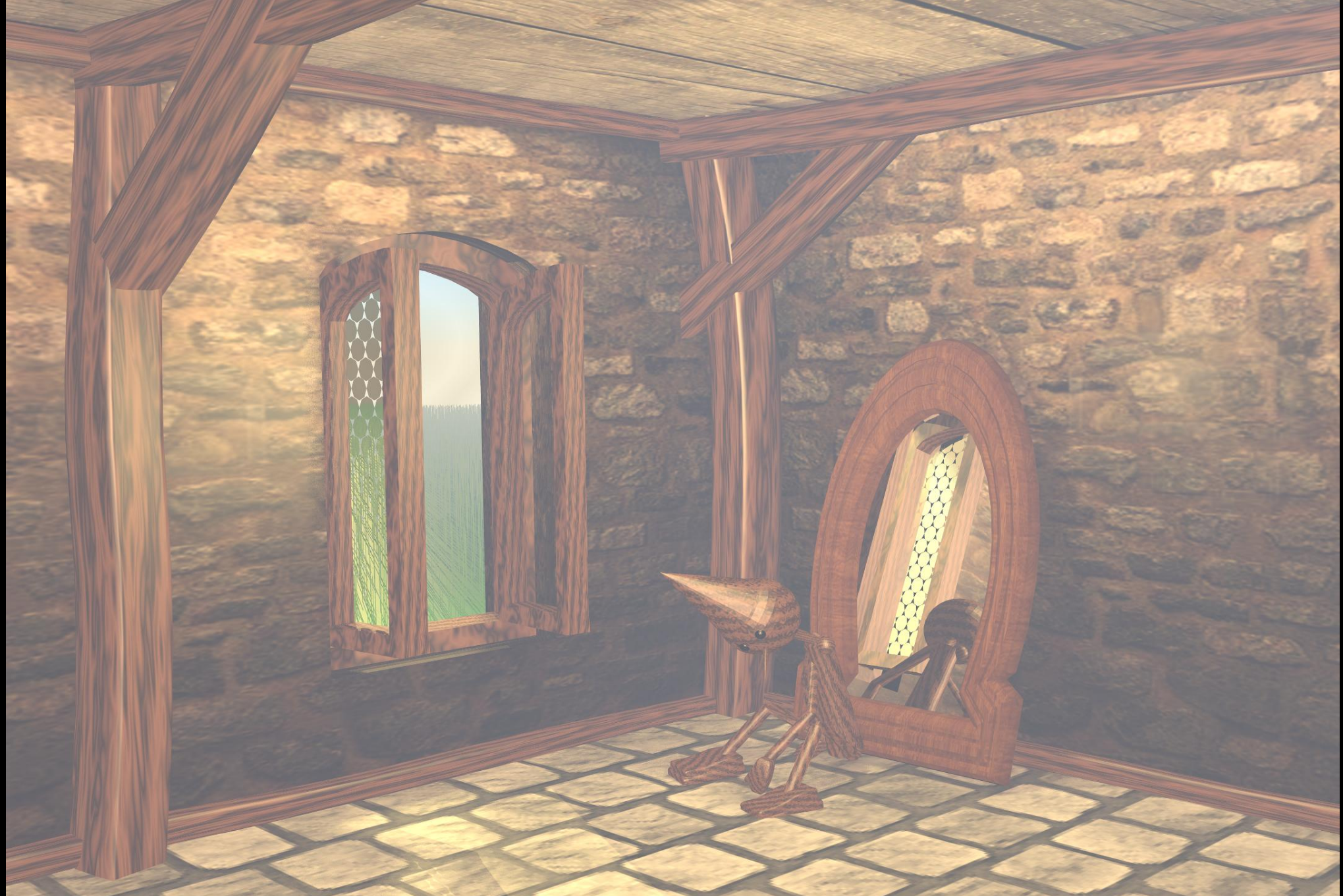
Najpopularniejszym narzędziem do tworzenia rysunków wektorowych - **programy CAD**.

Rodzaje grafiki - Grafika trójwymiarowa

Jest typem grafiki wektorowej. Stanowi jedyny, sensowny sposób kreowania świata przestrzennego.

Polega na tworzeniu trójwymiarowych modeli, na które składają się szeregi trójkątów, quadów bądź innych użytecznych prymitywów.

Przykład wymodelowanej przeze mnie sceny na potrzeby zajęć z fizyki w symulacji komputerowej i modelowaniu komputerowym znajduje się na następnym slajdzie.



Krótko o parametrach obrazu - jeszcze raz!

Główne parametry obrazu to

- **rozdzielczość**
- **głębia kolorów**
- **częstotliwość odświeżania**

Rozdzielczość - wartość określająca liczbę pikseli w pionie i poziomie, jaką zdolna jest wyświetlić karta graficzna i domyślne urządzenie wyświetlające (projektor, monitor, ekran).

przekątna	rozdzielczość
14 cali	640 x 480
15 cali	800 x 600
17 cali	1024 x 768
19 cali	1290 x 960
21 cali	1600 x 1200

Najpopularniejsze zestawienia rozdzielczości dla monitorów o proporcji 4:3

Głębia kolorów - określa wierność odwzorowania barw, czyli w najprostszym ujęciu - liczbę dostępnych kolorów.

<table>
<tr><th>ilość kolorów</th><th>liczba bitów</th><th>nazwa trybu</th><th>uwagi</th></tr>
<tr><td>2</td><td>1</td><td>monochromatyczny</td><td>czarno - biały</td></tr>
<tr><td>16</td><td>4</td><td>16 kolorów</td><td rowspan="2">oparty na palecie stałych barw</td></tr>
<tr><td>256</td><td>8</td><td>256 kolorów</td></tr>
<tr><td>65 536</td><td>16</td><td>High Color</td><td rowspan="2">kolor zapisany z użyciem RGB</td></tr>
<tr><td>16 777 216</td><td>24</td><td rowspan="2">True Color</td></tr>
<tr><td>16 777 216</td><td>32</td><td>j / w z wyrównaniem do czterech bajtów</td></tr>
</table>

Tryby głębi kolorów obsługiwane przez większość współczesnych monitorów

Częstotliwość odświeżania - określa zdolność **przerysowywania całego ekranu** w określonej jednostce czasu.

Zaleca się stosowanie częstotliwości odświeżania od **60 Hz wzwyż** (ogranicza to do minimum zmęczenie wzroku).

W celu uproszczenia realizacji odrysowywania obrazu na ekranie stosuje się:

- **synchronizację pionową**
- **buforowanie**
- **przełączanie stron**

Najczęstsze problemy:

- **migotanie**
- **szarpanie**

Podstawy Windows GDI

Najwyższa pora przejść do głównego punktu referatu - **Windows GDI**! Najpierw jednak należy poznać kilka kluczowych zagadnień, których znajomość znacząco uprości posługiwanie się udostępnionym API!

Zacznijmy od potoku graficznego!

Podstawy Windows GDI - Potok graficzny

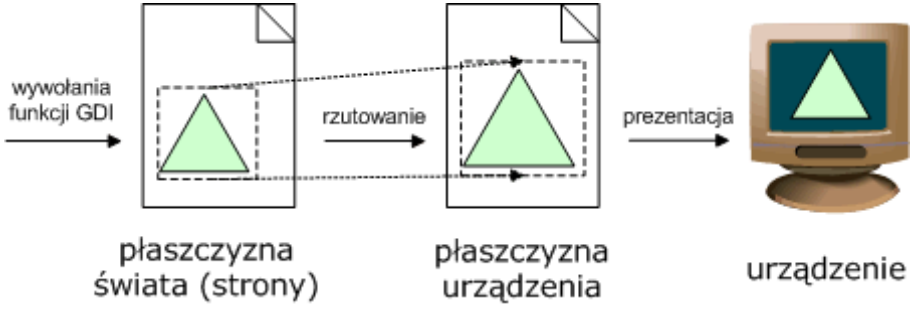
W myśl zasady "dziel i rządź" - potok graficzny stanowi kaskadę kolejnych szczebli, jakie pokonuje obraz - poprzez funkcje rysujące, przekształcenia aby ostatecznie pojawić się na ekranie.

Jest pewnego rodzaju abstrakcją, która umożliwia niezależność biblioteki od sprzętu.

Tryby grafiki - tryb kompatybilny

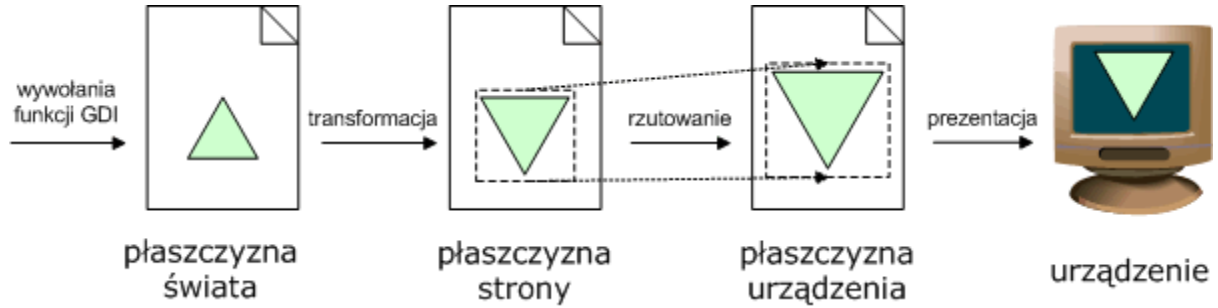
Stanowi domyślny tryb grafiki.

Nazwa związana jest z faktem, iż jest zgodny z 16 - bitowymi wersjami systemu Windows.



Ilustracja 16 - bitowego trybu kompatybilnego

Tryby grafiki - tryb zaawansowany



Ilustracja trybu zaawansowanego

Płaszczyzna świata

- Nie jest nieskończona (nie można rysować poza "kartką")
- Trzeba zmapować układ współrzędnych (określić sposób orientacji w przestrzeni)

Mapowanie układu współrzędnych:

- Precyzuje orientację płaszczyzny (kierunek osi pionowej)
- Określa jednostki logiczne

stała	tryb mapowania	rozmiar JL	zwrot osi	uwagi
MM_HIMETRIC	metryczny gęsty	0,01 mm	x a y a	przetwarzanie
MM_LOMETRIC	metryczny luźny	0,1 mm		
MM_HIENGLISH	angielski gęsty	0,001 cala		drukowanie
MM_LOENGLISH	angielski luźny	0,01 cala		
MM_TWIPS	twips	1/20 PD		
MM_ANISOTROPIC	anizotropowy	ustalany	dowolny	dowolne ustawienie parametrów
MM_ISOTROPIC	izotropowy			dba o proporcje
MM_TEXT	piksele urządzenia	Jeden piksel urządzenia	x a y a	domyślny

Tryby mapowania układu współrzędnych w Windows GDI

Kadr

Jeżeli korzysta się z mapowania izotropowego lub anizotropowego należy ustalić jednostkę oraz zwrot układu współrzędnych.

Pozycję kadru reguluje się za pomocą metody:

BOOL SetWindowOrgEx(HDC hdc, int X, int Y, LPSIZE lpPoint);

Rozciągłość osi ustawia się za pomocą metody:

BOOL SetWindowExtEx(HDC hdc, int nXExtent, int nYExtent, LPSIZE lpSize);

Płaszczyzna urządzenia

- Jest przedostatnim etapem potoku graficznego.
- Na tę płaszczyznę rzutowana jest płaszczyzna strony (świata).
- Jednostki logiczne są zastępowane jednostkami urządzenia

Wziernik - określa miejsce na płaszczyźnie urządzenia, gdzie pojawi się wygenerowany obraz.

Metoda odpowiedzialna za przesuwanie wziernika to:

BOOL SetViewportOrgEx(HDC hdc, int X, int Y, LPPOINT lpPoint);

Kontekst urządzenia

Jest to struktura przechowująca informacje na temat urządzenia graficznego. Na rzecz kontekstu można wywoływać funkcje GDI, tworząc w ten sposób obrazy wyświetlane na tym urządzeniu.

Kontekst może być związany z konkretnym urządzeniem:

- monitorem
- drukarką
- ploterem
- planszą rzutnika
- itp.

Kontekst urządzenia - pobieranie uchwytu

Są trzy metody pozyskiwania kontekstu urządzenia:

- od okna
  - podczas odrysowywania (WM_PAINT)

```
PAINSTRUCT ps;
HDC hdcKontekst = BeginPaint(hWnd, &ps);
```

  - z obszaru klienta okna

```
HDC hdcObszarKlienta = GetDC(hWnd);
// ReleaseDC(hWnd, hdcObszarKlienta);
```

  - całe okno

```
HDC hdcOkno = GetWindowDC(hWnd);
```

Kontekst urządzenia - pobieranie uchwytu

Są trzy metody pozyskiwania kontekstu urządzenia:

- od urządzenia
  - ekran - umożliwia pobranie kontekstu ekranu (tj.: wszystkiego co widać na płaszczyźnie ekranu; **pozwala rysować nawet na innych aplikacjach!**)

    HDC hdcEkran = GetDC(**NULL**);

    Pora na pierwszy program!

  - w ogólności można utworzyć kontekst dowolnego urządzenia!

- kontekst pamięciowy - to swego rodzaju bufor
  - Tworzy się go na podstawie istniejącego kontekstu urządzenia!
  - Aby zacząć z niego korzystać należy utworzyć bitmapę.

```
HBITMAP hbmpBitmapa = CreateCompatibleBitmap(hdcKontekst, nSzerokosc, nWysokosc);
```

  - Następnie wiążemy stworzoną bitmapę z kontekstem pamięciowym:

```
HBITMAP hbmpStaraBitmapa = (HBITMAP) SelectObject(hdcKontekstPamieciowy, hbmpBitmapa);
```

Atrybuty kontekstu - tryb mapowania

Określa wielkość jednostek logicznych (jednostek na płaszczyźnie świata / strony) dla danego kontekstu urządzenia.

Domyślne ustawienie -	MM_TEXT
Funkcja ustawiająca atrybut -	SetMapMode()
Funkcja pobierająca atrybut -	GetMapMode()

Atrybuty kontekstu - kadr

Jest to specjalny prostokąt na płaszczyźnie świata, który podczas przekształcenia na płaszczyznę urządzenia jest rzutowany do odpowiadającego mu wziernika.

Domyślne ustawienie -	(0,0)	(1,1)
Funkcja ustawiająca atrybut -	SetWindowOrgEx()	SetWindowExtEx()
Funkcja pobierająca atrybut -	GetWindowOrgEx()	GetWindowExtEx()

Atrybuty kontekstu - wziernik

Prostokąt położony na płaszczyźnie urządzenia. Określa, gdzie pojawi się wygenerowany obraz.

Domyślne ustawienie -	(0,0)	(1,1)
Funkcja ustawiająca atrybut -	SetViewportOrgEx()	SetViewportExtEx()
Funkcja pobierająca atrybut -	GetViewportOrgEx()	GetViewportExtEx()

Atrybuty pióra - obiekt pióra

Mogą istnieć niezależnie od urządzenia. Jeden kontekst = jedno pióro!

Domyślne ustawienie:	BLACK_PEN
Funkcja ustawiająca atrybut:	SelectObject()
Funkcja pobierająca atrybut:	GetCurrentObject()

Atrybuty pióra - aktualna pozycja

Jest czymś w rodzaju kursora graficznego. Wiele funkcji rozpoczyna od tego miejsca.

Domyślne ustawienie:	(0, 0)
Funkcje ustawiające atrybut:	MoveToEx()
	LineTo()
	PolylineTo()
Funkcja pobierająca atrybut:	GetCurrentPositionEx()

Atrybuty pióra - tryb rysowania

Określa w jaki sposób GDI ma reagować na przykrywanie już istniejących pikseli. Standardowo - piksele są zastępowane.

Domyślne ustawienie:	R2_COPYPEN
Funkcja ustawiająca atrybut:	SetROP2()
Funkcja pobierająca atrybut:	GetROP2()

Atrybuty pędzla - obiekt pędzla

Jeden kontekst = jeden pędzel! Domyślnie pędzel wypełniający jest biały.

Domyślne ustawienie:	WHITE_BRUSH
Funkcja ustawiająca atrybut:	SelectObject()
Funkcja pobierająca atrybut:	GetCurrentObject()

Atrybuty pędzla - punkt odniesienia pędzla

Stosowany tylko do pędzli, które malują regiony sąsiadującymi kopiami bitmap / deseniami.

Domyślne ustawienie:	(0, 0)
Funkcja ustawiająca atrybut:	SetBrushOrgEx()
Funkcja pobierająca atrybut:	GetBrushOrgEx()

Atrybuty bitmap - bitmapa kontekstu urządzenia

Każdy kontekst zawsze ma swoją bitmapę. Funkcje rysujące zmieniają piksele właśnie tej bitmapy.

Domyślne ustawienie:	Monochromatyczna bitmapa 1´1 lub bitmapa o parametrach zależnych od macierzystego urządzenia kontekstu
Funkcja ustawiająca atrybut:	SelectObject()
Funkcja pobierająca atrybut:	GetCurrentObject()

Atrybuty bitmap - tryb rozciągania

Właściwość ta definiuje sposób rozciągania bitmap przy kopiowaniu ich za pomocą funkcji StretchBit().

Domyślne ustawienie:	BLACKONWHITE
Funkcja ustawiająca atrybut:	SetStretchBltMode()
Funkcja pobierająca atrybut:	GetStretchBltMode()

Atrybuty tekstu - kolor tekstu

Kolor tekstu jest niezależny od czcionki i jej stylu.

Domyślne ustawienie:	czarny (RGB(0, 0, 0))
Funkcja ustawiająca atrybut:	SetTextColor()
Funkcja pobierająca atrybut:	GetTextColor()

Atrybuty tekstu - kolor tła

Parametr określa kolor tła najmniejszego prostokąta okalającego tekst. Jeżeli nie jest pożądany kolor tła należy zmienić tryb graficzny!

Domyślne ustawienie:	biały (RGB(255, 255, 255))
Funkcja ustawiająca atrybut:	SetBkColor()
Funkcja pobierająca atrybut:	GetBkColor()

Atrybuty tekstu - tryb tła

Precyzuje, czy tło tekstu ma być rysowane (OPAQUE) czy też nie (TRANSPARENT).

Domyślne ustawienie:	OPAQUE
Funkcja ustawiająca atrybut:	SetBkMode()
Funkcja pobierająca atrybut:	GetBkMode()

Atrybuty tekstu - czcionka

Określa parametry wyświetlanego tekstu (tj. czcionkę, styl, dekorację). Np.: Verdana, 10 punktów, pogrubiona i kursywa.

Domyślne ustawienie:	SYSTEM_FONT
Funkcja ustawiająca atrybut:	SelectObject()
Funkcja pobierająca atrybut:	GetCurrentObject()

Atrybuty tekstu - odstępy pomiędzy znakami

Podawany w jednostkach logicznych odstęp pomiędzy pojedynczymi znakami wypisywanego tekstu

Domyślne ustawienie:	0
Funkcja ustawiająca atrybut:	SetTextCharacterExtra()
Funkcja pobierająca atrybut:	GetTextCharacterExtra()

Zapisywanie i odtwarzanie atrybutów

Często konieczne jest operowanie na wielu kontekstach - stąd potrzeba ich zapisywania i odtwarzania.

```
int SaveDC(HDC hdc);

BOOL RestoreDC(HDC hdc, int nSavedDC);
```

Pora na kolejny program!

C. D. N.