18.A. Egy kisvállalat informatikusaként a telefonos ügyfélszolgálat munkáját támogató szoftver elkészítését kapja feladatként.
18.1 Ismertesse a szoftver ergonomikus használatához szükséges felület kialakításának főbb szabályait!
18.2 Mutassa be a szoftver fejlesztéséhez használható vizuális fejlesztőkörnyezet kényelmi szolgáltatásait!
18.3 Ismertesse a szoftver felületének kialakításához használható vizuális komponensek szerepét, főbb tulajdonságait!
Kulcsszavak, fogalmak:
- Szoftver ergonómia:
,– átláthatóság, – kezelhetőség, – szabványosság, – stb.
- Vizuális fejlesztőkörnyezetek szolgáltatásai:
– kiemelések, automatikus kiegészítések,
– hibakeresési szolgáltatások, – stb.
- Vizuális komponensek:
– gomb, – címke, – beviteli mező, – stb.

18.1 Ismertesse a szoftver ergonomikus használatához szükséges felület kialakításának főbb szabályait!
Csakúgy, amint az ergonómia alakítja a munkafolyamatokat, eszközöket az ember számára megfelelővé, kis erőkifejtéssel működtethetővé, úgy illeszti a szoftver-ergonómia a számítógépes rendszereket az ember kognitív és intellektuális tulajdonságaihoz, cselekvési, és észlelési jellemzőihez. Úgy alakítja az információközlés módját, hogy az az ember számára a lehető legkényelmesebb legyen. Azon túl, hogy a funkcionális elvárásokat teljesíti, egy rendszernek könnyen használhatónak kell lennie. A szoftver-ergonómia biztosítja, hogy a felhasználók minél könnyebben végezhessék el munkájukat, minél könnyebben tudjanak tájékozódni a szükséges funkciók és adatok között. Ez amellett, hogy elismertebbé, közkedveltebbé teszi a megfelelően kialakított alkalmazásokat, legtöbbször a használatuk során időt megtakarítva gazdaságosabbá teszi azok alkalmazását. Összességében a szoftver-ergonómia célja, hogy az alkalmazások jól használhatóak, lehetőleg egyszerűek, az emberre szabottak legyenek.

Használhatóság alatt egy minőségi tulajdonságot értünk, mely arra vonatkozik, hogy mennyi erőfeszítést igényel az adott rendszer használata. A használhatóság elég összetett tulajdonság,ntöbb komponens határozza meg:
· Tanulhatóság
· Hatékonyság
· Megjegyezhetőség
· Hibák
· Elégedettség

A tanulhatósággal jellemezzük, hogy az adott rendszer mennyire illeszkedik az emberi agy memóriaszerkezetéhez. A megismerési folyamat az észleléssel kezdődik. Ehhez a rövid távú memóriára van szükség, mely azonban csak kevés, meghatározott (általánosan 7±2) dolgot tud elraktározni, s – mint a nevéből adódik – ezeket is csak kevés ideig, mintegy 20 másodpercig. Az észlelés folyamata hierarchikus: először a nagyobb, színes, figyelemfelkeltő elemeket vesszük észre, majd lépésekben a többit. Ezért mindenképpen fontos, hogy a ténylegesen fontos elemek megfelelően ki legyenek emelve, megragadják a felhasználó figyelmét. Az észlelést követi az értelmezés, amikor is jelentés tulajdonítódik a látottakhoz. Kialakul az ún. célképzet: létrejön egy kognitív modell arra vonatkozóan, hogy mit akarunk csinálni. A modell a későbbiek során változhat, s természetesen gyakran változik is (a rendelkezésre álló információk számának növekedése következtében), vagy akár több modell is létrejöhet. Meg kell azonban jegyezni, hogy jellemzően, ha egyszer sikerült valamilyen módon elérni célunkat, legközelebb is úgy fogunk cselekedni – habár lehetséges, hogy nem az volt a legoptimálisabb megoldás.

Hatékonyság A rendszerünk hatékonyságát leginkább a következő kérdésekre adott válaszokkal mérhetjük:
· Használható-e arra, amire készült?
· Mennyire eredményesen végez el egy adott feladatot a felhasználó?
· Milyen gyorsan tudja elvégezni munkáját a kezelő?
Vegyük figyelembe, hogy pl. egy form kitöltése során milyen hibák ronthatják a hatékonyságot?
Sok esetben – leginkább nagy mennyiségű szöveges adat bevitelekor – az egér használata jelentősen lassíthatja a folyamatot. Ilyenkor jelenthet előnyt, ha a sok kattintgatás helyett tabokkal is végig tudunk menni a felületen – minden akadály nélkül (természetesen figyelni kell a megfelelő sorrendre). A jó felület már az adatbevitel folyamán jelzi a hibákat, megakadályozva ezzel a továbbmenetel előtti, felugró hibaüzeneteket. Gyakori hibaforrást jelent azonban a validációt végző aszinkron hívások miatt elveszett fókusz (pl. hibás szöveget tartalmazó mezőben való visszatörlés esetén a fókusz átugrik egy a már megfelelő adatok miatt engedélyezésre került gombra, megzavarva ezzel a közben tovább gépelő felhasználót).

Megjegyezhetőség Ahhoz, hogy egy alkalmazás jól használható legyen, fontos, hogy kövessük a szabványos megoldásokat, tartsuk magunkat a már jól bevált konvenciókhoz. A felhasználó ugyanis akaratlanul is megjegyzi, hogyan jutott el egy bizonyos szinten elhelyezkedő menüponthoz, milyen lépések kellettek az adott feladat elvégzéséhez. Az így megszerzett tudást pedig önkéntelenül is megpróbálja alkalmazni a többi alkalmazás kezelése esetén is. Gondoljunk csak bele, milyen problémát okozna, ha minden alkalmazás másképp működne, az azonos jelzések más-más jelentéssel bírnának. Amennyiben tartjuk magunkat ezen szabályokhoz (pl. a menü szinte mindig a bal felső sarokban van, a szokásos menüpontok általában egyazon sorrendben helyezkednek el), nem csak azt tudjuk biztosítani, hogy az alkalmazás hosszabb használat nélküli idő eltöltése után is újra könnyen használható maradjon, de a már korábban megszerzett tudását felhasználhassa a kezelő, valamint az itt szerzett tapasztalatok is alkalmazhatóak legyenek más rendszerek esetében is.

Hibák
Hibák alatt értjük, hogy:
· mennyi hibát ejt a felhasználó használat közben
· mennyire súlyosak ezek a hibák
· az ejtett hibák könnyen korrigálhatóak-e

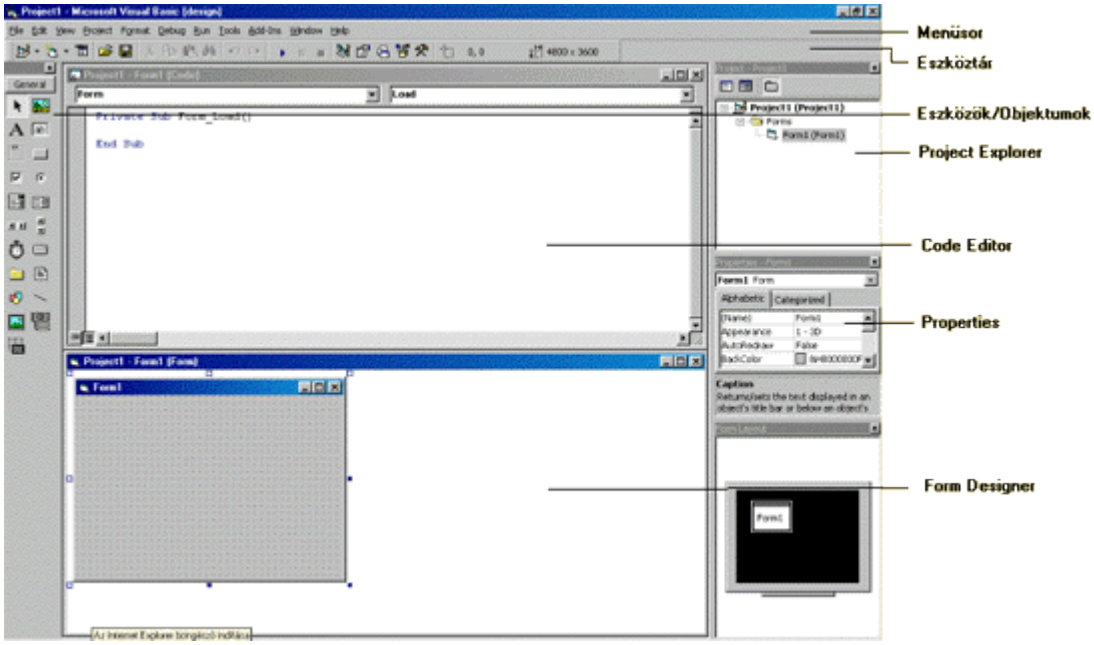
FELHASZNÁLÓI FELÜLETEK FONTOS TERVEZÉSI SZABÁLYAI
· A rendszernek mindig tájékoztatnia kell a felhasználót arról, hogy éppen mi történik. Fontos, hogy ha a felhasználó valamit végrehajt, arról legyen érthető visszajelzés. Például egy egyszerű keresésnél, a felhasználó beírja a kulcsszavakat, és rányom a keresésre. A rendszer elindítja a

keresést, mivel azonban sok az adat, egy-két másodpercig eltarthat a lekérdezés. Ha nem történik semmi szemmel látható a képernyőn, akkor szinte biztos, hogy még 3-4-szer rá fog kattintani a keresés gombra. Ezt elkerülhetjük valamilyen visszajelzéssel (pl. a képernyő leszürkítése, pörgőforgó ikon), illetve emellett érdemes a gombokat is letiltanunk erre az időre.
· A rendszernek a felhasználó nyelvét kell beszélnie. Azokat a szavakat, terminológiákat kell használnia, amit kezelője ismer, és használ.
· Biztosítani kell a reverzibilitást: a felhasználónak lehetőséget kell biztosítani, hogy megszakíthassa, visszavonhassa a már elkezdett folyamatot.
· A rendszernek konzisztensnek kell lennie: az azonos dolgokat azonos, a hasonló dolgokat hasonló módon és helyen kell jelölni. Ez biztosítja a könnyebb megjegyezhetőséget.
· Meg kell védeni a felhasználót attól, hogy végzetes hibát kövessen el. A nem triviális, destruktív műveleteket ki kell védeni, illetve megerősítést kell kérni!
· Minél kevesebb információra kell emlékeznie a felhasználónak, annál jobb. Ne kényszerítsük a felhasználót arra, hogy egyik oldalról a másikra meg kelljen jegyeznie dolgokat.
· Ne csak egyfajta felhasználónak készítsük az alkalmazást! Biztosítsunk gyorsításokat pl.gyors billentyűkkel a haladó felhasználóknak, tegyük lehetővé bizonyos lépések átugrását, illetve nyújtsunk támogatást a kezdőknek!
· A hibaüzeneteknek mindig érthető, magyarázó hangnemben kell megjelenniük. Ha van lehetőség rá, megoldási javaslatot is tehetünk.
· A legjobb, ha használható a rendszer felhasználó kézikönyv nélkül is, de mindenképpen szükséges dokumentációt, segédletet is biztosítani.
· Ügyeljünk az esztétikusságra! A felhasználói felület (UI – UserInterface) minőségét befolyásolja a felhasznált színek száma és fajtája, a grafikai objektumok élessége, felbontása, valamint az olvashatóság. A színes felület vonzóbb, gyorsabban áttekinthető, könnyebben megjegyezhető, azonban ügyeljünk arra, hogy a szín csak hozzáadott érték legyen, színtévesztők számára is egyértelmű legyen! Először inkább egyszínűre tervezzük meg a felületet, hiszen a rossz felületet színekkel nem lehet helyrehozni. Kiemelni csak kevés elemet érdemes. Legfeljebb 7 színt használjunk egy oldalon, mivel a túl sok szín között nehéz eligazodni, a túl harsány, vibráló színek pedig fárasztják a szemet.
· A felületi elemek elhelyezése legyen a szemnek kellemes, használjunk azonos sorközöket, az egyazon célt szolgáló elemek ugyanúgy nézzenek ki, méretükben, színükben ne térjenek el! A hasonló feladatokat ellátó elemeket csoportosítsuk, de csak egy domináns részfelület legyen!

18.2 Mutassa be a szoftver fejlesztéséhez használható vizuális fejlesztőkörnyezet kényelmi szolgáltatásait!

A Visual Basic integrált fejlesztőkörnyezete

A Visual Basic munkafelületét gyakran nevezik integrált fejlesztőkörnyezetnek - integrated development environment, IDE. Ezt az elnevezést azért kapta mivel egy közös környezetben több különböző funkciót is egyesít. Ugyanabban a környezetben végezhetjük el a felhasználói felületek tervezését, a programkód szerkesztését és a hibakeresést. (1. ábra)



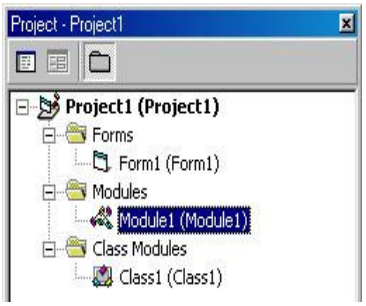
Menüsor Ez tartalmazza a Visual Basic programozás során használatos parancsokat. Megtalálhatjuk itt a legtöbb programra jellemző menüpontokat is mint pl.: File (Fájl), Edit (Szerkesztés), View (Nézet), Window (Ablak) és Help (Súgó). Ezeken kívül természetesen a programozással kapcsolatos menüpontokat is tartalmaz a fejlesztőkörnyezet mint pl.: Debug (Hibakeresés), Run (Futtatás), Project (Projekt).

Helyzetérzékeny menük Gyakran végzett műveletek gyors elérését teszik lehetővé. Ezen menü megjelenítését a jobb egérgomb kattintásával végezhetjük el. A menüben elérhető parancsok listája helyzetfüggő, vagyis attól függ hogy hol kattintottunk a jobb egérgombbal.

Eszköztárak A leggyakrabban használt parancsok gyors elérését teszik lehetővé. Az eszköztár egyes ikonjain kattintva az általuk képviselt parancs azonnal végrehajtódik. Alapértelmezésként a Basic az úgynevezett Standard (Szokásos) eszköztárak megjelenítésével indul, de a View menü Toolbars parancsával további eszköztárakat jeleníthetünk meg.

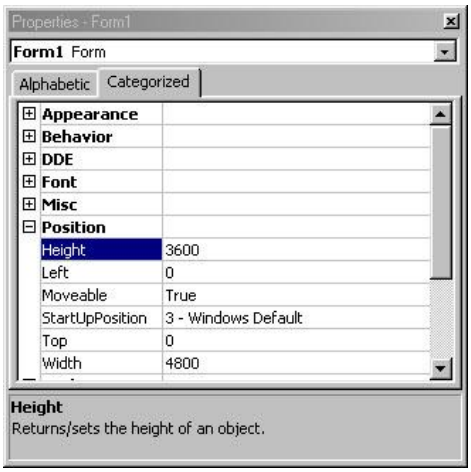
Eszközök/Objektumok Olyan eszközöket/objektumokat tartalmaz, amelyeket az űrlapok, felhasználói felületek elkészítése során alkalmazhatunk. Ilyenek pl.: TextBox (adatbeviteli ablak), Label (címke), CommandButton (nyomógomb), stb. Egy új projekt készítésekor van egy alapértelmezett beállítás, amikor csak a legáltalánosabb elemeket jeleníti meg a VB. Azonban lehetőség van új objektum, vezérlőelem, komponens hozzáadására, de erről egy kicsit később.

Project Explorer ablak Először is magyarázatra szorul, hogy mit is nevezünk VB projektnek? A projekt már nem egy egyszerű program, a projekt egy fejlesztési feladat, az elkészítendő alkalmazás fájljainak együttese, azaz egy projekt fogja össze a fejlesztett alkalmazás összes állományát és tartalmazza a fejlesztőrendszer adott alkalmazáshoz tartozó beállításait, működési paramétereit. A projektcsoport pedig egy olyan projekt ami nem fájlokat, hanem további projekteket foglal magába.



Most, hogy már tudjuk mi is a projekt egy kicsit talán érthetőbbnek tűnik a Project Exlorer (2. kép) ablak tartalma. Ez az ablak tartalmazza az aktuális projekthez tartozó modulok, űrlapok és egyéb fájlok listáját a megfelelő csoportosításban. A jobb egérgomb lenyomásával megjeleníthető menü segítségével kedvünk szerint módosíthatjuk a projektet. Hozzáadhatunk illetve elvehetünk elemeket a listából. Ilyenkor természetesen a módosítások elmentődnek a projektünkbe is. A Project Explorer ablakon három ikont találunk ezek közül kettő (a két baloldali) a programkód (View Code) és az űrlap tervezés (View Object) közti egyszerűbb váltást teszi lehetővé - természetesen ezek a váltások az éppen aktuálisan kiválasztott űrlap vagy modul esetén érvényesek.

Properties ablak Talán az egyik legfontosabb és a legtöbbet használt ablak a Visual Basic-ben. Segítségével könnyedén megváltoztathatjuk az aktuálisan kiválasztott űrlap, vagy vezérlőelem egyes tulajdonságait. Jelentősége még abban rejlik, hogy egyes objektumok rendelkezhetnek olyan csak tervezési időben beállítható tulajdonságokkal, amiket csak ezen az ablakon keresztül módosítható. Közvetlenül az ablak címsávja alatt található az objektum lista, ami az éppen aktív programmodul objektumainak, vezérlőelemeinek nevét és típusát mutatja. A listából tetszés szerint választhatunk és ilyenkor mindig az aktuálisan kiválasztott elem tulajdonságai kerülnek megjelenítésre.



Mint látható (3. kép) a Properties ablakon két fül található: *Alphabetic, Categorized.* E két fül segítségével az éppen aktuális vezérlőelem tulajdonságainak megjelenési sorrendjét választhatjuk meg. Lehetőség van a tulajdonságok abc sorrendben történő megjelenítésére (első fül), valamint különböző kategóriák alapján való csoportosításukra (második fül). Talán a kategóriák szerinti megjelenítés egy kicsit kényelmesebb hiszen a vezérlők összetartozó tulajdonságai egy helyen, egy csoportban helyezkednek el, így nem kell állandóan a listán navigálnunk a keresésük miatt. A tulajdonság mezők listája két hasábból áll. Az első tartalmazza a nevét, míg a másikban az aktuális érték olvasható. Az éppen kijelölt tulajdonság értékének módosítása attól függ, hogy milyen jellegű az adott tulajdonság. Ha hozzá csak néhány megengedett érték rendelhető, akkor a megjelenő listából történő választás segítségével hajtható végre a módosítás (pl.: ilyen tulajdonság a Visible ami csak két lehetséges értéket vehet fel *True vagy False).* Más esetekben az új érték egyszerűen begépelhető.
A Properties ablakon megjeleníthető egy ún. Státussor, ahol az éppen aktuális tulajdonságról kaphatunk egy rövid kis leírást. Ezt úgy érhetjük el, hogy az ablakon a jobb egérgombbal kattintunk és a megjelenő menüből kiválasztjuk a *Descriptionpontot.*

Form Designer Másnéven űrlaptervező ablak. Ennek segítségével tervezhetők meg az egyes felhasználói felületek, űrlapok. A VB egyik legfontosabb ablaka, mivel segítségével szinte pillanatok alatt készíthetünk adatbeviteli ablakokat, speciális adatmegjelenítő ablakokat anélkül, hogy komolyabb programkódok írásához fognánk. Talán ez az egyik oka, hogy manapság egyre többen kezdik használni a Visual Basic-et vagy a hozzá hasonló vizuális programozási nyelvek valamelyikét. Minden projekt annyi űrlaptervező nézetet tartalmaz, ahány űrlapot az tartalmaz.

Code Editor Ide jön a lényeg! Ez az ablak a tényleges programozói felület. Itt írhatjuk meg eljárásainkat, szerkeszthetjük az egyes vezérlők eseményeinek kódrészleteit, definiálhatunk új osztályokat. A projekt minden egyes moduljához, űrlapjához saját ablak tartozik, tehát a kódokat nem kell ömlesztve egy helyen elhelyeznünk, hanem minden modulhoz, űrlaphoz vagy osztályhoz csak a saját kódrészlete tartozik. Nagyjából ezek a Visual Basic fejlesztőkörnyezetének legfontosabb felépítői. Van még egy két ablak - Debug ablakok - melyekről nem beszéltem de ezek használatához már elengedhetetlen egy kevés kis VB-s programozási ismeret. Ezekről majd később.Mivel a programozás során leginkább a menüsor és az ott található parancsok - vagy az ún. forró billentyűk - használata a legkézenfekvőbb ezért pár mondatban bemutatnám a legfontosabb menüpontokat.

File menü Itt megtalálhatunk mindent amit a Fájl menükben szokásos elhelyezni. Új projekt létrehozása, létező projekt megnyitása, mentések, nyomtatások, stb. Azonban van egy menüpont amire illik egy kicsit több figyelmet fordítani. Ez a *Make...* parancs. Csak fejlesztési módban választható ki, és hatására megjelenik a *Make Project* párbeszédpanel, ahol beállíthatjuk, hogy hogyan jöjjön létre a az aktuális projektből a végrehajtható .EXE, .DLL vagy .OCX fájl. A párbeszédablakon található egy igen fontos gomb az *Option...*. Ennek segítségével jeleníthető meg a *Project Properties párbeszédablak. Itt állíthatjuk be aktív projektünk fontosabb tulajdonságait.*

Edit menü Szintén egy általános menüpont de itt is találunk speciális parancsokat. Ilyen például a List Properties/Methods menüpont. A menüpont kiválasztásával az éppen aktív kódablakban megjelenik egy lista annak az objektumnak a tulajdonságaival és metódusaival, amelynek a nevét éppen beírtuk, és már kitettük az objektum és tulajdonság/metódus nevét elválasztó pontot (.). Ha nem írtunk be semmit akkor a globális metódusok és eljárások nevei szerepelnek a listában. Ez a funkció azonban automatikussá is tehető az Option párbeszédablakban az Auto List Members kapcsolóval. Ha ez aktív akkor az objektumnév utáni pont (.) begépelésekor automatikusan megjelenik ez a lista. Hasonló könnyítést jeleni a List Constants menüpont is. Ennek aktiválásával az éppen beírt tulajdonság lehetséges értékeit tartalmazó lista jeleníthető meg. Ez a funkció is automatikussá tehető az előbb említett módon. A *Quick Info* parancs lehetőséget ad egy gyors segítség kérésére az aktív ablakban kiválasztott változó, függvény, utasítás, metódus, eljárás használatának helyes szintaxisáról. Automatizálható az *Option* panel *Auto Quick* kapcsolójával. Hasonló segítséget ad a *Parameter Info* parancs is, csak ennek aktiválásával a kiválasztott eljárás, metódus, függvény lehetséges aramétereiről kapunk bővebb információt.

View menü Itt találhatjuk meg azokat a menüpontokat melyek segítségével a már említett és a fejlesztőrendszer részeit képező ablakok jeleníthetők meg. Használata nem igazán igényel bővebb magyarázatot.

Project menü Ez tartalmazza a projekt szempontjából legfontosabb menüpontokat. Itt adhatunk új elemeket (modul, űrlap, osztály, stb.) az aktív projektünkhöz, beilleszthetünk új objektumgyűjteményeket, komponenseket és megváltoztathatjuk a projekt tulajdonságait. Objektumgyűjtemények hozzáadását a *References* menüpont kiválasztásával tehetjük meg. Ennek aktiválásakor megjelenik egy párbeszédablak, ahol a rendszerben regisztrált objektumtárak listájából választva fűzhetünk új objektumokat az aktív projekthez. Új komponensek projektbe helyezését a *Components* menüpont segítségével érhetjük el. A megjelenő párbeszédpanel három fül található. A *Control* lapon a vezérlőelemek, a *Design* lapon a tervezők, az *Insertable* lapon a beilleszthető objektumok listája jelenik meg. A megfelelő jelölőnégyzeteket bekapcsolva

adhatunk elemeket a projekthez. A projektünk fontosabb tulajdonságainak beállítását a *Project Properties* menüpont alatt tehetjük meg. Ennek kitöltése résszint egyértelmű, viszont bizonyos beállítások elvégzéséről még túl korai lenne említést tenni ezért ezt most nem is teszem.

Format menü Az egyes vezérlőelemek űrlapokon való elhelyezését befolyásolhatjuk az itt található menüpontokkal. Mivel ezek alkalmazása leginkább látványhoz köthető, ezért inkább az olvasóra bízom az egyes pontok próbálgatásait. Tegyen fel pár vezérlőt (pl.: Label, TextBox) egy űrlapra és figyelje meg mi történik az egyes menüpontok aktiválásakor. Jó szórakozást J!

Debug és Run menü E két menü használata már leginkább a programozás főbb fázisához a teszteléshez köthetők. A *Run* menüből tudjuk futtani és leállítani alkalmazásunkat, míg a *Debug menü segítségével hibakeresést végezhetünk. Debuggolás során ún. töréspontokat helyezhetünk el a programkódokban (Breakpoint) és szükség szerint soronké*nt is végigmehetünk az alkalmazásunkon. Lehetőség van ún. figyelő ablakok megjelenítésére is ahol egy adott változó, objektum értékét vagy tulajdonságait kísérhatjük figyelemmel.

Tools menü Ez is egy igen hasznos menünek tekinthető, hiszen segítségével igen könnyen tudunk új eljárásokat, függvényeket létrehozni valamint alkalmazásunk számára az egyes űrlapokon menüket is szerkeszthetünk. Itt található még a már említett *Option* menüpont is melynek segítségével a fejlesztőkörnyezet főbb tulajdonságai változtathatók tetszés szerint. Ezekkel a beállításokkal mindenki saját igényeinek megfelelően alakíthatja a VB fejlesztőkörnyezetét. Röviden talán ennyit a VB fejlesztőkörnyezetének bemutatásáról. Sok dolog van ami még említésre kerülhetett volna de mivel a környezet használata igen egyszerű és a VB-vel szállított help igen könnyen értelmezhető, átlátható ezért ezek felfedezését, megismerését az olvasóra bízom.

18.3 Ismertesse a szoftver felületének kialakításához használható vizuális komponensek szerepét, főbb tulajdonságait!

Vizuális komponensek

A komponensek előre készített "építőkockák", melyeket be lehet építeni egy projektbe. Vannak beépített komponensek, amelyeket a Delphi fejlesztői környezettel együtt kapunk meg. De komponenseket akár mi magunk is készíthetünk. Minden egyes komponenstípust valamilyen adott feladat elvégzésére terveztek meg, s egyféle típusból több is felvehető egy projektbe. Ilyen komponenstípus például az ún. Label (=címke, felirat), amely szöveg megjelenítésére alkalmas komponens. Ennek a komponensnek sokféle tulajdonságát tetszőlegesen beállíthatjuk (színe, nagysága, betűtípusa...), továbbá különböző eseményekre is reagálhat (pl. ha egérrel rákattintunk). Tehát egy-egy komponensegyedet, objektumot tulajdonságai, és eseményei jellemzik.

VCL-tulajdonságok

Name: minden komponensnek kötelező nevet adni, melynek egyértelműnek kell lennie. Ezzel a névvel hivatkozhatunk később az objektumra, ezért érvényes Pascal azonosító lehet csak. Ha a komponens neve és felirata azonos, akkor a név módosítása maga után vonja a felirat megváltozását is. Az eseménykezelő eljárások nevei is a Name tulajdonságtól függnek.
Left, Top, Width, Height: a vezérlők pozícióját a Left és Top, méretüket a Width és Height tulajdonságok határozzák meg. A rendszer minden komponensnek tárolja a helyzetét, ezek az adatok a DFM fájlban találhatók meg.
Enabled: ha a komponens Enabled (=Engedélyezett) tulajdonsága igaz értékű, akkor a komponens aktív, amennyiben hamis, akkor a komponens nem aktív, s ez általában a komponens szürkítésével valósul meg.
Visible: a komponens láthatósági tulajdonsága.
Color, Font: színek, betűtípusok. A színeket nevükkel (pl. clBlue) vagy egy négybájtos hexadecimális számmal (pl. $00FF0000) adhatjuk meg.

Lehetőségek kiválasztására alkalmas komponensek

CheckBox: jelölőnégyzet, segítségével úgy állíthatunk be lehetőségeket, hogy azok függetlenek a többi jelölőnégyzet állapotától.
RadioButton: választógomb, amely egymást kizáró beállításokat tesz lehetővé.
GroupBox: a választógombok csoportjainak működését, elhelyezkedését hangolhatjuk össze segítségével.
RadioGroup: olyan csoportmező, melyben választógombok másolatai helyezkednek el. Képes egy vagy több oszlopba rendezni választógombjait.

Listák

ListBox: engedélyezhetjük egyszerre több elem kiválasztását. Fontos tulajdonsága az Items (elem), és az Itemindex (kiválasztott elem sorszáma). A lista első eleme a nulladik.
ComboBox: kombinált lista, a szerkesztőmező és a lenyíló lista tulajdonságait egyesíti. Használhatjuk a Text tulajdonságot a szöveg elérésére.
CheckListBox: minden elem előtt egy jelölőnégyzetet találhatunk.

Gombok

Button: szerepe főleg a kattintással indított tevékenység elvégzésére irányul. Feliratát a Caption tulajdonságban állíthatjuk.
BitBtn: nemcsak feliratot, hanem különböző rajzokat is tartalmazhatnak. A Kind tulajdonsággal beállítható a rendszer által felkínált típus. A Glyph jellemző segítségével egy bittérképes kép adható meg.
SpeedButton: az eszköztárakon megjelenő gombok komponense. Többnyire grafikákat tartalmaznak, lehetnek "beragadt", vagy "felengedett" állapotban.

Menük MainMenu: főmenü, amely az űrlapok tetején látható vízszintes menüsor.
PopupMenu: gyorsmenü, amely az egér jobb gombjának segítségével előhívható függőleges menü

Mi az ablak? A rendszer olyan eleme, mely kerettel, címsorral és általában menüvel ellátott, mozgatható, bezárható, nagyítható. Vannak főablakok és párbeszédablakok. Az ablak egy memóriaterület is egyben, amely a képernyő egy látható eleméhez kapcsolódik, és tartozik hozzá futtatható kód. A rendszer nyilvántartja az ablakokat és viselkedésükkel kölcsönhatásban áll. Minden ablakhoz tartozik egy függvény, az ún. ablakfüggvény, mely feldolgozza az ablak számára lényeges üzeneteket.
Mi a form? A form valójában ablak. Viselkedését főleg a hozzá tartozó kód határozza meg, de megjelenését több tulajdonság is befolyásolja. Megadhatjuk például a form és keretének stílusát.
Ablakstílusok átlapolt ablakok: ezek a klasszikus ablakok
előugró ablakok: párbeszédablakok, üzenetablakok
gyermekablakok: olyan ablakok, melyek nem hagyhatják el a szülőablak területét. A szülőablak egyben a tulajdonos is. A gyermekablak szülője belső területének koordinátáit használja.

18.B. Milyen viselkedési normák betartására van szüksége az ügyfélszolgálati munkatársnak? Ismertesse egy reklamáció kezelésének helyes módját!
- Külső megjelenés. - Udvariasság, türelem.- Tárgyszerűség. - Magas figyelemkoncentráció.
- Reklamáció kezelése:
– az okok feltárása, a felelősség megállapítása, – ügyfél-elégedettség szem előtt tartása,
– kompenzáció, amennyiben az ügyfelet kár érte.

A reklamáció negatív hangulatú, meglepő, ezért érzelmi és védekezési reakciókat vált ki belőlünk. Azonban a reklamáló – az esetek túlnyomó részében – nem azért fordul panaszával hozzánk, hogy megbántson, megalázzon bennünket, hanem mert lelke mélyén abban bízik, ha elmondja nekünk, mivel elégedetlen, mi törekedni fogunk hibánk kijavítására, és akkor ő megbékélhet velünk.

Személyes megfelelés nélkül hozzá sem szabad kezdeni reklamáció kezeléshez. A személyes megfelelés azt jelenti, hogy a panaszkezelő rendelkezik mindazokkal a készségekkel és képességekkel, amelyek - megfelelő felkészültség és intézkedési jogkör biztosítása esetén - alkalmassá teszik a reklamációk szakszerű kezelésére.

Milyen tulajdonságokkal bírjon a jó ügyfélszolgálati munkatárs?
- jó empátiás érzék - jó problémamegoldó készség – türelem - talpraesettség
- rugalmasság – gyorsaság - udvariasság

Mennyi időt töltsünk egy panaszos ügyféllel? A reklamáló ügyfelek egy része csak annyit vár tőlünk, hogy türelmesen, félbeszakítás nélkül hallgassuk végig. Alkalmazkodjunk az ügyfél igényéhez! Ha gyorsan szereti, intézzük gyorsan az ügyét, ha komótos, ne sürgessük. Abból kiindulva, hogy az ügyfélpanasz nem kellemetlenség, hanem értékes segítség az ügyfél részéről, biztosítsuk a megszerzéséhez szükséges időt!

Mit vár tőlünk a reklamáló ügyfél? A legtöbben azt gondolják: „panasza megoldását várja". Végső soron, fő célként valóban ezt akarja, de hogy valóban értékelje a neki nyújtott megoldást, és elégedett legyen, szükség van némi előkészítésre is.
- megértésre, empátiára
- őszinte bocsánatkérésre, vagy legalább együttérzésünk kifejezésére
- megnyugtatásra, hogy ez az eset nem fordulhat elő (vele) még egyszer
- termékcserére, a hiba kijavítására, árengedményre
- kompenzációra
- mérgének levezetésére

A megértés, a bocsánatkérés, a megnyugtatás rögtön a beszélgetés elején, a reklamáció fogadásakor szükségesek, ezek elmaradása a legtöbb esetben nem a megnyugtató és ügyfélmegtartó reklamáció kezelés felé sodorja a beszélgetést.

Őszinte bocsánatkérést, vagy legalább együttérzésünk kifejezését
Tekintet nélkül arra, hogy jogos-e a panasza vagy megalapozatlan! Bocsánatot kérni nehéz, mivel személy szerint szinte soha nem az követte el a hibát, akitől az ügyfél a bocsánatkérést elvárja, ráadásul attól is tarthatunk, hogy „a bocsánatkéréssel elismerjük a hibát, még mielőtt meggyőződtünk volna a tényekről". Ezért használjuk például a „Sajnálom, hogy Önt ilyen kellemetlenség érte...", „Őszintén sajnálom, hogy bosszúsággal kezdődött a napja" mondatkezdeteket. Ez még nem a hiba elismerése, ugyanakkor az ügyfél számára már megfelel.

Megnyugtatást, hogy ez az eset nem fordulhat elő (vele) még egyszer Sajnos, ismétlődő hiba esetén ezt elég nehéz lesz elhitetni, de az ügyfél számára nélkülözhetetlen a bizalom, és hinni akar nekünk. Abban az esetben, ha az ügyfélszolgálat munkatársa tudja, hogy ezt az ígéretet nem tudják betartani, használja a „Törekszünk rá, hogy…" fordulatot.

Termékcserét, a hiba kijavítását, árengedményt Bár a reklamáció alapcélja ez, de mégsem térhetünk rá azonnal, mert a reklamáló nincs rá felkészülve. Amennyiben vita nélkül megkapja, amit akar, viszont a megértést, együttérzést, és megnyugtatást nem, elégedetlen marad és rossz érzésekkel gondol ránk.

Kompenzációt Amennyiben az ügyfélnek kellemetlenséget, kárt okoztunk, nem éri be azzal, amit már eredetileg is meg kellett volna kapnia, hanem további kárenyhítést vár tőlünk – ez a kompenzáció.

Levezetést Gyakran csak annyit akar, hogy levezethesse mérgét. Sajnos, néhányan valóban csak arra használják az ügyfélszolgálatot, hogy valakin büntetlenül levezethessék dühüket. Az ilyen ügyfelek ellen valamennyien tehetetlenek vagyunk.

Reklamáció fogadásának főbb lépései:
- A reklamáló ügyfél megnyugtatása
- A panasz mögött rejtőzködő valódi probléma megértése, azonosítása
- A reklamáció értékelése (jogos vagy megalapozatlan)
- A megoldás, döntéshozás, majd ügyintézés, a szükséges intézkedések megtétele
- A hiba kijavítása
- Lezárás, kommunikáció, a döntés közlése a panaszossal,
- Ügyfél elégedettség ellenőrzése

Jogosnak tekintünk egy reklamációt akkor, ha az ügyféllel kötött szerződésben foglaltaknak nem megfelelően jártunk el. Ez lehet akár egy ráutaló magatartással létrejött szerződés, pl. egy bolti eladás. A vevő vásárol, otthon pedig szomorúan tapasztalja, hogy a készülék nem működik. A szerződés így is létezik, mi viszont nem az abban elvártaknak megfelelően jártunk el, működőképes készüléket árusítottunk, nem működik, nagyon is jogos a reklamáció.

A probléma megismerése és a szükséges információk beszerzése, majd a reklamáció értékelése után megoldást kell találnunk a panaszra, és el is kell intéznünk azt.
Semmiképpen ne feledkezzünk meg arról, hogy a panaszt okozó hibát ki kell javítani!
Az eredmény valamilyen döntés a panaszkezelésben arra jogosultak részéről, amelyet valamilyen intézkedés követ a cégen belül.