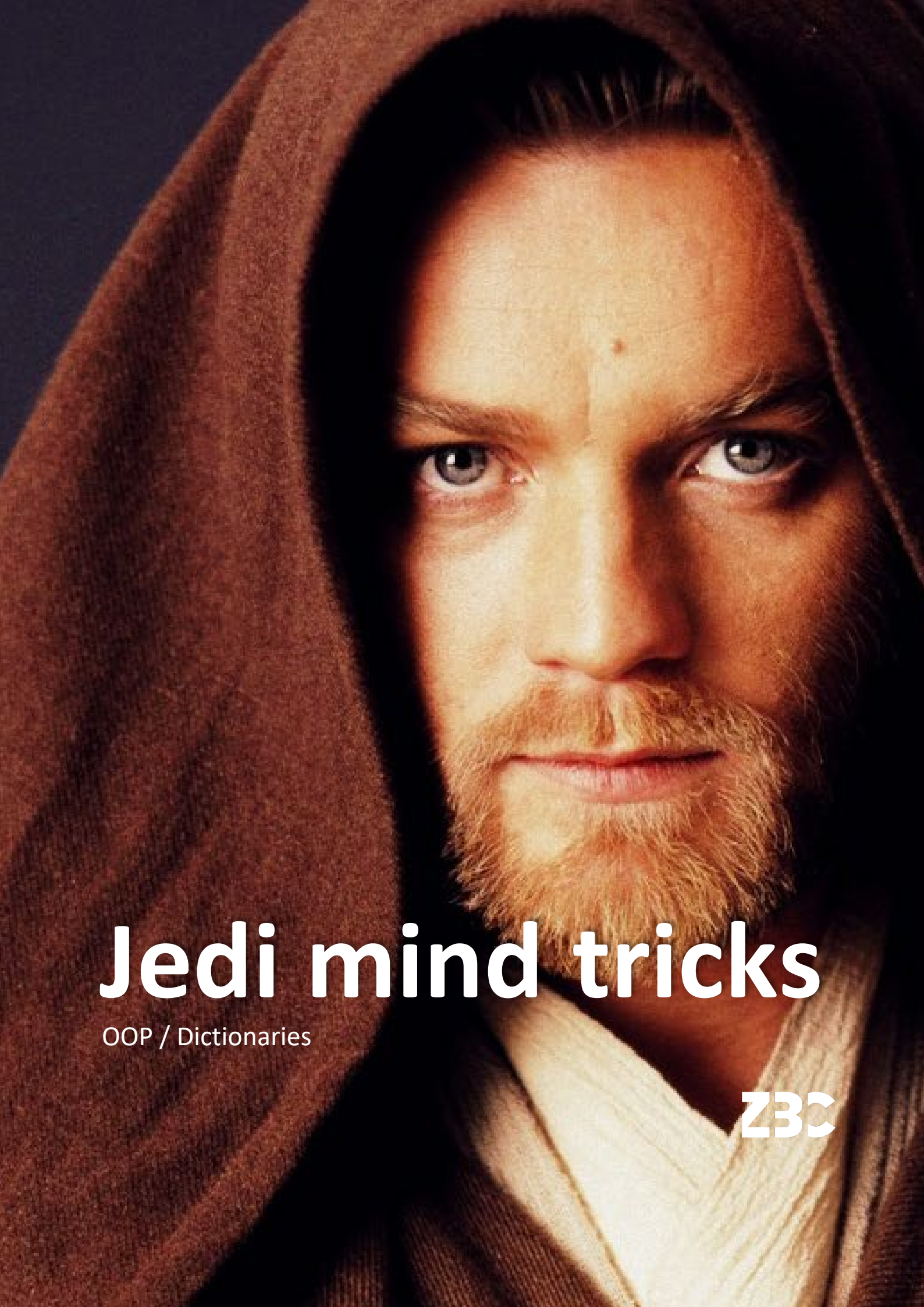


Jedi mind tricks

OOP / Dictionaries

ZBC

Jedi mind tricks (C#)

Et dictionary er en populær samlingstype/collection, og den er lidt anderledes end en liste. Et dictionary giver dig mulighed for at angive en nøgletype samt en værditype.

```
using System;
using System.Collections.Generic;

public class Program
{
    public static void Main()
    {
        Dictionary<string, bool> people = new Dictionary<string, bool>();
        people.Add("Luke", true);
        people.Add("Leia", false);
        people.Add("Han", false);
        people.Add("Yoday", true);
        people.Add("Chewbacca", false);

        foreach (KeyValuePair<string, bool> pair in people)
        {
            // A KeyValuePair provides the string as the Key property
            // and the bool as the Value property.

            if (pair.Value)
            {
                Console.WriteLine(pair.Key + " is a Jedi");
            }
            else
            {
                Console.WriteLine(pair.Key + " is not a Jedi");
            }
        }
    }
}
```

Resultant:

```
Luke is a Jedi
Leia is not a Jedi
Han is not a Jedi
Yoday is a Jedi
Chewbacca is not a Jedi
```

Et dictionary er en datastruktur, som giver dig mulighed for at tilføje et element og henvise/referer til elementet via en nøgle.

Når du erklærer et dictionary, skal du give 2 generiske typer som argumenter:

```
Dictionary<string, bool> people = new Dictionary<string, bool>();
```

Dette ligner List <T>, men dictionary generiske type er Dictionary <TKey, TValue>.

Du kan bruge en hvilken som helst type for begge typer argumenter, f.eks.

```
    Dictionary<Person, DateTime> birthdays = new Dictionary<Person, DateTime>();
    Dictionary<Point, string> locations = new Dictionary<Point, string>();
```

Du kan også benytte var ☹ ligesom i en enhver anden erklæring:

```
var droids = new Dictionary<string, Droid>();
```

Du kan også initialisere et dictionary og tilføje værdier:

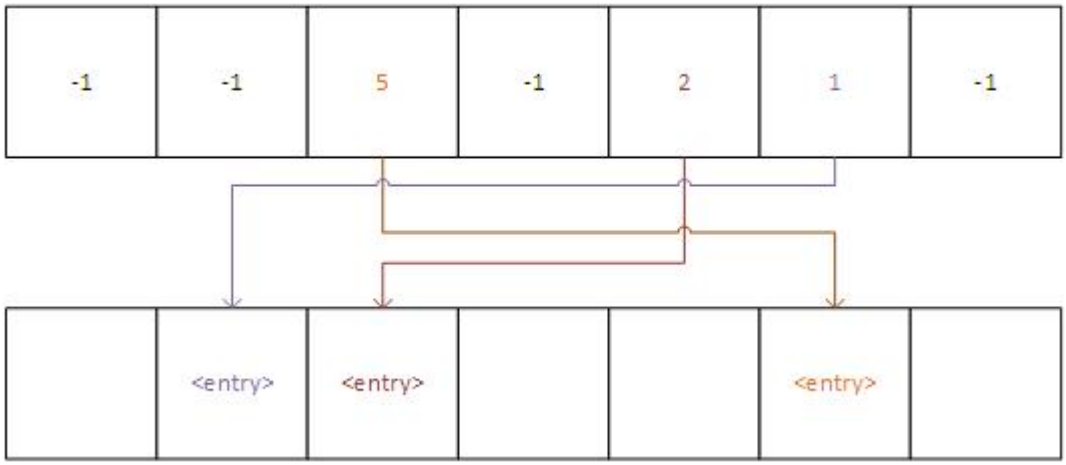
```
Dictionary<string, int> ages = new Dictionary<string, int> { { "Luke", 53 }, { "Leia", 53 } };
```

Jedi Mind Tricks eller simpel logik

I modsætning til lister benytter dictionary ikke et array som ”backing source”. Et dictionary bruger en struktur kendt som en hashtable.

Hvis der skal lageres/hentes en værdi i en hashtable, genereres et nummer kendt som en hash-kode for den nøgle, du angiver. Hashkode bruges som indekset i et internt array af hashtable, som faktisk lagrer værdien.

Ideen bag dette koncept er, at når du har fået en hashkode, er det langt enklere at finde den tilhørende værdi, i stedet for at gennemløb af hele tabelen for at for at identificere nøglen manuelt.



I .NET Dictionary, er hashkoen indekset for en "bucket". Bucket array er en forud tildelt størrelse, og kan vokse, når der tilføjes elementer, men kun til max kapacitet. Bucket gemmer indekset for den faktiske værdi i et "entries" array.

Adgang til listeværdier.

For at få adgang til en værdi fra listen benyttes samme "indeks" notation som i array, men i stedet for at bruge et nummer bruger du værdi af samme type som argumentet "Key" type:

```
var droids = new Dictionary<string, Droid>() { { "R2 D2", new Droid() } };
Droid r2d2 = droids["R2 D2"];
```

Hvis du angiver en nøgle, der ikke findes, kastes en KeyNotFoundException.

Tilføj dictionary værdier

For at tilføje et element til dictionary er der to muligheder, hvoraf den første benytter Add-metoden:

```
droids.Add("R2 D2", new Droid());
```

Add-metoden advarsel

Hvis du tilføjer et element med en nøgle, der allerede findes i dictionary, kastes en ArgumentException, der angiver, at nøglen allerede findes i dictionary. Alternativt kan du bruge indeksnotationen til at indstille værdien:

```
droids["R2 D2"] = new Droid();
```

Denne metode giver dig mulighed for at erstatte eksisterende elementer uden at en exception kastes.

Fjern værdier

For at fjerne et emne fra en dictionary benyt Remove metoden:

```
droids.Remove("R2 D2");
```

Igen kan du ved at angive nøglen fjerne værdien. Remove metoden returnerer en værdi (boolsk), der angiver, om elementet blev fjernet:

```
if (droids.Remove("R2 D2"))
{
  Console.WriteLine("Removed R2 D2")
}
else
{
   Console.WriteLine("Couldn't find R2 D2")
}
```

Iteration gennem dictionary

Ligesom lister understøtter dictionarys enumeration. Den store forskel mellem dem er, at hvor en List returnerer en enumeration af typen T, vil dictionary <TKey, TValue> typen returnere en enumeration af typen KeyValuePair <TKey, TValue>, som er en tuple bestående af nøglen og værdien for hvert element . Så ved at bruge en for-each løkke kan vi bevæge os gennem arrayet:

```
foreach (var pair in droids)
{
    Console.WriteLine("Found: " + pair.Value + ", with key: " + pair.Key);
}
```

Opgave 1 – erklær et dictionary

Opret et nyt dictionary, hvor nøgletypen er en streng, og værditypen er en int. Dette dictionary skal ikke indholde værdier.

```
using System;
using System.Collections.Generic;

public class Program
{
    public static void Main()
    {
        // Din kode her.
    }
}
```

Opgave 2 – læs og skriv dictionary værdier

Tilføj en ny værdi, hvis nøgle er dit navn, og hvis værdi er din alder. Gør dette ved hjælp af Add -metoden.

Dernæst tilføj en anden værdi til dictionary ved hjælp af indeksnotationen. Denne gang skal du bruge et andet navn og en anden alder.

Til sidst skal du læse det første emne, du tilføjede til dictionary, og skrive det til konsollen ved hjælp af Console.WriteLine.

```
using System;
using System.Collections.Generic;

public class Program
{
    public static void Main()
    {
        Dictionary<string, int> people = new Dictionary<string, int>();

        // din kode her.
    }
}
```

Opgave 3 – fjern et element fra dictionary

Fjern element med nøglen 'Han' fra ordbogen.

```
using System;
using System.Collections.Generic;

public class Program
{
    public static void Main()
    {
        Dictionary<string, bool> characters = new Dictionary<string, bool>()
        {
            { "Luke", true },
            { "Han", false },
            { "Chewbacca", false }
        };

        //Din kode her
    }
}
```

Opgave 4 – gennemløb af dictionary

Opret et foreach-loop efter dictionary deklarationen gennemløb elementer, skriv værdien til konsollen ved hjælp af Console.WriteLine. Din loop-variabel skal angives som "var".