Kapitola 8

Ladění programů ve Visual Basicu

Po dokončení této kapitoly budete schopni:

- Rozlišit jednotlivé typy chyb ve svých programech
- Používat nástroje Visual Studia pro nastavení zarážek a pro opravu chyb
- Používat okna *Autos* a *Watch* k prohlížení proměnných v době, kdy program běží
- Používat vizualizéry pro prohlížení řetězcových a komplexních datových typů
- Používat okna *Immediate* a *Command* ke změně hodnot proměnných a ke spouštění příkazů ve Visual Studiu
- Odebírat zarážky

V několika posledních kapitolách jste měli spoustu příležitostí, kdy jste mohli udělat v programovém kódu chyby. Na rozdíl od lidské konverzace, která většinou funguje dobře navzdory občasným gramatickým chybám, je komunikace mezi vývojářem softwaru a kompilátorem jazyka Microsoft Visual Basic úspěšná pouze v případě dodržení přesných pravidel a předpisů programovacího jazyka Visual Basic.

V této kapitole se blíže seznámíte s možnými chybami softwaru, které brání bezproblémovému používání programů vytvořených ve Visual Basicu. Dozvíte se více o různých typech chyb, ke kterým může v programech dojít, a naučíte se používat nástroje Visual Studia, jež slouží k detekci a opravě těchto chyb. Získané znalosti se vám budou hodit při práci s programy v této knize, ale samozřejmě i při psaní dalších programů v budoucnu.

Proč se nyní zaměřujeme právě na ladění? Některé knihy o programování vynechávají toto téma zcela nebo je uvedeno až na konci knihy (poté co jste se seznámili se všemi funkcemi konkrétního produktu). Jistě je v odkládání této látky určitá logika, ale podle mého názoru má smysl, abyste zvládli postupy ladění během výuky programování, a detekce i oprava chyb se tak staly součástí vaší práce při psaní programů a řešení potíží. Dosud jste z knihy získali dostatek informací o objektech, rozhodovacích strukturách a syntaxi příkazů, abyste mohli vytvářet zajímavé programy, při jejichž tvorbě však mohou nastat určité potíže! Jak ale brzy uvidíte, Visual Studio 2010 zjišťování a opravu chyb značně usnadňuje.

Vyhledávání a opravy chyb

Až dosud jste se ve svých programech setkávali pravděpodobně jen s chybami způsobenými překlepy či se syntaktickými chybami. Co když ale v programu dojde k jinému problému – nemůžete chybu najít a opravit jednoduchou kontrolou použitých objektů, vlastností

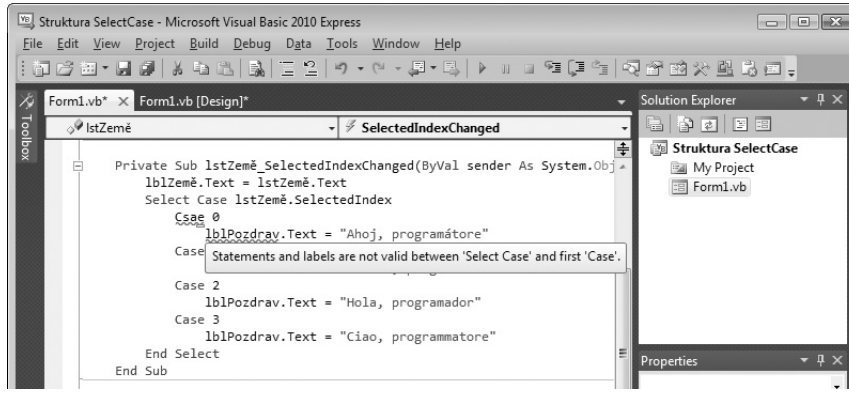
a příkazů? Vývojové prostředí Visual Studia nabízí několik nástrojů, které vám pomohou vyhledávat a opravovat chyby ve vašich programech.

Tři typy chyb

V programu vytvořeném v jazyce Visual Basic se mohou vyskytnout tři typy chyb: syntaktické chyby, chyby vzniklé za běhu programu a logické chyby.

- *Syntaktická chyba* (či *chyba vyvolaná kompilátorem*) je chyba (například nesprávně napsaná vlastnost nebo klíčové slovo), která porušuje programovací pravidla jazyka Visual Basic. Visual Basic označí určité typy syntaktických chyb v programech během zadávání programových příkazů a neumožní vám spustit program, dokud všechny syntaktické chyby neopravíte.
- *Chyba vzniklá za běhu programu* způsobí neočekávané zastavení programu. K tomuto typu chyb dochází v případě, kdy si vnější událost nebo nezjištěná syntaktická chyba vynutí zastavení programu. Pokud například špatně napíšete název souboru při použití metody `System.Drawing.Image.FromFile` nebo se pokusíte číst z disku, v němž není vloženo potřebné CD nebo DVD, váš program vygeneruje chybu.
- *Logická chyba* je chyba způsobená lidským faktorem – jedná se o chybu, která způsobí, že program vrací nesprávné výsledky (nechová se dle očekávání). Velká část úsilí vynaloženého na ladění je obvykle věnována na odhalení logických chyb, které má na svědomí programátor.

Zjistíte-li, že došlo k syntaktické chybě, můžete problém často vyřešit pomocí dokumentace k Visual Studiu, kde získáte informace o chybové zprávě. Chybu pak můžete opravit kontrolou přesné syntaxe použitých funkcí, objektů, metod a vlastností. V okně editoru kódu jsou nesprávné příkazy podtrženy vlnovkou a informace o chybě se zobrazí, když umístíte ukazatel myši nad daný příkaz. Následující obrázek ukazuje chybovou zprávu, která se ve Visual Studiu zobrazí, když klíčové slovo `Case` napíšete nesprávně jako „Csae“ a poté umístíte ukazatel myši nad chybu.



Tip: Ve výchozím nastavení zelená vlnovka označuje varování, červená syntaktickou chybu a fialová nějakou jinou chybu. Tyto barvy a většinu dalších prvků uživatelského rozhraní lze upravit přes příkaz *Options* v nabídce *Tools*, klepnutím na volbu *Fonts And Colors* v položce *Environment* a také úpravou výchozích hodnot v poli *Display Items*.

Pokud dojde k chybě za běhu programu, řešením je často oprava napsaného kódu. Jestliže se například rastrový obrázek nesprávně načítá do objektu obrázkového pole, může problém spočívat v chybně napsané cestě. Mnoho chyb vzniklých za běhu programu však vyžaduje složitější řešení. Do svých programů můžete přidat strukturovanou obsluhu chyby – speciální blok kódu, který rozpozná chybu ve chvíli, kdy k ní dojde, potlačí všechny chybové zprávy a přizpůsobí podmínky programu tak, aby se problém vyřešil. Novou syntaxi strukturované obsluhy chyb si představíme v kapitole 9.

Odhalování logických chyb

Logické chyby v programech se často opravují nejobtížněji. Jsou výsledkem nesprávných úvah a plánování a nespočívají v nepochopení syntaxe jazyka Visual Basic. Prohlédněte si následující rozhodovací příkaz `If...Then`, který vyhodnocuje dva podmíněné výrazy a na základě výsledku zobrazuje jednu ze dvou zpráv.

```
If Věk > 13 And Věk < 19 Then
   TextBox2.Text = "Jste teenager"
Else
   TextBox2.Text = "Nejste teenager"
End If
```

Odhalíte, v čem spočívá problém tohoto příkazu? Teenager je osoba ve věku 13 až 18 let včetně, ale příkaz tak neoznačí osobu, které je přesně 13 let. (Pro tento věk chybně zobrazí zprávu „Nejste teenager“.) Tato chyba není syntaktická (protože příkazy odpovídají pravidlům jazyka Visual Basic), ale jedná se o nesprávnou úvahu, tedy o logickou chybu. Správná rozhodovací struktura bude v prvním porovnání po příkazu `If...Then` obsahovat operátor větší nebo roven (>=):

```
If Věk >= 13 And Věk < 19 then
```

Věřte tomu nebo ne, tento typ chyby je nejčastějším problémem v programech vytvářených ve Visual Basicu. Testování a oprava kódu, který většinou – ne však vždy – zajistí očekávané výsledky, je nejobtížnější.

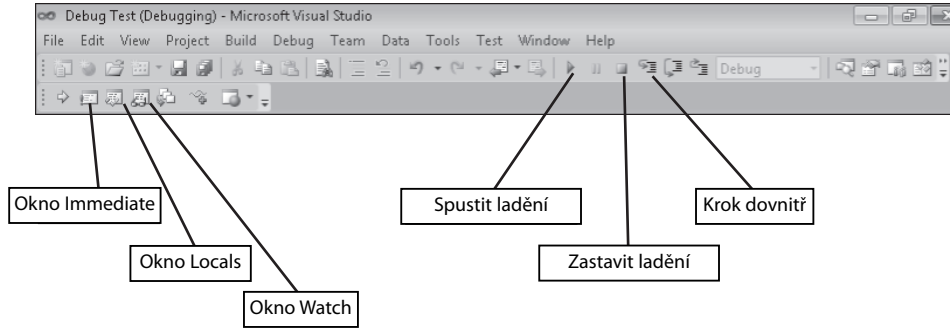
Použití režimu ladění

Jedním ze způsobů zjištění logické chyby je spouštění kódu programu po jednotlivých řádcích a kontrola obsahu jedné či více proměnných nebo vlastností při jejich změnách. Když je program spuštěn, můžete přejít do režimu ladění a prohlížet svůj kód v okně editoru kódu. Režim ladění vám nabízí podrobnější pohled na program, zatímco jej kompilátor Visual Basicu provádí. Je to, jako kdybyste seděli za pilotem a sledovali, jak ovládá letadlo. V případě Visual Basicu se však můžete dotýkat ovládacích prvků.

Při ladění své aplikace budete používat tlačítka na panelech nástrojů *Standard* (standardní) a *Debug* (ladicí), spolu s příkazy v nabídce *Debug* a speciálními tlačítky a okny ve vývojovém prostředí. Následující obrázek ukazuje tlačítka pro ladění na těchto panelech nástrojů. V nabídce *View* přejděte na položku *Toolbars* a vyberte příslušné panely nástrojů. V této kapitole použijete příkazy *Immediate*, *Locals*, *Start Debugging*, *Stop Debugging* a *Step Into*.

V následujícím cvičení nastavíte zarážku, což je místo, v němž se provádění programu přeruší. Poté použijete režim ladění k vyhledání a opravě logické chyby, kterou jste objevili v dříve uvedeném příkazu `If...Then`. (Chyba je součástí programu.) Pro izolaci problému využijete tlačítko *Step Into* na panelu nástrojů *Standard* k postupnému provádění jednotlivých instrukcí v programu a ke kontrole hodnoty klíčových proměnných a vlastností v programu použijete

okno *Autos*. Při této strategii ladění buďte velmi pozorní. Můžete ji použít k nápravě různých typů chyb i ve vlastních programech.



Ladění programu Test ladění

1. Spusťte Visual Studio.
2. V nabídce *File* zvolte příkaz *Open Project*.

   Otevře se dialogové okno *Open Project*.
3. Otevřete projekt Test ladění ze složky *c:\vb10kzk\kap08\Test ladění*.

   Projekt se otevře ve vývojovém prostředí.
4. Nevidíte-li formulář, zobrazte jej.

   Program Test ladění požádá uživatele o zadání věku. Když uživatel klepne na tlačítko Test, program jej informuje, zda se jedná o teenagera nebo ne. V programu však stále existuje problém s 13letými, o kterém jsem se zmiňoval dříve v této kapitole. Nyní otevřete panel nástrojů pro ladění a nastavíte zarážky pro vyhledání chyby.
5. Pokud panel nástrojů pro ladění nevidíte, přejděte v nabídce *View* na položku *Toolbars* a zvolte příkaz *Debug*.

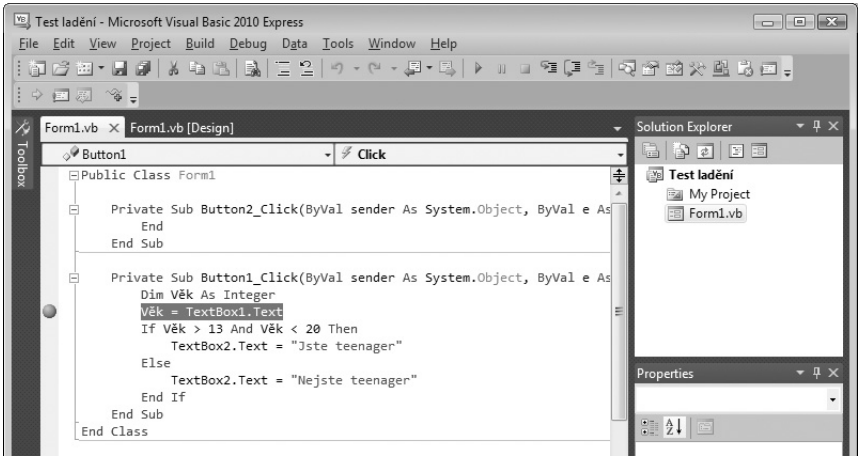
   Panel nástrojů pro ladění se objeví po pravé straně panelu nástrojů *Standard*.
6. Na panelu nástrojů *Standard* klepněte na tlačítko *Start Debugging*.

   Program se spustí a zobrazí formulář *Test ladění*.
7. Z textového pole pro zadání věku odstraňte hodnotu 0, napište číslo **14** a klepněte na tlačítko *Test*.

   Program zobrazí zprávu „Jste teenager“. Zatím je vše v pořádku.
8. Nyní do textového pole pro zadání věku napište číslo **13** a klepněte na tlačítko *Test*.

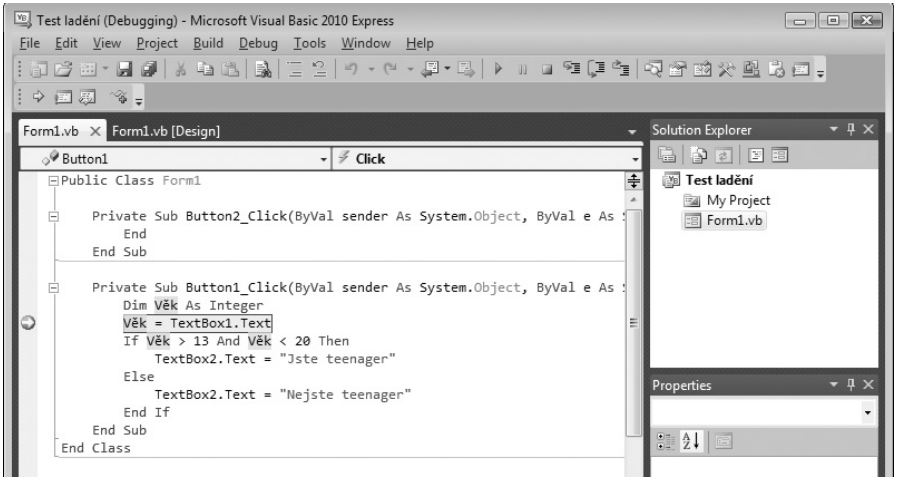
   Program zobrazí zprávu „Nejste teenager“, jak zobrazuje následující obrázek.

   Zobrazená odpověď není správná, a musíte tedy opravit chybu v kódu programu.



9. Klepněte na tlačítko Konec a otevřete okno editoru kódu.
10. Umístěte ukazatel myši na panel *Margin Indicator* (šedý panel u levého okraje okna editoru kódu) vedle příkazu `Věk = TextBox1.Text` v proceduře události `Button1_Click` a poté klepnutím na tento panel nastavte zarážku.

    Zarážka se ihned objeví v červené barvě. Na následujícím obrázku si prohlédněte umístění a tvar zarážky:



11. Klepnutím na tlačítko *Start Debugging* program znovu spusťte.

    Formulář se zobrazí jako předtím a můžete pokračovat se svými testy.
12. Do textového pole pro zadání věku napište číslo **13** a klepněte na tlačítko *Test*.

    Visual Basic otevře znovu okno editoru kódu a zobrazí proceduru události `Button1_Click`, tedy kód programu, který kompilátor právě provádí. Příkaz, který jste vybrali jako zarážku, je označen žlutě a na panelu *Margin Indicator* se objeví šipka, jak ukazuje následující obrázek:

```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
        Dim Věk As Integer
        Věk = TextBox1.Text    Věk 0
        If Věk > 13 And Věk < 20 Then
            TextBox2.Text = "Jste teenager"
        Else
            TextBox2.Text = "Nejste teenager"
        End If
    End Sub
End Class
```

Režim ladění poznáte ve Visual Studiu tak, že je v záhlaví uvedeno slovo „Debugging". V režimu ladění máte možnost sledovat, jak je vyhodnocována logika ve vašem programu.

Poznámka: Režim ladění můžete v programu napsaném ve Visual Basicu spustit i umístěním příkazu `Stop` do kódu programu v místě, kde chcete program přerušit. Jedná se sice o starší, ale spolehlivou metodu pro zahájení režimu ladění v programu, který je napsán v jazyce Visual Basic.

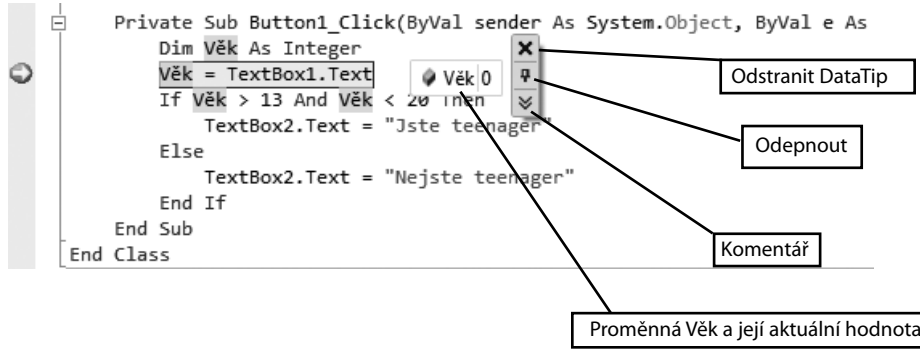
13. V okně editoru kódu umístěte ukazatel myši nad proměnnou `Věk`.

Visual Studio zobrazí zprávu „Věk | 0" a vedle hodnoty se objeví malinká ikona špendlíku. V režimu ladění můžete zobrazit hodnotu proměnných tak, že nad ně v kódu programu prostě umístíte ukazatel myši. Proměnná Věk má aktuálně hodnotu 0, protože zatím nebyla naplněna hodnotou z textového pole `TextBox1` – tento příkaz je dalším příkazem, který bude kompilátor vyhodnocovat.

14. Ikona špendlíku je novým prvkem Visual Studia 2010, který vám umožňuje přišpendlit při ladění hodnotu daného výrazu na nějaké místo v rámci vývojového prostředí. Takto přišpendlený výraz se označuje jako *DataTip*, přičemž k dispozici jsou čtyři příkazy v nabídce *Debug*, které s tímto prvkem souvisejí. Nyní si tento prvek vyzkoušíte pro sledováním hodnoty proměnné `Věk`. Klepnutím na ikonu špendlíku vytvořte ve vývojovém prostředí okno *DataTip* pro proměnnou `Věk`.

15. Držte ukazatel myši nad oknem *DataTip*, dokud se vedle proměnné `Věk` nezobrazí tři malá tlačítka.

Vaše obrazovka by měla vypadat takto:



Dokud toto okno *DataTip* neodstraníte, bude ve vývojovém prostředí zobrazovat hodnotu proměnné `Věk`. Když klepnete na tlačítko *Unpin From Source*, zůstane proměnná `Věk`

na své aktuální pozici v rámci vývojového prostředí, a to i tehdy, pokud posunete okno editoru kódu nahoru nebo dolů. Tlačítko *Comment* umožňuje přidat k proměnné `Věk` popisný komentář a pomocí tlačítka *Close* můžete okno *DataTip* z vývojového prostředí odstranit.

16. Klepnutím na tlačítko *Close* umístěné vedle okna *DataTip* odstraňte prozatím proměnnou `Věk` i s její hodnotou `0`.

Jak vidíte, jedná se o užitečný způsob sledování změn proměnných za běhu programu, takže neváhejte při ladění svého kód používat také okna *DataTip*. Ovšem ještě před tím, než budete používat pouze tato okna, je dobré prozkoumat další techniky, což provedete v následujících krocích.

Poznámka: Pokud do svého programového kódu přidáte více oken *DataTip*, pak nezapomeňte používat příkazy *Clear All DataTips*, *Import DataTips* a *Export DataTips* umístěné v nabídce *Debug*. Tyto prvky jsou zvláště užitečné v rozsáhlých projektech, kdy máte množství proměnných a výrazů a řadu aktivních oken *DataTip*. Konkrétně příkazy *Import* a *Export* vám umožní přenášet okna *DataTip* z jednoho projektu do druhého.

17. Pokračujte klepnutím na tlačítko *Step Into* na panelu nástrojů *Standard*.

Tlačítko *Step Into* provede další příkaz programu v proceduře události (řádek, který je právě označen). Klepnutím na tlačítko *Step Into* zjistíte, jak se stav programu změní po vyhodnocení jednoho příkazu. Když nyní umístíte ukazatel myši nad proměnnou `Věk`, bude obsahovat hodnotu 13.

18. V nabídce *Debug* přejděte na položku *Windows* a klepněte na příkaz *Autos*.

Tip: Pokud používáte Visual Basic 2010 Express, pak nebude okno *Autos* dostupné. Další možností pro sledování hodnoty proměnné `Věk` je otevřít okno *Locals*. Toto okno zobrazuje odlišnou skupinu proměnných.

Podnabídka *Windows* nabízí přístup k celé řadě oken pro ladění. Okno *Autos* zobrazuje stav právě používaných proměnných a vlastností (a to nejen ty, které právě nastavujete, ale také ostatní). Jak vidíte na následujícím obrázku, proměnná `Věk` má hodnotu 13 a vlastnost `TextBox1.Text` obsahuje řetězec `"13"`.

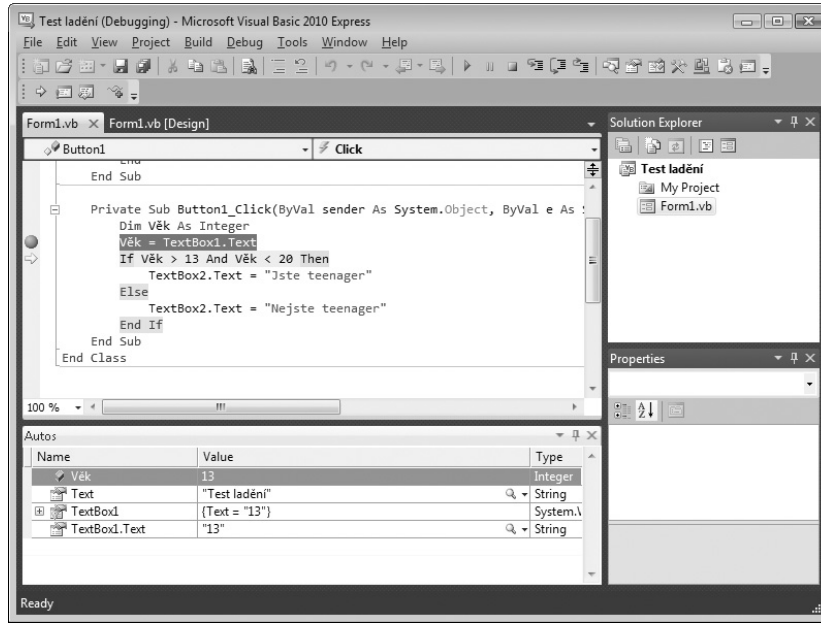
19. Dvakrát za sebou klepněte na tlačítko *Step Into*.

Příkaz `If` vyhodnotí podmíněný výraz na `False` a kompilátor přejde k příkazu `Else`. A zde je naše chyba – logika rozhodovacího příkazu je nesprávná, protože 13letý je teenagerem. Už vidíte, v čem tento problém spočívá? První porovnání vyžaduje operátor větší nebo rovno (`>=`), aby se správně otestovala mez 13 let. Nyní zastavíte ladění, abyste mohli tuto logickou chybu opravit.

20. Na panelu nástrojů *Standard* klepněte na tlačítko *Stop Debugging*.

21. V editoru kódu přidejte operátor = do první podmínky v příkazu `If`, který pak bude vypadat takto:

```
If Věk >= 13 And Věk < 20 Then
```



22. Spusťte program znovu a otestujte své řešení, přičemž se zaměřte zvláště na hraniční hodnoty 12, 13, 19 a 20, které by mohly způsobovat potíže.

Nezapomeňte, že zarážku máte stále nastavenou, takže při dalším spuštění programu opět vstoupíte do ladicího režimu. Použijte tlačítko *Step In* ke sledování toku programu kolem rozhodujícího příkazu `If` a použijte okno *Autos* ke kontrole hodnoty svých proměnných po dokončení testů. Když se formulář zobrazí, zadejte novou hodnotu a vyzkoušejte test znovu. (O odstraňování zarážek se dozvíte v pozdější části této kapitoly.)

23. Po dokončení práce v režimu ladění klepněte na panelu nástrojů na tlačítko *Stop Debugging*, čímž program ukončíte.

Blahopřeji! Úspěšně jste použili režim ladění k vyhledání a opravě logické chyby v programu.

Sledování proměnných v okně Watch

Okno *Autos* je vhodné ke kontrole stavu určitých proměnných a vlastností při jejich vyhodnocování kompilátorem, avšak položky v okně *Autos* setrvávají či udržují své hodnoty pouze pro aktuální příkaz (příkaz označený v okně ladění) a předchozí příkaz (právě provedený příkaz). Pokud váš program pokračuje v provádění kódu, který nepoužívá proměnné, položky z okna *Autos* zmizí.

Pro zobrazení obsahu proměnných a vlastností během provádění programu musíte použít okno *Watch*, což je speciální nástroj Visual Studia, který pro vás sleduje důležité hodnoty, dokud pracujete v režimu ladění. Ve Visual Studiu můžete otevřít až čtyři okna *Watch*, která mají čísla *Watch 1*, *Watch 2*, *Watch 3* a *Watch 4*. Pokud používáte Visual Basic 2010 Express, pak máte k dispozici pouze jedno okno *Watch*. V režimu ladění lze tato okna otevřít tak, že v nabídce *Debug* přejdete na příkaz Windows, poté na položku *Watch* a v podnabídce *Watch* pak klepnete na okno, které chcete otevřít. V okně *Watch* můžete také přidávat výrazy, jako je například `Věk >= 13`.

Otevření okna Watch

Tip: Projekt Test ladění naleznete ve složce *c:\vb10kzk\kap08\Test ladění*.

1. Na panelu nástrojů *Standard* klepněte na tlačítko *Start Debugging* pro opětovné spuštění programu Test ladění.

   Zarážka z předchozího cvičení pro řádek `Věk = TextBox1.Text` by měla být stále nastavená. Není-li tomu tak, ukončete nyní program a nastavte zarážku klepnutím na panel *Margin Indicator* vedle tohoto příkazu (viz krok 10 v předchozím cvičení) a poté spusťte program znovu.

2. Do textového pole pro zadání věku napište číslo **20** a klepněte na tlačítko *Test*.

   Program se zastaví v místě zarážky a Visual Studio spustí režim ladění, v němž lze přidávat proměnné, vlastnosti nebo výrazy do okna *Watch*. Položku můžete přidat, když označíte hodnotu v okně editoru kódu, pravým tlačítkem myši klepnete na výběr a v místní nabídce zvolíte příkaz *Add Watch*.

3. Označte proměnnou `Věk`, klepněte na ni pravým tlačítkem myši a v místní nabídce zvolte příkaz *Add Watch*.

   Visual Studio otevře okno *Watch 1* a přidá do něj proměnnou `Věk`. Hodnota této proměnné je aktuálně `0` a sloupec *Type* v okně označuje proměnnou `Věk` jako typ `Integer`.

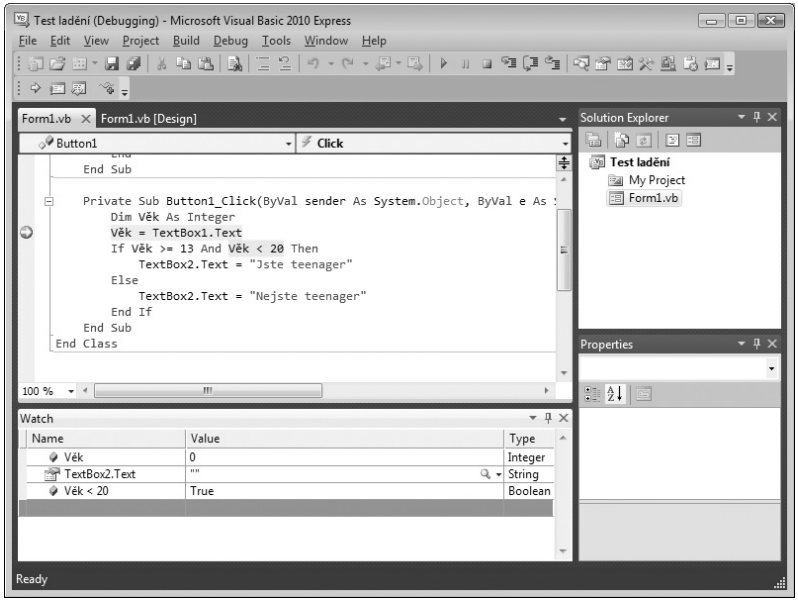
   Druhou možností přidání položky je její přetažení z okna editoru kódu do okna *Watch*.

4. Vyberte vlastnost `TextBox2.Text` a přetáhněte ji do prázdného řádku okna *Watch 1*.

   Když uvolníte tlačítko myši, Visual Studio přidá vlastnost a zobrazí její hodnotu (v tuto chvíli obsahuje prázdný řetězec).

5. Označte výraz `Věk < 20` a přidejte jej do okna *Watch*.

   Výraz `Věk < 19` je podmíněným výrazem a okno *Watch* můžete použít k zobrazení jeho logické či booleovské hodnoty. Okno *Watch* nyní vypadá takto:



Nyní spustíte kód programu, abyste viděli, jak se hodnoty v okně *Watch 1* mění.

6. Na panelu nástrojů pro ladění klepněte na tlačítko *Step Into*.

Tip: Místo klepnutí na tlačítko *Step Into* na panelu nástrojů pro ladění můžete stisknout klávesu F11.

Proměnná `Věk` je nastavena na `20` a podmínka `Věk < 20` se vyhodnotí jako `False`. Tyto hodnoty se v okně *Watch* zobrazí červeným písmem, protože se právě aktualizovaly.

7. Třikrát klepněte na tlačítko *Step Into*.

   V rozhodovacím příkazu je provedena klauzule `Else` a hodnota vlastnosti `TextBox2`. Text v okně *Watch* se změní na `"Nejste teenager"`. Tento test podmínky funguje správně. Protože vám tato podmínka vyhovuje, můžete odebrat test z okna *Watch*.

8. V okně *Watch* klepněte na řádek `Věk < 20` a stiskněte klávesu Delete.

   Visual Studio odstraní hodnotu z okna *Watch*. Jak vidíte, přidávání a odebírání hodnot z okna *Watch* je velmi rychlý proces. Zatím ponechejte Visual Studio v režimu ladění. V další části budete i nadále pracovat v okně *Watch*.

Vizualizéry: Ladicí nástroje pro prohlížení dat

Ačkoli můžete prostřednictvím oken *DataTip*, *Watch*, *Autos* a *Locals* prohlížet ve vývojovém prostředí jednoduché typy dat, jako je `Integer` a `String`, setkáte se ve svých programech i se složitějšími daty. Můžete chtít například zkontrolovat proměnnou nebo vlastnost obsahující strukturované informace z databáze (datové sady) nebo řetězec obsahující kód jazyka HTML či XML z webové stránky. Abyste mohli podrobněji prozkoumat tento typ položky v relaci ladění, Visual Studio nabízí ve vývojovém prostředí sadu nástrojů označovanou jako *vizualizéry*. Ikonou vizualizéru je malá lupa.

Ve vývojovém prostředí Visual Studia 2010 jsou k dispozici čtyři standardní vizualizéry: pro text, HTML, XML (pro práci s řetězcovými objekty) a pro datovou sadu (určenou pro objekty typu `DataSet`, `DataView` a `DataTable`). Společnost Microsoft naznačila, že bude nabízet další typy vizualizérů, které bude možné stáhnout z webu, a navrhla Visual Studio tak, aby vývojáři třetích společností mohli vytvářet vlastní vizualizéry a instalovat je do ladicího prostředí Visual Studia. V následujícím cvičení uvidíte, jak funguje vizualizér pro text. (V tomto cvičení předpokládám, že jste stále v režimu ladění a okno *Watch* je otevřené a obsahuje několik výrazů z programu Test ladění.)

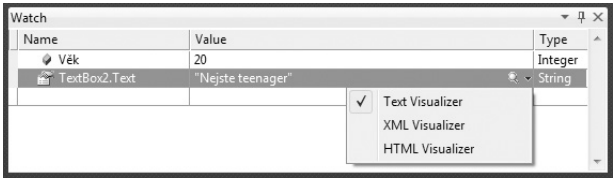
Otevření vizualizéru pro text v ladicím prostředí

1. Na pravé straně okna *Watch* vyhledejte malou ikonu lupy.

   Tato ikona indikuje, že pro danou proměnnou nebo vlastnost prohlíženou v okně *Watch*, *Autos* nebo *Locals* je k dispozici vizualizér. Pokud jste dokončili předchozí cvičení, měli byste lupu vidět u vlastnosti `TextBox2.Text`.

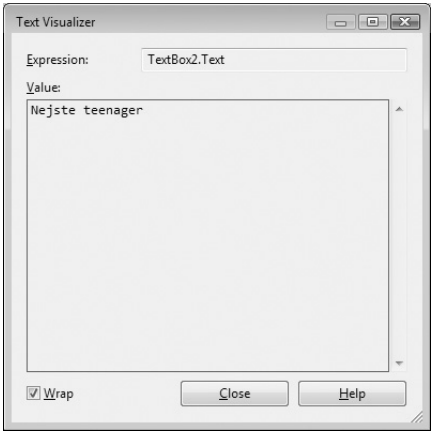
2. Klepněte na šipku u ikony lupy.

   Jestliže se jedná o textovou (řetězcovou) vlastnost, Visual Studio nabízí tři možnosti: jednoduchý vizualizér pro text, který zobrazí vybraný řetězcový výraz jako čitelný text; vizualizér pro HTML, jenž převádí kód HTML na webovou stránku, a vizualizér pro XML, který převádí kód XML na dokument vhodný k prohlížení. Vaše obrazovka bude vypadat jako na následujícím obrázku.



3. Klepněte na možnost *Text Visualizer*.

   Visual Studio otevře dialogové okno a zobrazí obsah vlastnosti `TextBox2.Text`. Vaše obrazovka by měla vypadat přibližně takto:



   Ačkoli tento konkrétní výsledek nabízí jen o málo více než okno *Watch*, výhody vizualizéru jsou na první pohled zřejmé při zobrazení vlastnosti `Text` objektu víceřádkového textového pole nebo v případě, že prohlížíte proměnné či vlastnosti obsahující informace z databází nebo webových dokumentů. S těmito komplexními typy dat budete pracovat dále v této kapitole.

4. Klepnutím na tlačítko *Close* zavřete dialogové okno *Text Visualizer*.

   Ponechejte Visual Studio v režimu ladění. Okno *Watch* použijete i v následující části.

Tip: V režimu ladění se vizualizér někdy zobrazí i v okně editoru kódu vedle zajímavých proměnných či vlastností. Pokud se nějaký vizualizér objeví, klidně na něj podobně jako v předchozím příkladu klepněte – získáte tím více informací o daných datech.

Používání oken Immediate a Command

Až dosud jste používali ladicí nástroje Visual Studia, které nabízely zahájení režimu ladění, provádění kódu po jednom příkazu a prohlížení hodnot důležitých proměnných, vlastností a výrazů z vašeho programu. Nyní se naučíte, jak změnit hodnotu proměnné prostřednictvím okna *Immediate*, a dozvíte se, jak v rámci vývojového prostředí Visual Studia pomocí okna *Command* spouštět příkazy, jako je například *Save All* nebo *Print*. Okna obsahují posuvníky, takže můžete provádět více než jeden příkaz a prohlížet si výsledky pomocí kláves se šipkami.

Následující cvičení ukazuje, jak okna *Immediate* a *Command* fungují. Tato okna probíráme současně, protože se mezi nimi můžete přepínat pomocí těchto speciálních příkazů:

- V okně *Immediate* se příkazem `>cmd` přepnete do okna *Command*.
- V okně *Command* použijete příkaz `immed` pro přepnutí do okna *Immediate*.

Cvičení předpokládá, že je program Test ladění v režimu ladění.

Použití okna Immediate k úpravě proměnné

1. Na panelu nástrojů *Standard* nebo *Debug* klepněte na tlačítko *Immediate*. (Další možností je klepnout na nabídku *Debug*, poté na nabídku *Windows* a nakonec na příkaz *Immediate*.)

   Když vyberete uvedený příkaz, Visual Studio otevře okno *Immediate* a připraví kompilátor na přijetí informací, zatímco je program Test ladění spuštěn. Jedná se o velmi užitečnou funkci, protože můžete rovnou testovat podmínky programu, aniž byste museli program zastavovat a vkládat programové příkazy do okna editoru kódu.

2. V okně *Immediate* napište **Věk = 17** a stiskněte klávesu Enter.

   Právě jste použili okno *Immediate* ke změně hodnoty proměnné. Hodnota proměnné `Věk` se v okně *Watch* okamžitě změní na 17 a při příštím provádění příkazu `If` se hodnota ve vlastnosti `TextBox2.Text` změní na `"Jste teenager"`. Okno *Immediate* nyní vypadá takto:

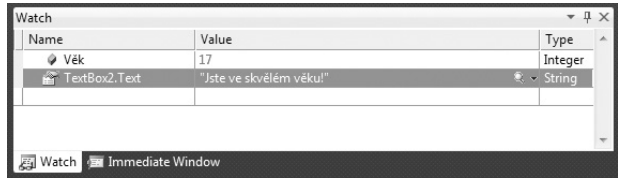
3. Napište do okna *Immediate* následující příkaz a poté stiskněte klávesu Enter:

   ```
   TextBox2.Text = "Jste ve skvělém věku!"
   ```

   Vlastnost `Text` objektu `TextBox2` se ihned změní na text `"Jste ve skvělém věku!"`. V okně *Immediate* můžete změnit hodnotu vlastností i proměnných.

4. Pokud jej nevidíte, zobrazte okno *Watch 1*. (Ve vývojovém prostředí Visual Studia klepněte na záložku *Watch 1*.)

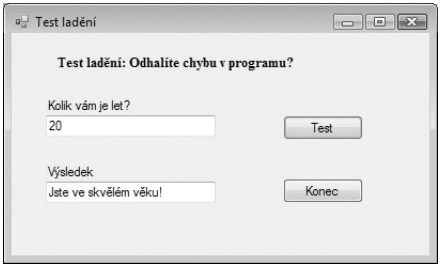
   Okno *Watch* vypadá takto:



   Jak vidíte, obě položky nyní obsahují nové hodnoty a máte možnost dále program testovat.

5. Klepněte dvakrát na tlačítko *Step Into* pro opětovné zobrazení formuláře *Test ladění*.

Všimněte si, že vlastnost `Text` objektu `TextBox2` se změnila, ale vlastnost `Text` objektu `TextBox1` stále obsahuje hodnotu `20` (ne `17`). Důvodem je to, že jste v programu změnili proměnnou `Věk`, ale ne vlastnost, která přiřazuje hodnotu k proměnné `Věk`. Vaše obrazovka vypadá jako na obrázku:



Okno *Immediate* nabízí mnoho možností využití – jedná se o výborné rozšíření k oknu *Watch* a může vám pomoci při zkoušení určitých testovacích případů, které byste jinak velmi složitě vkládali do svých programů.

Přepnutí do okna Command

Textově založené okno *Command* doplňuje okno *Immediate*. Připomíná příkazový řádek v operačním systému Unix nebo MS-DOS a lze jej použít ke spouštění příkazů rozhraní ve vývojovém prostředí Visual Studia. Když například zadáte příkaz `File.SaveAll`, uloží se všechny soubory v aktuálním projektu. (Tento příkaz odpovídá použití příkazu *Save All* v nabídce *File*.) Pokud již máte otevřené okno *Immediate*, můžete se přepínat mezi okny *Immediate* a *Command* zadáním příkazů `>cmd` nebo `immed`. Pro otevření okna *Command* můžete také klepnout na nabídku, otevřít podnabídku *Other Windows* a poté klepnout na příkaz *Command Window*. V následujícím cvičení si vyzkoušíte použití okna *Command*.

Tip: Visual Basic 2010 Express okno *Command* neobsahuje. (Pokud používáte verzi Express, pak nebudete moci toto cvičení dokončit.)

Spuštění příkazu File.SaveAll

1. Klepnutím na záložku *Immediate Window* zobrazte okno *Immediate*.
2. Napište `>cmd` a stiskněte klávesu Enter pro přepnutí do okna *Command*.

   Otevře se okno *Command* a okna *Immediate* nebo *Watch* mohou být nyní částečně (nebo zcela) skrytá. (Do okna *Immediate* se můžete vrátit klepnutím na jeho záložku nebo použitím příkazu `immed` v okně *Command*.) Zobrazí se výzva `>`, která naznačuje, že nyní pracujete v okně *Command*.
3. V okně *Command* napište **File.SaveAll** a stiskněte klávesu Enter.

   Jakmile začnete psát slovo **File**, zobrazí se v rozbalovacím seznamu všechny příkazy související s nabídkou *File* a s operacemi se soubory. Tato funkce Microsoft IntelliSense poskytuje možnost seznámit se s mnoha příkazy, které lze provádět v rámci okna *Command*. Když dopíšete příkaz `File.SaveAll` a stisknete klávesu Enter, Visual Studio uloží aktuální projekt a vrátíte se do příkazového řádku, jak ukazuje tento obrázek:

4. Chcete-li, vyzkoušejte si i další příkazy. (Při psaní příkazů začínejte názvy nabídek, abyste si mohli prohlédnout různé příkazy, které jsou k dispozici.) Po skončení práce klepněte na tlačítko Zavřít v oknech *Command* a *Immediate*.

O krok dál: Odebírání zarážek

Pokud jste se pečlivě drželi pokynů v této kapitole, je program Test ladění stále spuštěný a obsahuje zarážku. Pro její odebrání a ukončení programu proveďte následující kroky. Ladění programu Test ladění je nyní hotové.

Tip: Visual Basic 2010 Express níže zmíněný příkaz *Delete All Breakpoints* neobsahuje, a proto musíte zarážky vymazat postupně, jednu po druhé.

Odebrání zarážky

1. V okně editoru kódu klepněte na červené kolečko spojené se zarážkou na panelu *Margin Indicator*.

   Zarážka zmizí. A to je vše, co musíte udělat! Měli byste však vědět, že pokud máte v programu více než jednu zarážku, můžete je všechny odebrat zvolením příkazu *Delete All Breakpoints* v nabídce *Debug*. Visual Studio ukládá zarážky společně s vaším projektem, takže je důležité vědět, jakým způsobem je můžete odebrat, jinak totiž zůstanou ve vašem projektu stále, i když zavřete a restartujete Visual Studio!

2. Na panelu nástrojů *Standard* klepněte na tlačítko *Stop Debugging*.

   Program Test ladění se ukončí.

3. V nabídce *View* zvolte příkaz *Toolbars* a klepněte na položku *Debug*.

   Panel nástrojů *Debug* se uzavře.

Naučili jste se základní postupy při ladění programů ve Visual Basicu v rámci vývojového prostředí Visual Studia. Dejte si k této kapitole záložku, abyste se k ní mohli vrátit, pokud byste dále v této knize narazili na nějaké potíže. V další kapitole se dozvíte, jak zpracovávat chyby vzniklé za běhu programu prostřednictvím strukturované obsluhy chyby.

Rychlý přehled kapitoly 8

Pro	učiňte následující
Zobrazení panelu nástrojů pro ladění	V nabídce *View* přejděte na položku *Toolbars* a zvolte příkaz *Debug*.
Nastavení zarážky	V okně editoru kódu klepněte na panel *Margin Indicator* vedle příkazu, u kterého chcete přerušit provádění programu. Když kompilátor dojde k zarážce, zahájí režim ladění. Nebo uveďte v programu příkaz Stop v místě, kde chcete zahájit režim ladění.
Provedení jednoho řádku kódu v okně editoru kódu	Na panelu nástrojů *Standard* klepněte na tlačítko *Step Into*.
Prozkoumání proměnné, vlastnosti nebo výrazu v okně editoru kódu	V režimu ladění vyberte hodnotu v okně editoru kódu a umístěte nad ni ukazatel myši.
Použití okna *Autos* k prozkoumání proměnné v aktuálním a předchozím řádku	V režimu ladění přejděte v nabídce *Debug* na položku *Windows* a klepněte na příkaz *Autos*.
Přidání proměnné, vlastnosti nebo výrazu do okna *Watch*	V režimu ladění označte hodnotu v okně editoru kódu, klepněte na ni pravým tlačítkem myši a v místní nabídce zvolte příkaz *Add Watch*.
Zobrazení okna *Watch*	V režimu ladění přejděte v nabídce *Debug* na možnost *Windows*, poté na položku *Watch* a klepněte na jedno z dostupných oken.
Zobrazení informací o HTML, XML nebo datové sadě během relace ladění	V okně *Autos*, *Watch*, *Locals* nebo *DataTip* klepněte během relace ladění na ikonu vizualizéru (malá lupa).
Otevření okna *Immediate*	V nabídce *Debug* přejděte na položku *Windows* a zvolte příkaz *Immediate*.
Spuštění příkazu ve vývojovém prostředí Visual Studia z okna *Command*	V příkazovém řádku napište za `>` název příkazu a stiskněte klávesu Enter. Například pro uložení aktuálního projektu napište *File.SaveAll* a stiskněte klávesu Enter.
Přepnutí do okna *Command* z okna *Immediate*	Napište příkaz `>cmd` a stiskněte klávesu Enter. Pro zpětné přepnutí do okna *Immediate* napište příkaz `immed` a stiskněte klávesu Enter.
Odebrání jedné či více zarážek	Na panelu *Margin Indicator* v okně editoru kódu klepněte na zarážku nebo v nabídce *Debug* zvolte příkaz *Delete All Breakpoints*.
Ukončení ladění	Na panelu nástrojů *Standard* klepněte na tlačítko *Stop Debugging*.