Zpravodaj moderníProgramování 09/2012: Nulovatelné typy

Obtížnost: začátečníci

Pokud chcete uložit číslo, vytvoříte si proměnnou typu `int`. Jak to však udělat, když tam to číslo někdy nemusí být? U odkazových typů je to jednoduché, prostě se přiřadí hodnota `null`. Jak to však udělat u čísel?

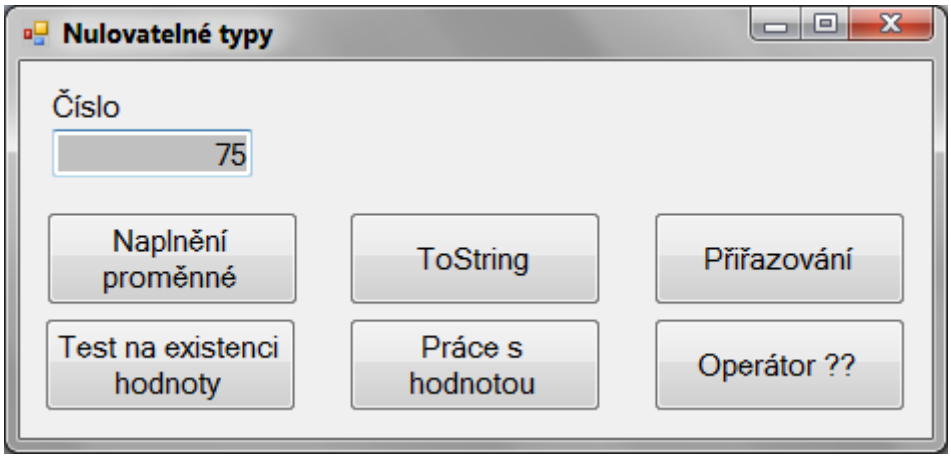
Jednou z možností jak signalizovat, že v číselné proměnné žádná smysluplná hodnota není, je přiřadit do ní nějakou hodnotu, která by se tam normálně neměla vyskytnout - nulu, minus jedničku, `int.MinValue` ap. Tímto způsobem například funguje `SelectedIndex` u listboxu. Normálně vrací pořadové číslo (počínaje nulou) vybraného prvku. Pokud však žádný prvek vybrán není, vrací -1, která se jinak nemůže vyskytnout.

Toto řešení však mnohdy může být problematické. Nemusí být vždy jasné, která hodnota má být ta „nesmyslná“. Často se například v řešeném problému mohou vyskytnout záporná čísla i tam, kde je zprvu vůbec nečekáme ap.

Nulovatelné typy

Lepší řešení nabízejí tzv. ***nulovatelné typy***. Pokud místo typu `int` použijeme nulovatelný `int`, budeme moci do něj uložit jednak všechna čísla, jednak hodnotu `null`. Ukažme si vše prakticky.

Vytvořte si uživatelské rozhraní programu podle obrázku.



Dále deklarujte členské proměnné okna:

```
int? nulovatelnéČíslo;
Nullable<int> jinéNulovatelnéČíslo;
```

Typ `Nullable<int>` znamená „nulovatelný `int`“. Totéž lze alternativně (a častěji) zapsat jako `int` s otazníkem.

Naplnění proměnné

Budeme pracovat především s proměnnou `nulovatelnéČíslo`. Jak jsme si řekli, do typu `int?` je možno uložit jednak číslo, jednak hodnotu `null`. Provedeme tedy konverzi hodnoty zadané uživatelem. Když se povede (když uživatel zadá korektní celé číslo), uložíme zadané číslo do naší proměnné. Když ne, uložíme `null`:

```
private void tlačítkoNaplněníProměnné_Click(object sender, EventArgs e)
{
    try
    {
        nulovatelnéČíslo = Convert.ToInt32(poleČíslo.Text);
    }
    catch
    {
        nulovatelnéČíslo = null;
    }
}
```

ToString

Práce s proměnnými nulovatelných typů je komplikovanější než práce s běžnými proměnnými. Nejprve si však ukážeme něco, co je úplně stejné. Hodnotu nulovatelného typu lze bez problémů převést na textový řetězec. Pokud je hodnota `null`, převede se na prázdný řetězec.

```
private void tlačítkoToString_Click(object sender, EventArgs e)
{
    MessageBox.Show(nulovatelnéČíslo.ToString());
}
```

Přiřazování

Hodnotu proměnné `int?` lze bez problémů přiřadit do jiné proměnné `int?`:

```
jinéNulovatelnéČíslo = nulovatelnéČíslo;
```

Nemůžeme ji však přiřadit do proměnné `int`, do obyčejného čísla:

```
int číslo = nulovatelnéČíslo; // nelze
```

Pokud jsme si jisti, že v nulovatelné proměnné není `null`, můžeme přiřazení provést s pomocí vlastnosti `Value`:

```
int číslo = nulovatelnéČíslo.Value;
```

Tento příkaz se nám sestaví, zdůrazňuji však ještě jednou podmínku „pokud jsme si jisti“. V případě, že by v proměnné `nulovatelnéČíslo` byla hodnota `null`, došlo by k běhové chybě.

Celá obsluha třetího tlačítka vypadá následovně:

```
private void tlačítkoPřiřazeníDoInt_Click(object sender, EventArgs e)
{
    jinéNulovatelnéČíslo = nulovatelnéČíslo;
    int číslo = nulovatelnéČíslo.Value;
    MessageBox.Show("Přiřazení OK");
}
```

Test na existenci hodnoty

Jak si můžeme být jisti, že v nulovatelné proměnné není `null`? Buď to víme z principu našeho řešení, nebo můžeme udělat test. Jsou dvě ekvivalentní alternativy:

- Porovnání s hodnotou `null`;
- Dotaz na vlastnost `HasValue`.

```
private void tlačítkoTest_Click(object sender, EventArgs e)
{
    MessageBox.Show(
        nulovatelnéČíslo.HasValue.ToString() +
        Environment.NewLine +
        (nulovatelnéČíslo != null).ToString());

}
```

Práce s hodnotou

Předchozí tlačítko nám demonstruje ekvivalentnost obou testů, nyní jeden z nich zapojíme do podmínky:

```
private void tlačítkoPráceSHodnotou_Click(object sender, EventArgs e)
{
    if (nulovatelnéČíslo != null)
    {
        int o1víc = nulovatelnéČíslo.Value + 1;
        MessageBox.Show("O jedničku větší je " + o1víc.ToString());
    }
    else
        MessageBox.Show("V proměnné není hodnota");
}
```

Operátor ??

Ve speciální situaci, kdy ve výpočtu chceme případnou hodnotu `null` nahradit nějakou implicitní hodnotou, s výhodou využijeme operátor `??`:

```
private void tlačítkoOperátorOtazníky_Click(object sender, EventArgs e)
{
   int číslo = nulovatelnéČíslo ?? 9999;
   MessageBox.Show(číslo.ToString());
}
```

Pokud `nulovatelnéČíslo` je `null`, dosadí se 9999.

Závěr

Nulovatelné typy jsou šikovná věc. Je však třeba umět s nimi pracovat, v čem by vám mělo pomoci právě toto číslo Zpravodaje.

Radek Vystavěl, 8. listopadu 2012

Pokud Vám Zpravodaje moderníProgramování připadají užitečné, doporučte jejich odběr svým známým. Mohou se přihlásit na webu www.moderniProgramovani.cz.