Programowanie [cz. 1] w MicroStation Basic

Artur Krawczyk

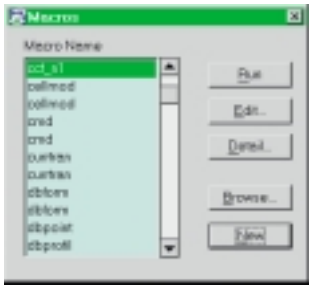
Niniejszym artykułem rozpoczynamy cykl publikacji poświęconych programowaniu makr za pomocą MicroStation Basic (MStB). W pierwszej serii artykułów zostaną przedstawione podstawy programowania w języku skryptowym MicroStation oraz przykłady jego zastosowań.

Prezentowany język programowania należy uznać za bardzo łatwy w użytkowaniu. Jego składnia jest prosta, a organizacja środowiska pracy nie wymaga żadnych dodatkowych konfiguracji MicroStation czy systemu operacyjnego. MStB charakteryzuje się dużymi możliwościami przetwarzania danych. Dzięki wspomnianej prostocie obsługi z powodzeniem może być wykorzystywany przez osoby, które nie mają żadnego doświadczenia w programowaniu. Praktycznie natychmiast po zainstalowaniu MicroStation środowisko Basica jest gotowe do użycia.

Dlaczego MicroStation Basic?

W środowisku MicroStation mamy do wyboru kilka języków programowania. Jednym z podstawowych kryteriów ich oceny jest szybkość działania oraz stopień „dostępu do wnętrza MicroStation", czyli możliwość bezpośredniego operowania na funkcjach użytych do budowy samego MicroStation (bulit-in functions). Biorąc pod uwagę tę cechę, należy stwierdzić, że najlepszy jest język MDL (również opisywany na łamach CCF), dość dobra jest także Java (JMDL). Wykorzystanie tych języków jest jednak możliwe pod warunkiem posiadania umiejętności programowania w języku C lub Javie.

Potrzebne są także własne narzędzia edytorskie, służące do pisania i zarządzania plikami z kodami źródłowymi programów. Dopiero w ten sposób przygotowany użytkownik jest w stanie podjąć naukę programowania MicroStation za pomocą MDL lub Javy. Natomiast implementacja Basica w środowisku MicroStation jest kompletna. Razem z programem dostarczany jest bowiem edytor pozwalający na uruchamianie i debugowanie (usuwanie błędów) kodu programu oraz edytor okien dialogowych aplikacji.



Rys. 1. Okno zarządzania makrami.

Rys. 2. Uruchomienie makra.

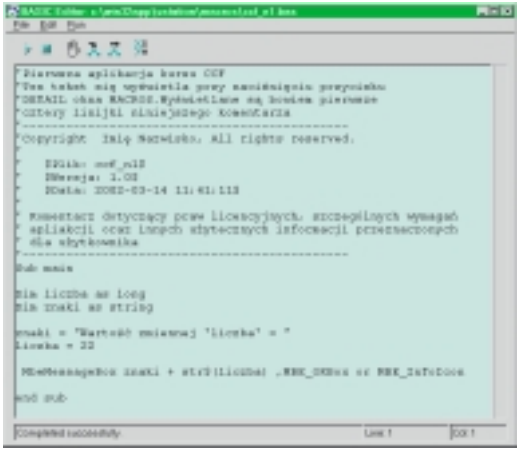
Dzięki tym elementom rozpoczęcie programowania w Basicu jest bardzo łatwe. Wadą Basica jest brak dostępu do jądra MicroStation. W celu wykorzystania niektórych funkcji MicroStation trzeba użyć rozszerzenia, o nazwie MicroStation Basic Extensions, dołączonego do zestawu standardowych poleceń Basica. Skrót Mbe jest typowym przedrostkiem wszystkich zaimplementowanych nazw funkcji MicroStation, które można wykorzystać w Basicu. Natomiast specyficzne dla MicroStation predefiniowane stałe mają przedrostek MBE_. Z powodu braku dostępu do jądra możliwości programów napisanych w Basicu są mniejsze, niż aplikacji stworzonych w innym języku. Należy jednak wziąć pod uwagę, że dostęp do jądra MicroStation powoduje, iż każda modyfikacja jądra najczęściej skutkuje koniecznością wprowadzenia zmian w aplikacji napisanej np. MOL, która korzysta bezpośrednio z funkcji wbudowanych (np. MDL). Tymczasem większość aplikacji napisanych w MicroStation Basic 5.5 działa równie dobrze w nowszych wersjach MicroStation: 5.7, 7.0 czy 7.1. Sprawnie pracują one także w środowisku najnowszej wersji MicroStation v.8.

Większość użytkowników programów CAD często nie posiada żadnej wiedzy na temat programowania, część ma doświadczenia w programowaniu za pomocą Turbo Pascala ze szkoły średniej czy też ze studiów. Wszystkie te osoby mogę zapewnić, że właśnie Basic jest dobrym punktem wyjścia do nauki programowania. W przeciwieństwie bowiem do MDL i JMDL, składni języka Basic można bez problemu nauczyć się w środowisku MicroStation. Wiedza zdobyta dzięki lekturze niniejszego kursu może wkrótce zaprocentować przy zakupie najnowszej 8 wersji MSt. Wersja ta, poza MicroStation Basic posiada zaimplementowane dużo bardziej zaawansowane wcielenie Basica – MicroStation Visual Basic

for Applications (MVBA). Według informacji udzielanych przez producenta wyraźnie zmalał dystans dzielący możliwości MStB od języków MDL czy JMDL.

Limity w MicroStation Basic

Przed podjęciem nauki programowania trzeba koniecznie poznać cechy środowiska, w którym będziemy pisać swoje programy. Projektanci Bentleya przyjęli bowiem założenie, że nawet bardzo źle napisane programy w Basicu nie mogą w żadnym przypadku zagrozić stabilności środowiska MicroStation. Jednym z kluczowych problemów podczas pisania aplikacji jest prawidłowa gospodarka pamięcią operacyjną. Najczęściej popełniane są błędy powodujące zawieszanie systemu CAD. Dlatego wprowadzono ograniczenia dotyczące gospodarki pamięcią przeznaczoną dla aplikacji języka Basic. Producent zdecydował się, między innymi, ograniczyć wielkości pliku wykonywalnego aplikacji Basic do 64 KB. Wielu użytkowników MicroStation przyzwyczaiło się do tego ograniczenia. Na listach dyskusyjnych zaobserwowałem typowe reakcje użytkowników na informacje o programowaniu za pomocą języków skryptowych. Charakterystycznym „męskim" zachowaniem osób zajmujących się programowaniem jest preferowanie atrybutu wielkości, który podświadomie przekładany jest na jakość kodu. Zdarzają się stwierdzenia użytkowników: „zacząłem programować w Basicu, ale 64 KB szybko przestało mi wystarczać". Co zatem oznacza granica 64 KB? Jest to limit wielkości pliku *.ba zawierającego pseudokod przygotowany do uruchomienia przez MicroStation. Natomiast sam tekst programu zapisujemy w pliku z rozszerzeniem *.bas. Na przykład makro, zawierające prawie 1000 linii kodu oraz kilka rozbudowanych okien dialogowych, po translacji do postaci *.ba osiąga zaledwie połowę dopuszczalnego limitu wielkości pliku. Możemy spokojnie pisać w Basicu, nie przejmując się komunikatem „Error:Overflow" interpretowanym jako przekroczenie tego limitu. Komunikat ten pojawia się, gdy początkujący użytkownik stara się przypisać zmiennej typu integer wartość większą od 32 767, co powoduje przekroczenie jej zakresu.



Rys. 3. Okno edytora.

Ponadto producent określił limity dotyczące przestrzeni danych dla zmiennych globalnych (public) – do 16KB i przestrzeni danych dla zmiennych lokalnych (private) – 32 KB. Natomiast do przechowywania danych w postaci łańcuchów znaków (string) oraz tablic (array) przeznaczono blok alokacji pamięci o wielkości 1 MB. Zatem w makro o wielkości 64 KB można określić zmienne przekraczające 1 MB. Wystarczy przecież zdefiniować kilka bajtów jednej zmiennej, aby w całym 32 MB pliku *.dgn zmienić wybraną cechę danych graficznych.

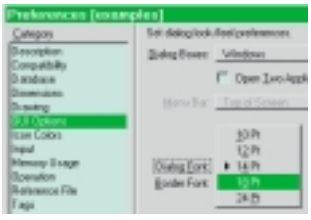
Przed przystąpieniem do pisania programu koniecznie trzeba zapoznać się z fragmentem dokumentacji dotyczącym wyboru języka programowania. Opisane bowiem zostały sytuacje, gdy aplikacja musi zostać napisana w bardziej zaawansowanym języku niż Basic. Dotyczy to na przykład stosowania niemodalnych okien dialogowych, dynamicznego wyświetlania elementów graficznych podczas ich rysowania oraz możliwości używania zewnętrznych bibliotek (dlm). Takie wymagania w stosunku do aplikacji wymuszają wybór języka MDL. Szczególnie zniechęcające jest zalecenie używania Basica tylko do przetwarzania elementów graficznych, bez możliwości kontrolowania ich wstawiana do pliku rysunkowego. Ograniczenia języka można czasami obejść, o czym napiszemy w kolejnych odcinkach kursu programowania w MStB.

Środowisko pracy, pierwszy program

Podczas instalowania MicroStation należy pamiętać o instalacji pliku pomocy, tzw. helpa. Ze strony docs.bentley.com/dl.asp można pobrać najnowszą dokumentację do MStB, w formacie *.pdf.

Aby rozpocząć pracę z makrami, należy z głównego menu wybrać opcję Utilites -> Macros. W oknie (rys. 1) wyświetlona zostanie lista dostępnych makr w domyślnym katalogu makr.

Okno ułatwia nam uruchamianie makr, wystarczy bowiem wskazać makro z listy i uruchomić je przyciskiem Run. Inną możliwością uruchomienia makra, o ile użytkownik zna jego nazwę, jest wpisanie w oknie key-in komendy: macro nazwa_makra (rys. 2).

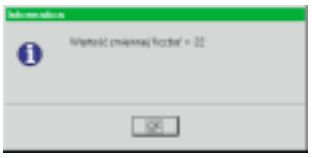


Rys. 4. Ustalenie wielkości czcionki.

Jeśli makro znajduje się poza katalogiem, możemy je uruchomić przyciskiem Browse, który otwiera nam okno managera plików, pozwalając na wybranie katalogu i wskazanie pliku makra. Po zatwierdzeniu

wyboru danego makra pojawia się możliwość wyboru edycji lub uruchomienia makra. Ciekawą i niezwykle użyteczną funkcję udostępnia nam opcja Detail. Jeżeli makro posiada w swoich pierwszych czterech linijkach jakiś komentarz pozwalający zidentyfikować jego przeznaczenie, może ona spowodować jej wyświetlenie. Ostatnie dwie opcje New i Edit uruchamiają edytor Basica (rys. 3). Ze względu na wysokie rozdzielczości ekranu często zdarza się, że czcionka stosowana w edytorze jest zbyt mała. Należy wtedy uruchomić menu Workspace->Preferences i zmienić jej wielkość (rys. 4).

Po zmianie ustawień trzeba uruchomić ponownie MicroStation. Przyciskiem New w oknie Macros ponownie uruchamiamy edytor Basica. W nowym pliku przystępujemy do napisania pierwszego swojego makra. Będzie ono po prostu wyświetlać komunikat. Zanim przystąpimy do pisania tekstu programu, musimy wpisać komentarz z informacją o przeznaczeniu nowo powstającej aplikacji. Ułatwi to wykorzystanie przycisku Detail w oknie Macros, który tę informację może wyświetlić bez edytowania lub uruchamiania makra.



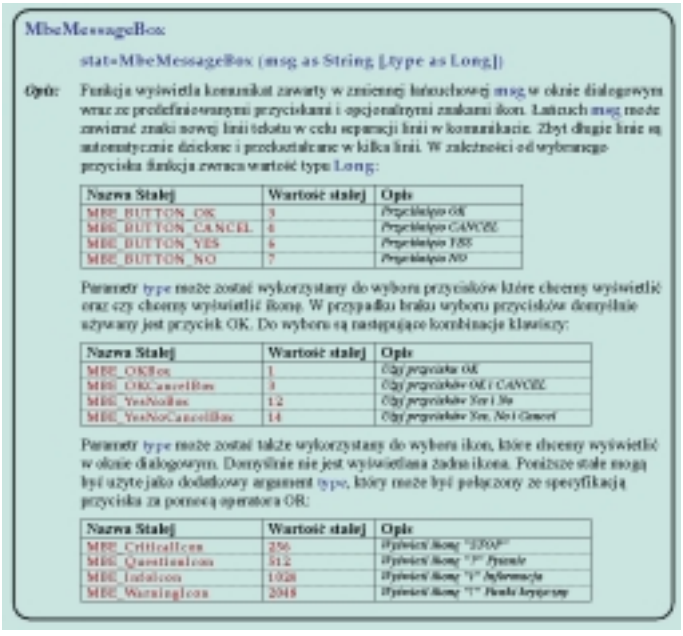
Rys. 5. Komunikat naszego programu.

Objaśnienia oznaczeń znajdujących się w programie:

- Znak apostrofu – rozpoczyna tekst komentarza. Rozbudowany komentarz został przedstawiony w tekście makra na rys. 3. Komentarz nie jest interpretowany przez kompilator;
- Sub main – wykonywanie makra zaczyna się zawsze od instrukcji sub main, która rozpoczyna część wykonywalną każdej aplikacji;
- Dim liczba as Integer i Dim znaki as String – definicja dwóch zmiennych. Pierwsza zmienna „liczba” została określona jako liczba całkowita. Zmienna „znaki” została zdefiniowana jako łańcuch znaków (w tym przypadku tekst). Dla zmiennych używamy słów kluczowych Dim nazwa_zmiennej as typ_zmiennej;
- znaki = „Wartość zmiennej 'liczba' = ” oraz liczba = 22 – nadawanie wartości zmiennym. Tekst dla zmiennej łańcuchowej musi zostać podany w cudzysłowie. Po otwarciu cudzysłowu znaki przestają być interpretowane jako komendy i zamieniają się w zwykły tekst, który będzie wyświetlany. Apostrof przy słowie liczba i samo słowo liczba staną się tekstem (łańcuchem znaków);
- MbeMessageBox znaki + str$(liczba), MBE_OKBox or MBE_InfoIcon – użycie obiektu rozszerzeń MicroStation dla Basica (charakterystyczny przedrostek Mbe). Dzięki temu zyskamy możliwość wyświetlania standardowego okna dialogowego. Argumentem tego obiektu powinna być zmienna łańcuchowa. Jest nią zmienna „znaki”. Natomiast zmienna „liczba” musi zostać przekształcona do postaci literału. W tym celu używamy funkcji str$() konwertującej zmienne liczbowe do postaci łańcucha znaków. MessageBox może mieć tylko jeden łańcuch znaków, stąd konieczność „sklejenia” znaków w jeden łańcuch. Po przecinku podajemy nazwy stałych, które określą, jakie elementy projektowanego okna będą wyświetlane w czasie jego wywołania w programie (rys. 7);
- End sub – zakończenie części wykonywalnej naszego programu.

Po uruchomieniu makra, powinniśmy otrzymać komunikat przedstawiony na rys. 5. W wyniku uruchomienia programu zostaje wyświetlone modalne okno dialogowe. Określenie modalne oznacza, że nie możemy wykonać żadnych innych operacji dopóki okno dialogowe nie zostanie zamknięte. Opis obiektu MbeMessageBox, przedstawiony na rys. 6, ułatwi decyzję o ewentualnych modyfikacjach programu. W pierwszej kolejności można zmienić wartości stałych określających wybór ikony oraz stałych określających wybór przycisków widocznych w oknie dialogowym.

W drugiej części kursu programowania MStB przedstawimy jedną z trzech metod przetwarzania danych graficznych w pliku *.dgn.


MbeMessageBox

stat=MbeMessageBox (msg as String [,type as Long])

Opis: Funkcja wyświetla komunikat zawarty w zmiennej łańcuchowej msg w oknie dialogowym wraz ze predefiniowanymi przyciskami i opcjonalnymi znakami ikon. Łańcuch msg może zawierać znaki nowej linii (ctlotu w celu separacji linii w komunikacie. Zbyt długie linie są automatycznie dzielone i przekształcane w kilka linii. W zależności od wybranego przycisku funkcja zwraca wartość typu Long:

Nazwa Stałej	Wartość stałej	Opis
MBE_BUTTON_OK	3	Przycisk/przy OK
MBE_BUTTON_CANCEL	4	Przycisk/przy CANCEL
MBE_BUTTON_YES	6	Przycisk/przy YES
MBE_BUTTON_NO	7	Przycisk/przy NO

Parametr type może zostać wykorzystany do wyboru przycisków które chcemy wyświetlić oraz czy chcemy wyświetlić ikonę. W przypadku braku wyboru przycisków domyślnie używany jest przycisk OK. Do wyboru są następujące kombinacje klawiszy:

Nazwa Stałej	Wartość stałej	Opis
MBE_OKBox	1	Ikona przycisku OK
MBE_OKCancelBox	3	Ikona przycisków OK i CANCEL
MBE_YesNoBox	12	Ikona przycisków Yes i No
MBE_YesNoCancelBox	14	Ikona przycisków Yes, No i Cancel

Parametr type może zostać także wykorzystany do wyboru ikon, które chcemy wyświetlić w oknie dialogowym. Domyślnie nie jest wyświetlana żadna ikona. Poniższe stałe mogą być użyte jako dodatkowy argument type, który może być połączony ze specyfikacją przycisku za pomocą operatora OR:

Nazwa Stałej	Wartość stałej	Opis
MBE_CriticalIcon	256	Wyświetl ikonę "STOP"
MBE_QuestionIcon	512	Wyświetl ikonę "?" Pytanie
MBE_InfoIcon	1024	Wyświetl ikonę "i" Informacja
MBE_WarningIcon	2048	Wyświetl ikonę "!" Ostrzeżenie

Rys. 6. Opis obiektu MbeMessageBox.

Na stronie http://galaxy.uci.agh.edu.pl/~artkraw znajdą Państwo, opisaną w niniejszym artykule, aplikację