Zpracování výjimek v .NET

Výjimka (Exception) je objekt, který nese informaci o chybě, k níž došlo během vykonávání programu. Proces zachycení a zpracování této chyby se nazývá **Exception Handling**.

1. Základní konstrukce: try-catch-finally

Pro práci s výjimkami používá .NET čtyři základní klíčová slova:

- **try:** Blok kódu, ve kterém očekáváme, že by mohlo dojít k chybě.
- **catch:** Blok, který se vykoná pouze v případě, že v sekci `try` nastane výjimka. Zde se provádí logování nebo náprava chyby.
- **finally:** Blok, který se vykoná **vždy**, bez ohledu na to, zda výjimka nastala či nikoliv. Ideální pro uvolňování zdrojů (pokud nepoužíváme blok using).
- **throw:** Slouží k ručnímu vyvolání výjimky.

```
try {
    int vysledek = 10 / nula; // Zde vznikne DivideByZeroException
}
catch (DivideByZeroException ex) {
    Console.WriteLine("Nelze dělit nulou!");
}
catch (Exception ex) {
    Console.WriteLine($"Nastala obecná chyba: {ex.Message}");
}
finally {
    Console.WriteLine("Operace ukončena.");
}
```

---

2. Hierarchie výjimek

V .NET všechny výjimky dědí ze základní třídy **System.Exception**. Při odchytávání je důležité řadit bloky `catch` od **nejspecifičtějších po nejobecnější**.

- **SystemException:** Chyby generované runtimem (např. `NullReferenceException`, `StackOverflowException`).
- **ApplicationException:** Dříve doporučovaný základ pro vlastní výjimky (dnes se doporučuje dědit přímo z `Exception`).

---

3. Propagace výjimek a Stack Trace

Pokud výjimka není zachycena v aktuální metodě, „probublává“ nahoru do volající metody. Tento

proces pokračuje, dokud není nalezena odpovídající sekce `catch`, nebo dokud aplikace nespadne.

Objekt výjimky obsahuje vlastnost **StackTrace**, což je textový výpis cesty, kterou výjimka prošla. To je klíčové pro debugging.

> **Pozor:** Při opětovném vyhazování výjimky v bloku catch používejte samotné `throw;`. Pokud použijete `throw ex;`, přepíšete původní StackTrace a ztratíte informaci o tom, kde chyba skutečně vznikla.

---

4. Filtry výjimek (Exception Filters)

Od verze C# 6.0 lze použít klíčové slovo `when`, které umožní zachytit výjimku jen za určité podmínky:

```
catch (HttpRequestException ex) when (ex.StatusCode ==
HttpStatusCode.NotFound) {
    Console.WriteLine("Stránka nebyla nalezena.");
}
```

---

5. Best Practices

- **Nechytejte všechno:** Nepoužívejte prázdný `catch (Exception) {}` (tzv. „swallowing exceptions“). Chyby by neměly být ignorovány.
- **Vlastní výjimky:** Vytvářejte si vlastní třídy výjimek pro specifické doménové chyby (např. `InsufficientFundsException`).
- **Předcházejte výjimkám:** Pokud můžete chybě předejít podmínkou (např. `if (obj != null)`), je to výkonově mnohem lepší než nechat vyvolat `NullReferenceException`.

Související články:

- Rozhraní IDisposable a blok using
- Unit Testing (testování chybových stavů)
- Garbage Collector a správa paměti

Tagy: programming dot-net csharp exception-handling error-management clean-code