

No to zaczynamy :-). Zapnijcie mocno pasy, przytrzymajcie się mocno foteli, bo oto wkraczamy we wspaniały świat grafiki komputerowej (i nie tylko). Nie będziemy jednak sterzeć przed monitorem do 4 nad ranem, ochlapując ściany litrami krwi w Quake'u, czy wyciskać ostatnie tchnienia z silnika naszego czerwoniutkiego Porshe. Dowiemy się jak to robią inni, czyli poznamy pierwsze tajniki biblioteki DirectX. Ufam, że to co chcę Wam przekazać, zaowocuje już w niedalekiej przyszłości wspaniałymi dziełami na miarę Quake'a czy Unreala, a może powstaną nowe, zupełnie odlotowe produkcje. Ale żeby niepotrzebnie nie przedłużać, dowiedzmy się na początek, jak to wszystko działa.

Co to jest...

No właśnie... czym jest tak w zasadzie pakiet DirectX? Zależy z której strony na to patrzeć. Dla graczy - wiadomo, instalujemy najnowszą wersję, restartujemy kompa, odpalamy i dajemy czadu w zachwycających wirtualnych światach stworzonych tylko i wyłącznie dla naszej rozrywki. My jednak popatrzymy na to z nieco innej perspektywy - my będziemy programistami. Pamiętam mój pierwszy kontakt z DirectX, było to już ho, ho dawno temu. Widok pierwszego przykładowego kodu przyprawił mnie o mały ból głowy i przez pewien czas poprzestałem jedynie na bezmózgowej rozrywce. Kiedyś jednak ciekawość zwyciężyła i zaczęło się. Dziesiątki godzin ślęczenia nad przykładami i dokumentacją nie poszły jednak na marne. W końcu zrozumiałem i wiem, że to nic straszego, a jak sami będziecie mieli się okazję przekonać, daje niesamowitą satysfakcję :-). Czym jest DirectX dla nas, przyszłych twórców Quake 4? Jest to nic innego jak... API (ang. *Application Programming Interface*). Ktoś zapyta, zaraz, zaraz, ale przecież API to Windows i w ogóle... A jednak... API Windows to jedno a API DirectX to drugie. Jak wspominałem kiedyś, API to zestaw funkcji i nie ważne czego będzie się dotyczył. Taki zestaw może służyć do pisania aplikacji dla systemu operacyjnego (API Windows, czyli WinAPI32), do tworzenia gier i aplikacji multimedialnych (API DirectX), czy jeszcze do innych zastosowań. Zestawów funkcji API może być wiele i możemy tworzyć sobie własne, tylko komu będzie się chciało? Przecież mamy gotowe. Nas interesuje jednak w tym momencie tylko jedno - jak to nasze kochane API wykorzystać. W pakiecie DirectX-a znajdziemy wiele funkcji, które pomogą nam tworzyć dwu- i trójwymiarową grafikę, generować efekty dźwiękowe, odgrywać muzykę, obsługiwać urządzenia wejściowe oraz tworzyć aplikacje pracujące w sieci, co, jak wiemy, jest najfajniejszą zabawą :-). W naszych rozważaniach zajmiemy się ostatnią wersją pakietu DirectX-a czyli numerem 8. Dotychczas mieliśmy okazję widzieć i używać wersji: 3.0, 5.0, 6.0, 6.1, 7.0 no i tą ostatnią. Pierwsze dwie wersje, pomimo swoich niebagatelnych, nawet jak na owe czasy, możliwości, nie wzbudziły zachwytu wśród programistów. Niezłe ciągi dostały się DirectX-owi, a zwłaszcza modułom grafiki, od naczelnego wodza twórców Quake'a - Johna Carmacka, który stwierdził, że Direct3D nie ma przyszłości, ale pomimo całego swojego geniuszu, tym razem jednak się pomylił. DirectX nadal ma się znakomicie, a sądząc po bogactwie różnorodnych kart na rynku, jeszcze długo będzie królować na platformie Windows. Wielu krytykuje DirectX-a za jego nieprzenośność na inne systemy, no ale powiedzcie sami, kto używa komputerów CRAY do grania w gry? Tę wadę DirectX rekompensuje natomiast niezłe dobranym zestawem funkcji, dzięki którym można tworzyć aplikacje w pełni wykorzystujące możliwości sprzętu, jakim dysponujemy, na poziomie dostępu do niego dotychczas niespotykanym. To prawie jak pisanie w assemblerze :-). W zasadzie DirectX to pakiet bibliotek, które zawierają zestaw obiektów COM, które z kolei zawierają wiele pożytecznych interfejsów i metod, pozwalających na dobranie się do odpowiednio zbudowanego sprzętu, który może realizować "na bieżąco" nasze zachcianki w postaci funkcji API. Możemy osiągnąć naprawdę niezłe efekty, których uzyskanie w czasie rzeczywistym, na normalnym procesorze, jest po prostu niemożliwe (mowa o grafice i muzyce oczywiście).

Podział

Co znajdziemy w pakiecie DirectX? Ci, którzy nie tracą czasu, wiedzą bardzo dobrze, natomiast leniwym jak zwykle trzeba wszystko wyjaśnić. Pakiet zawiera kilka modułów do obsługi poszczególnych urządzeń, które przy połączeniu w pewną całość, w pełni pozwolą rozkoszować się nam pięknym ciałem Larry Croft, która już n... ups... wracajmy może na drogę cnoty :-). Cóż to więc za moduły? W wersji 8 MS postanowił zrobić wszystkim programistom "miłą" niespodziankę i trochę zmienił swoją dotychczasową filozofię budowy pakietu. Ogólnie rzecz biorąc, mamy cztery moduły, czyli: obsługa grafiki, muzyki, urządzeń wejścia i sieci. Dawniej obsługą grafiki zajmowały się dwa moduły DirectDraw i Direct3D. Wszystko było piękne i proste do czasu, kiedy programiści MS postanowili to "usprawnić" i połączyli w wersji 8 grafikę 2D i 3D w jeden moduł zwany DirectX Graphics. Dźwięk i muzyka - to też było proste: DirectMusic i DirectSound. Dzisiaj musimy zadowolnić się jednym wielkim kombajnem, czyli DirectX Audio. Urządzenia wejścia, czyli jak łatwo się domyśleć myszki i joysticki, to stary poczciwy DirectInput. Za obsługę sieci odpowiedzialny jest natomiast DirectPlay. Mamy w ręce jeszcze jeden milutki pakiecik, który może nam posłużyć do równie fascynujących celów, jakimi są: odtwarzanie grafiki i muzyki oraz ich nagrywanie (MP3 i te rzeczy, sami wiecie :-). W następnych punktach omówimy sobie co zawiera moduł DirectX Graphics, jego podział i filozofię działania. Ponieważ nie jestem za dobry ani w muzyce, ani w całej reszcie, więc z następnymi pewnie sporo poczekamy, no chyba, że znajdą się faszynaci, którzy pomogą w tworzeniu tej strony, bo Ci, którzy próbowali, to wiedzą, że nie jest to łatwe. Przy okazji omawiania modułu grafiki dowiemy się też, jak udziela się w tym wszystkim technologia COM i czemu ona tu właściwie służy. W kilku następnych powiemy także o obiektach obecnych w module i za co one odpowiadają. Nie będziemy analizowali poszczególnych funkcji, bo to za dużo czasu by nam zajęło, a dokumentacja do nich jest przebogata. Ale może po kolei...

DirectX Graphics

Jak wspominałem już wiele razy, cały pakiet, tak więc i moduł DirectX Graphics, to obiekty COM. Ktoś zada pytanie dlaczego? Jeśli czytaliście dokładnie lekcję o COM-ie, to wiecie o sposobach dziedziczenia obiektów COM. Dziedziczymy

interfejsy i moduły wykonywalne, a nie kod. To powoduje, że każdy nowy obiekt zawiera wszystkie poprzednie. DirectX Graphics nie jest tu wyjątkiem. Mając do dyspozycji obiekty z serii 8, możemy bez problemu korzystać z interfejsów wersji 7, 6, 5 a nawet 3. Może nie jest to wielka zaleta, no bo po co się pechać w starocie, ale czasem jak przyjdzie nam ochota skompilować jakieś dawne programy, korzystające ze starszych wersji, to nie ma problemu. Mamy więc do dyspozycji coraz to nowsze interfejsy, "obciążone" coraz to większą ilością funkcji, które pozwalają na realizację coraz to nowych fanaberii programistów. Jak wspomniałem wyżej, w wersji 8 panowie z MS postanowili nam jak zwykle umilić życie i scalili dawniejsze DirectDraw, czyli moduł odpowiedzialny za grafikę 2D z Direct3D, który służył do generowania, jakże długo pożądanej i przez wielu umiłowanej ponad wszystko, grafiki 3D :-). Niektórzy twierdzą, że DirectDraw został w ogóle wyrzucony z pakietu, co nie jest do końca prawdą. Ci, którzy kiedykolwiek cokolwiek napisali z wykorzystaniem choćby tylko 3D, wiedzą, że bez DirectDraw nie można się obejść. Tworzenie obiektów, korzystanie z powierzchni, to wszystko zawierał moduł DirectDraw. Zresztą sami twórcy zapewniają, że moduły te zostały w pewien sposób "zintegrowane" i korzystanie z DirectX Graphics ma być o wiele prostsze, ale jak z tym będzie, to się jeszcze zobaczy. W każdym razie mamy jeden moduł i funkcje do 2D są zaszyte gdzieś w funkcjonalność tego modułu. Wygląda na to, że MS zakłada, że nie interesuje nas już grafika 2D na przykład i teraz wszyscy będziemy robić tylko 3 wymiary, no ale nas to nie martwi, prawda? Hmm... od czego by tu zacząć opis... może najpierw powiedzmy sobie, jakie cechy sprawiają, że DirectX Graphics w ogóle nadaje się do naszych celów:

- DirectX3D daje nam, programistom niezależny od urządzenia dostęp do urządzeń akceleratorów na niskim poziomie.
- Pozwala na bezpośredni dostęp do urządzeń graficznych (kart), zachowując kompatybilność (zgodność) ze starym, dobrym GDI.
- Oferuje popularne metody cieniowania obiektów - płaskie i Gourauda.
- Obsługuje wiele źródeł światła. Mamy też do dyspozycji wiele typów światła.
- Pełna obsługa materiałów i tekstur, włączając w to zaawansowane techniki, takie jak mipmapping.
- Dostarcza szerokie możliwości emulacji pewnych funkcji akceleratorów na sprzęcie nie posiadającym takich możliwości.
- Obsługuje transformacje i tzw. obcinanie.
- Obsługuje takie technologie jak: MMX, SIMD, SSE, 3D-Now!
- Działa na wszystkich wersjach Windows od 95 w górę, za wyjątkiem Windows NT.
- Obsługuje buforowanie z możliwością ustawiania różnej liczby buforów dla aplikacji pełnoekranowych.
- Obsługuje obcinanie geometrii dla aplikacji pełnoekranowych i pracujących w oknie (ang. *clipping*)
- Obsługuje bufory: *Z* i *W*.
- Daje jednoczesny dostęp do różnych obszarów pamięci karty grafiki.
- Daje bezpośredni dostęp do urządzenia renderującego, umożliwiając np. bezpośrednią zmianę rozdzielczości.
- Obsługuje mechanizmy pixel shader i vertex shader - niezwykle fajna rzecz! :-)

Tworzenie świata 3D w DirectX Graphics, podobnie jak w OpenGL-u, polega na wykorzystaniu wierzchołków, wielokątów oraz poleceń, które pozwalają utrzymać nad nimi kontrolę. DirectX umożliwia bezpośredni dostęp do transformacji i oświetlenia. DirectX Graphics jest nakładką, która jest bardzo blisko zbliżona do sprzętu. Z tego też powodu, aby móc tworzyć grafikę, musimy jawnie ustawić wszelkie transformacje, oświetlenie, wszelkie niezbędne macierze oraz, co bardzo ważne, musimy się najpierw dowiedzieć, jakie cechy posiada nasz sprzęt - chodzi o kartę grafiki. Jesteśmy świeżo po premierze GeForce 3 i to, co potrafi ta karta, chyba każdego, kochającego grafikę tworzoną w czasie rzeczywistym, może przyprawić o szybsze bicie serca :-). Gdy przejdziemy do szczegółowych tutoriali, będziemy wyjaśniać sobie bardziej skomplikowane pojęcia i zagadnienia grafiki 3D, gdyż nie sposób objąć tego wszystkiego w jednym krótkim artykule.

D3DX

Powiedzmy sobie jeszcze o czymś takim jak D3DX. Jest to nowa rzecz wprowadzona w wersji 7.0. Programiści piszący w OpenGL-u na pewno mieli przyjemność pracować z popularnym dodatkiem do GL-a, jakim jest GLUT, czyli biblioteka pozwalająca na łatwiejszą inicjalizację i zarządzanie urządzeniami (myszka, klawiatura) i, co bardzo ważne, w sposób niezależny od systemu operacyjnego. Wszystkie aplikacje pisane były jako aplikacje konsolowe, a tworzeniem okien do renderingu zajmował się właśnie GLUT. DirectX, jak wiemy, działa jak narazie tylko w Windows i obsługą wejścia/wyjścia oraz całej aplikacji zajmujemy się w sposób dla Windows specyficzny. Jednak tworzenie takich rzeczy jak macierze (dowiemy się następnym razem co to jest i po co jest to nam potrzebne) świata, widoku i rzutowania, tworzenie funkcji do działań na macierzach i tym podobne rzeczy zawsze musieliśmy pisać od nowa. Można oczywiście było sobie napisać własną bibliotekę albo po prostu kopiować odpowiedni kod. Ale programiści tworzący DirectX poszli nam na rękę i stworzyli coś takiego jak właśnie biblioteka D3DX. Czym ona jest? Zawiera zestaw funkcji, struktur, interfejsów oraz makr, które wiele z wyżej wymienionych przeze mnie czynności wykonują automatycznie, dając nam do ręki gotowe narzędzia. Mamy odpowiednie funkcje do działania na macierzach, o wiele łatwiejszą inicjalizację i ładowanie tekstur i modeli, wiele makr, które pomagają w przeliczaniu pewnych wartości (np. kątów z radianów na stopnie i odwrotnie, czy też kolorów). Wiele małych, acz bardzo przydatnych drobiazgów, których często brakowało pod ręką, przy pisaniu w starym stylu. Dobra jej znajomość bardzo ułatwi nam życie. Biblioteka ta zawiera także coś nowego, a jednocześnie bardzo fajnego, z czego postaramy się skorzystać, jak już więcej się nauczymy - **vertex i pixel shader**, ale na to przyjdzie jeszcze czas. To byłoby chyba tyle na początek. Wiemy po co nam DirectX, wiemy mniej więcej jak jest zbudowany, wiemy jak działa, przyjrzymy

się więc w następnej lekcji podstawowym zagadnieniom, jakie będą nam potrzebne, aby móc pojąć, o co w tej całej grafice 3D chodzi. Następnym razem będzie więc o układach współrzędnych, cieniowaniu, macierzach, buforach i prostokątach.