



Serijska komunikacija

Najjednostavnija realizacija serijske komunikacije u Visual Basicu je pomoću korisničke kontrole *MSComm Control*.

 Aleksandar Tošić

Visual Basic kao RAD (Rapid Application Development) alat omogućava programeru da relativno brzo pronađe rešenje za neki problem. Poželjna osobina programera koja se zahteva pri izradi Windows aplikacija je snalažljivost. Ako je VB odabran za izradu aplikacije (ovde je korištena verzija 5.0), upotreba *Helpa* je veoma korisna. Pronalaženje delova *Helpa* koji su interesantni za serijsku komunikaciju je moguće ostvariti preko izbora opcije *Index* i reči za pretragu *communications errors*. Rezultat pretrage je: *communications errors*. Pritiskom na taster *Display* na ekranu *Helpa* se pojavljuje naslov: *OnComm Event*. Ovo još ne zadovoljava naše potrebe. Sledeće što preostaje je izbor *opcije See Also*. Ono što nam treba je *MSComm control* (u verziji VB 4.0 umesto *MSComm* je stajalo *Communications*), odnosno korisnička kontrola (custom control) koja podržava rad sa serijskom komunikacijom.

Ikonica koja simbolizuje serijsku komunikaciju je *telefon*. U fazi izrade aplikacije u *VBIDE* (Visual Basic Interface Development Environment) u **ToolBox-u** VBAsica inicijalno nema ikonice sa telefonom. Da bi se pojavila tražena ikonica potrebno je u glavnom meniju odabrati opciju **Projects**, i u podmeniju opciju **Components**. Rezultat je niz dodatnih kontrola koje možemo postaviti na *ToolBox*. Potrebno je »čekirati« izabranu korisničku kontrolu, u ovom slučaju **Microsoft Comm Control 5.0** (verzija VB 5.0). Posle pritiska na dugme *Apply*, »telefon« se vidi u *ToolBox-u* (na levoj strani ekrana).

Ime fajla u kome je sadržana komunikaciona kontrola je **MSComm32.ocx**. Ekstenzija **ocx** označava **ActiveX** fajl. Možda je važno kratko napomenuti da reč **ActiveX** predstavlja tehnologiju »pakovanja« napravljenih klasa objekata u fajlove koji se posle toga mogu distribuirati ostalim korisnicima. U ovom slučaju klasa koja je upakovana u fajl **MSComm32.ocx** se zove **MSComm**.

Jedan objekt – jedan port

Računar ima dva serijska porta koje obično označavamo sa 1 i 2. Upotreba jednog ili oba serijska porta za komunikaciju zahteva po jednu *MSComm* kontrolu za svaki port. Dodavanjem kontrole na *Form-u* u fazi izrade aplikacije stvara se objekat koji je namenjen za rad sa izabranim portom. Objekti se redom automatski imenuju *MSComm1*, *MSComm2*, itd. Međutim, kako postoje dva serijska porta, indeksi veći od 2 verovatno neće biti potrebni. Primetimo da je *MSComm* ime za klasu objekata, dok su *MSComm1*, *MSComm2*, itd, konkretni objekti sa kojima se

MSComm Control Constants

OnComm Constants

Constant	Value	Description
comEvSend	1	Send event.
comEvReceive	2	Receive event.
comEvCTS	3	Change in clear-to-send line.
comEvDSR	4	Change in data-set ready line.
comEvCD	5	Change in carrier detect line.
comEvRing	6	Ring detect.
comEvEOF	7	End of file.

Error Constants

Constant	Value	Description
comEventBreak	1001	Break signal received
comEventCTSTO	1002	Clear-to-send timeout
comEventDSRTO	1003	Data-set ready timeout
comEventFrame	1004	Framing error
comEventOverrun	1006	Port overrun
comEventCDTO	1007	Carrier detect timeout
comEventRxOver	1008	Receive buffer overflow
comEventRxParity	1009	Parity error
comEventTxFull	1010	Transmit buffer full
comEventDCB	1011	Unexpected error retrieving Device Control Block (DCB) for the port

može manipulirati. Dve stvari su važne za klasu *MSComm*:

1. *Properties* (osobine klase)
2. *Events* (dogadjaji vezani za klasu).

Postoji samo jedna procedura namenjena za obradu događaja vezanih za serijsku komunikaciju: *OnComm Event*. U ovu proceduru treba smestiti programski kod koji će obraditi različite vrste događaja i grešaka koje se mogu javljati prilikom serijske komunikacije (gore). Na slici desno se vide osobine kontrole za komunikaciju koje se mogu koristiti u fazi izrade. Postoji njih nekoliko sa kojima programer (ako koristi serijsku komunikaciju) treba biti familijaran:

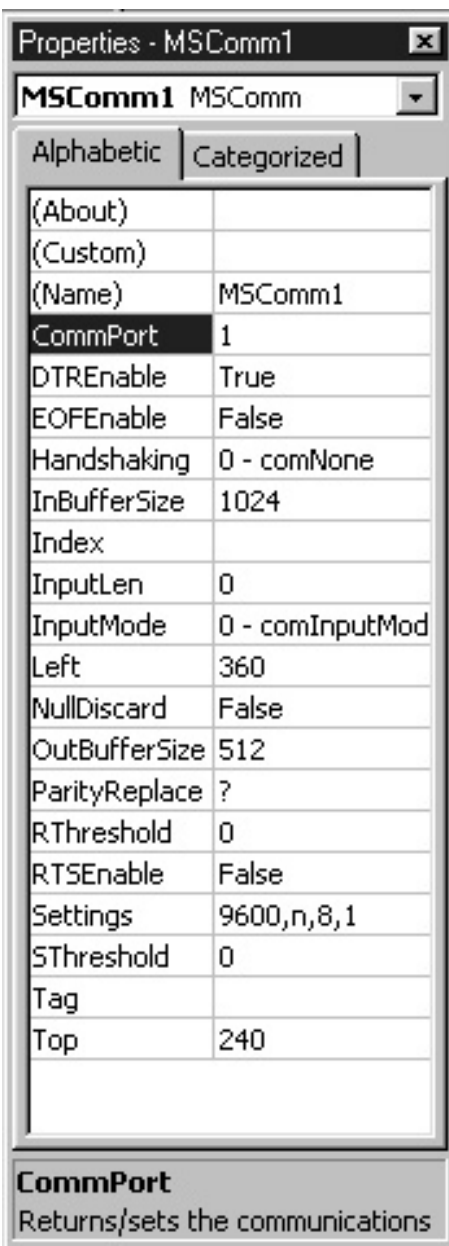
- *CommPort* postavlja i vraća broj komunikacionog porta (port number)
- *Settings* postavlja i vraća brzinu prenosa (baud rate), paritet (parity), broj bitova podataka (data bits) i stop bite

(stop bits) koje se mogu koristiti pre starta aplikacije, i

- *PortOpen* Postavlja i vraća stanje komunikacionog porta. Otvara i zatvara komunikacioni port.
- *Input* Uzimanje (i istovremeno uklanjanje) karaktera iz prijemnog bafera
- *Output* Upisuje karaktere (string) u bafer za slanje podataka

koje se koriste u run modu.

Postavljanje vrednosti osobina kontrole i pisanje procedure za obradu komunikacionih događaja će biti predstavljeni u sledećem broju.



Peter Norton Norton **Visual Basic 6**

Osobina koja je karakteristična za Nortonove knjige je da menjaju "crnu kutiju" u "staklenu kutiju". Pažljivijim čitanjem ove obimne knjige (ili nekih delova) čitalac se može uveriti da način izlaganja opravdava navedeni kompliment. Oni koji su nestrpljivo čekali knjigu o Visual Basicu 6, dobili su zaista obilje materijala za proučavanje. Knjiga može zadovoljiti sve kategorije čitalaca, od onih koji nameravaju koristiti VBasic za pravljenje zabavnih aplikacija do ozbiljnih programera. Za početnike je najinteresantnije prvo poglavlje u kome se opisuje rad u VBasic okruženju. Slede poglavlja o objektno orijentisanom programiranju, korištenju printera, radu sa bazama podataka, integraciji sa ostalim aplikacijam (Microsoft Office 97) i razvoju Interneta, koja će čitaocu pružiti mogućnost da ovlada ovom aktuelnom problematikom. Teme za naprednije programere su obrađene u poslednjem poglavlju i obuhvataju rad sa Windows API funkcijama, INI fajlove i sistemski registar, korišćenje čarobnjaka i profesionalni VBasic razvoj. Uz knjigu se dobija i CD sa programima koji će čitaocu pomoći da efikasnije proučava atraktivno programiranje Windows aplikacija u VBasicu. Ovo je knjiga čija se aroma može osetiti, a zasniva se na bogatom programerskom iskustvu autora.

Izdavač:

Kompjuter biblioteka

Vladana Šićevića 19

32 000 Čačak

tel/fax: 032/32 322

