

Goal

- Code
 - reference types
 - array
 - var
 - foreach

Diffuculty

- Intermediate
- 2-3x45 min

Reference vs value

S příchodem struktur vznikl nový problém s místem v operační paměti. Zatímco třeba `bool` nebo `int` zabírají minimum místa, `struct` žádné omezení nemá. V jedné struktuře můžou být tisíce dalších struktur, a v nich tisíce dalších property. Jedna proměnná takové mega-struktury může být klidně 0.5 GB. A představme si následující kód.

```
MegaStruct myMegaVariable = /* initialize */  
MegaStruct copyForJohn = myMegaVariable;  
MegaStruct copyForArchive = myMegaVariable;  
MegaStruct copyInCaseOfRestore = myMegaVariable;  
  
// do stuff with myMegaVariable
```



Tyto čtyři řádky najednou sežraly 2GB operační paměti (RAM). A neopatrný programátor se špatně navrženým cyklem mohl shodit i největší servery. Z toho důvodu (a dalších samozřejmě), vznikl nový koncept a to tzv. předávání odkazem (reference/pointer).

Reference

Si můžete představit jako ten odkaz na internetu. Než posílat kamarádovi velké video, pošlete mu krátký odkaz na youtube. Když bude chtít, video najde na odkazu.

Nebo jako poštovní adresu. Než stěhovat celý dům, pošlete jen jeho adresu. Kdo bude chtít, najde dům na adrese.

V programování můžete do proměnné uložit odkaz na místo v paměti, kde se nachází jiná proměnná.

Example - value type

Nejprve ukázka s klasickým datovým typem předávaným hodnotou. S těmi jsme pracovali do teď. Použijme `struct`.

```
public struct CoordsStruct
{
    public int X;
    public int Y;
}
```

Jednoduchý datový typ pro reprezentaci bodu v 2D prostoru. Známe (pokud ne, projděte si prerekvizity tohoto cvičení). Následující kód vytvoří v paměti novou proměnnou a nastaví hodnoty.

```
CoordsStruct firstPointStruct;
firstPointStruct.X = 10;
firstPointStruct.Y = 20;
```

Následující kód vytvoří v paměti *jinou* proměnnou a nastaví jí hodnoty podle první proměnné.

```
CoordsStruct secondPointStruct;
secondPointStruct = firstPointStruct;
```

Paměť si pak můžete představit třeba takto:

firstPointStruct	secondPointStruct
10	10
20	20

Zkuste odhadnout, jaký bude výpis na obrazovku, když provedeme následující kód.

```
firstPointStruct.X = 999;
Console.WriteLine(firstPointStruct.X);
Console.WriteLine(secondPointStruct.X);
```

Správně. Vypíše se `999` a poté `10`. Paměť totiž vypadá následovně.

firstPointStruct	secondPointStruct
999	10
20	20

Example - reference type

Protože struktury byly oblíbené, přišel C# s novým typem `record`, který funguje úplně stejně, jen je předáván odkazem.

```
public record CoordsRecord
{
    public int X;
    public int Y;
}
```

V deklaraci se změnilo `struct` na `record`. Jinak to samé. Pokud chci ale vytvořit novou proměnnou, musím použít klíčové slovo `new`. To je hlavní rozdíl.

```
CoordsRecord firstPointRecord;
firstPointRecord = new CoordsRecord()
{
    X = 10,
    Y = 20
};
```

`new` programu říká běž a vytvoř v paměti novou proměnnou tohoto typu. Pro druhou proměnnou nepoužijeme `new`, takže do druhé proměnné se v tu chvíli uloží **odkaz na první bod**.

```
CoordsRecord secondPointRecord;
secondPointRecord = firstPointRecord;
```

Operační paměť si můžete představit takto:

firstPointRecord	secondPointRecord
10	<-- odkaz/reference/pointer
20	

Když teď zavoláme následující kód

```
firstPointRecord.X = 999;
Console.WriteLine(firstPointRecord.X);
Console.WriteLine(secondPointRecord.X);
```

Vypíše se dvakrát `999`, protože druhá proměnná jen odkazuje na první, kterou jsme změnili. V obou výpisech totiž čerpáme ze stejného místa v paměti.

firstPointRecord	secondPointRecord
999	<-- odkaz/reference/pointer
20	

Explained by GIF

 <https://blog.penjee.com/wp-content/uploads/2015/02/pass-by-reference-vs-pass-by-value-animation.gif>

Explained by a professional

<https://youtu.be/mvieNUe9Urs?si=XDkopWWdL-7al7-1>

Important

Není nutné si pamatovat, které typy jsou hodnotou a které odkazem. Zaprvé se to v čase mění a zadruhé je datových typů šíleně moc. Jo a každý jazyk to má trochu jinak. Důležité je vědět, jaký je v tom rozdíl. Až budete chtít použít datový typ, kde si nejste jistí, stačí pár vteřin googlení.

`new` je klíčové slovo, které říká vytvoř v paměti proměnnou daného typu.

`null` je defaultní hodnota typů předávaných odkazem. Prázdný odkaz. Něco jako `void` u metod.

Pros and cons

Výhodou tohoto přístupu je, že mnoho procesů může sdílet stejná data. (Ušetřená paměť)

Nevýhodou tohoto přístupu je, že mnoho procesů může sdílet stejná data. (Více kandidátů na editaci, větší zmatek, větší potřeba dávat bacha. Podrobněji ve cvičení.)

Instructions

Naklonujte si projekt a postupně vyřešte všechna zadání.