# Zbirke podatkov v Spring4D



**Primož Gabrijelčič**

http://**primoz.gabrijelcic**.org

http://www.**thedelphigeek**.com

# The Delphi Geek

random ramblings on Delphi, programming, Delphi programming, and all the rest

Friday, November 25, 2022

## "Design Patterns", rescheduled

For the last Delphi meeting in Slovenia this year we have organized a small workshop about Design Patterns. As usual, it is intended for Slovenian programmers and will be given in the Slovenian language.

Read more »

Posted by gabr42 at 09:48 No comments:

Labels: Delphi, design patterns, presentations

Sunday, October 16, 2022

## "Design Patterns" workshop in Ljubljana

~~For the last Delphi meeting in Slovenia this year we have organized a small workshop about Design Patterns. As usual, it is intended for Slovenian programmers and will be given in the Slovenian language.~~

Read more »

Posted by gabr42 at 17:15 No comments:

Labels: presentations

Tuesday, May 17, 2022

## We'll meet again (finally)!

After two long-distance years we are finally moving back to normality, starting with a Slovenian RAD Studio meeting next Wednesday in Ljubljana.

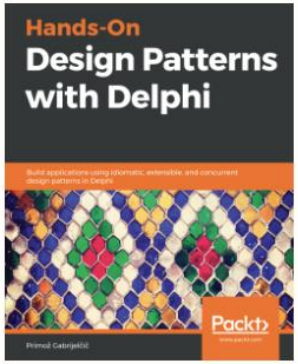
Po dveh letih virtualnih konferenc vas končno spet vabimo na srečanje v živo! Za izgovor za druženje si bomo ogledali novosti v RAD Studiih iz zadnjih dveh let (10.4, 10.4.1, 10.4.2, 11, 11.1), predvsem pa bomo dogodek izkoristili za klepet ob hrani in pijači in ponovno spoznavanje.

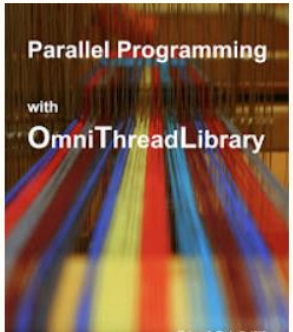
Pridružite se nam 25. maja ob 9h! (klikni za več podatkov in prijavo)

embarcadero MVP

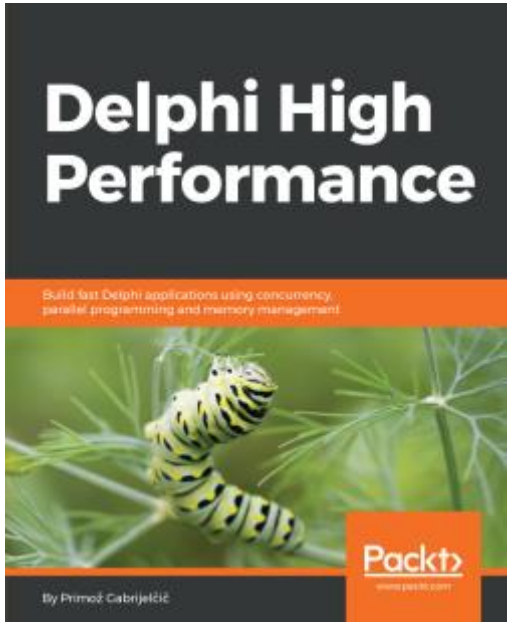
Pages

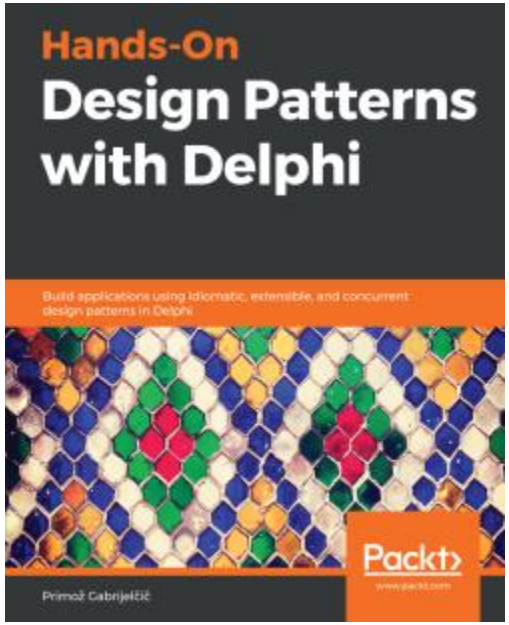
Presentations



# Knjige


https://**delphi-books**.com

# Spring4D

# Spring4D

- Zbirke podatkov z bogatim vmesnikom

- Množica malih pomočnikov (smart pointers, nullable objects, multi-dispatch events …)
- Vsebnik za Dependency Injection
- Knjižnica za prestrezanje in oponašanje (unit testing)
- Nadgradnja RTTI
- ORM
- Enkripcijska knjižnica
- … in še in še

# Zbirke podatkov

- Načeloma
  - https://bitbucket.org/sglienke/spring4d
- Trenutno
  - https://bitbucket.org/sglienke/spring4d/src/develop/
- Licenca: Apache 2.0
  - https://www.tldrlegal.com/license/apache-license-2-0-apache-2-0

# Namestitev

- Prenesi
- Odpakiraj
- Dodaj v potko (.dcu ali .pas)
  - Build.exe

## Spring v1

- Zbirke osnovane na Generics.Collections
- Redefinira
  - TAction (const)
  - TPredicate (const)
- Filtri: TFunc
- Nedefiniran vrstni red nekaterih neurejenih zbirk

## Spring v2

- Samostojna implementacija
- Redefinira
  - TDictionaryOwnerships
  - TPair
- Action, Func, Predicate
- Filtri: Func (const)
- Spremenjen vrstni red param. v CreateDictionary
- Predelan IBidiDictionary
- Vedno znan vrstni red
- ~~IEvent<T>.Invoke~~

# Spring4D v2

- Možne nezdružljivosti, kot je ta:
  - **for var** key **in** dict.Keys **do**
    dict[key] := dict[key] + 'x';

# Zbirke podatkov

# Zbirke podatkov

- Spring.Collections
- Bogat nabor funkcij
- Bogat nabor zbirk – za vsakogar nekaj
- Osnovane na vmesnikih
  - IList<T> namesto TList<T>
- Vse izhajajo iz IEnumerable<T>
- Ne-generični IEnumerable in ICollection
  - Shranjuje Tvalue
  - Namenjen serializaciji podatkov

# IEnumerable in potomci

- IEnumerable<T>
  - IReadOnlyCollection<T>
    - IReadOnlyList<T>
    - IReadOnlyMap<K,V>
      - IReadOnlyDictionary<K,V>
      - IReadOnlyOrderedDictionary<K,V>
      - IReadOnlyMultiMap<K,V>
    - IReadOnlyMultiSet<T>
  - IStack<T>
  - IQueue<T>
  - IDeque<T>

- IEnumerable<T>
  - ICollection<T>
    - IList<T>
      - IObjectList
      - IInterfaceList
    - IMap<K,V>
      - IDictionary<K,V>
        - ISortedDictionary<K,V>
        - IOrderedDictionary<K,V>
        - IBiDiDictionary<K,V>
      - IMultiMap<K,V>
    - ISet<T>
      - ISortedSet<T>
      - IOrderedSet<T>
    - IMultiSet<T>

# IEnumerable<T>

# IEnumerable<T>

- Niz podatkov, po katerem se lahko sprehajamo
- Vrstni red je enak zaporedju vstavljanja
  - Privzeto
  - Zbirke lahko vrstni red spremenijo
- Podpora za **for** .. **In**
- Filtriranje, iskanje, izračuni, testi

# Poizvedbe: IEnumerable<T>

- Concat
- Memoize
- Ordered
- Reversed
- Shuffled
- Skip
- SkipLast
- SkipWhile
- Take
- TakeLast
- TakeWhile
- Where

# Poizvedbe

- Nizanje
  - **for var** el **in** Zbirka.Skip(3).SkipLast(3).Reversed **do**
- Zakasnjeno izvajanje
  - Optimalno delovanje
  - Poizvedbe lahko shranimo in jih izvedemo kasneje

# Dostop: T

- ElementAt[OrDefault]
  - Hiter, ali pa tudi ne
- First[OrDefault]
- Last[OrDefault]
- Single[OrDefault]

# Dostop: boolean

- TryGetElementAt
- TryGetFirst
- TryGetLast
- TryGetSingle

# Izračuni: T

- Aggregate
- Max
- Min
- Sum

# Testi: boolean

- All
- Any
- AtLeast
- AtMost
- Between
- Contains
- EqualsTo
- Exactly
- IsEmpty

# Razno

- Count
- ForEach
- ToArray

# TEnumerable

- Aggregate
- Chunk
- DefaultIfEmpty
- Distinct
- DistinctBy
- Except
- Empty
- From
- Group
- Intersect
- MaxBy
- MinBy
- OfType
- OrderBy[Descending]
- Range
- Repeated
- Select
- SelectMany
- ToDictionary
- ToLookup
- Union
- Zip

# Adapters

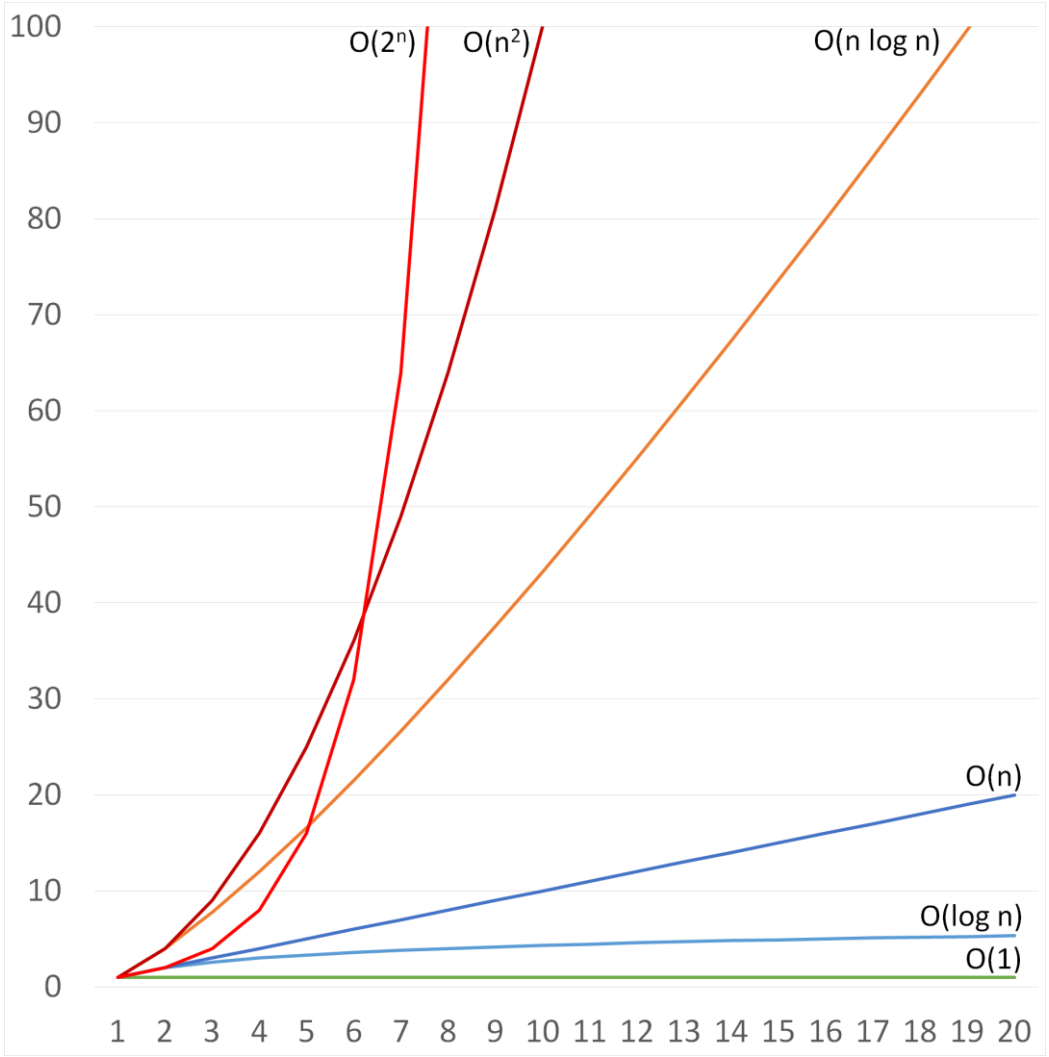
- Spring.Collections.Adapters
- TEnumerableHelper
  - From<T>(TEnumerable<T>): IEnumerable<T>
    - TEnumerable = Spring.Collections
    - TEnumerable<T> = System.Generics.Collections
- TCollectionAdapters
  - CreateDictionary<K,V>(TDictionary<K,V>): IDictionary<K,V>
  - CreateList<T>(TList<T>): IList<T>
  - CreateQueue<T>(TQueue<T>): IQueue<T>
  - CreateStack<T>(TStack<T>): IStack<T>

# ICollection<T>

- Add
- AddRange
- Extract
- Clear
- CopyTo
- MoveTo
- Remove
- RemoveAll
- RemoveRange
- ExtractAll
- ExtractRange
- OnChanged

# Učinkovitost algoritmov

# O()


100
90
80
70
60
50
40
30
20
10
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
O(2ⁿ)
O(n²)
O(n log n)
O(n)
O(log n)
O(1)

# Učinkovitost

- Konstantne faktorje ignoriramo
  - O(10 * n) = O(n)
- Konstantni faktorji so lahko pomembni!
  - Linearno iskanje po majhni tabeli < iskanje po razpršeni tabeli
  - Za majhne *n*

# Učinkovitost

| | |
|---|---|
| **Dostop** | do poljubnega elementa |
| **Iskanje** | elementa po vrednosti |
| **Vstavljanje** | na poljubno mesto (če zbirka to dopušča) |
| **Brisanje** | poljubnega (najdenega) elementa |

# Enostavne strukture

# Enostavne strukture

- Osnovane na TArray<T>
- Hiter dostop do elementov
- Ostale operacije so potencialno počasne

# Seznam

- Zbirka z direktnim dostopom do elementov
  - Items[]

# Seznam - IList<T>

- Insert
- InsertRange
- Delete
- DeleteRange
- ExtractAt
- ExtractRange
- Exchange
- Move
- Reverse
- Sort
- IndexOf
- LastIndexOf
- AsReadOnly
- TrimExcess
- Capacity
- Count
- Items[]
- OwnsObjects

# Seznam - tovarne

- TCollections
  - class function **CreateList**<T>: IList<T>;
    class function **CreateObjectList**<T: class>: IList<T>;
    class function **CreateInterfaceList**<T: IInterface>:IList<T>;
    class function **CreateStringList**: IList<string>;
    class function **CreateSortedList**<T>: IList<T>;
    class function **CreateSortedObjectList**<T>: IList<T>;
    class function **CreateSortedInterfaceList**<T: IInterface>:
    IList<T>;

# Seznam – učinkovitost

| | |
|---|---|
| **Dostop** | O(**1**) |
| **Iskanje** | O(**n**) *neurejen*, O(**log n**) *urejen* |
| **Vstavljanje** | O(**1**) *na koncu, najboljše*<br>O(**n**) *neurejen, povprečno*<br>O(**log n**)/O(**n**) *urejen, povprečno* |
| **Brisanje** | O(**1**) *na koncu, najboljše*<br>O(**n**) povprečno |

# Sklad

- Seznam z dodajanjem/brisanjem na isti strain
- LIFO
  - Last In, First Out

# Sklad – IStack<T>

- Push
- Pop
- Extract
- Peek[OrDefault]
- TryPop
- TryExtract
- TryPeek
- TrimExcess
- Capacity
- OnChanged

# Sklad - tovarne

- **CreateStack**<T>: IStack<T>
- **CreateBoundedStack**<T>(size): IStack<T>

# Vrsta

- Seznam z dodajanjem na eni strani in brisanjem na drugi
- FIFO
  - First In, First Out

# Vrsta – IQueue<T>

- Clear
- Enqueue
- Dequeue
- Peek[OrDefault]
- TryDequeue
- TryEnqueue
- TryPeek
- TrimExcess
- Capacity
- OnChanged

# Vrsta - tovarne

- **CreateQueue**<T>: IQueue<T>
- **CreateBoundedQueue**<T>(size): IQueue<T>
- **CreateEvictingQueue**<T>(size): IQueue<T>

# Dvojna vrsta

- Vrsta z dodajanjem in brisanjem na obeh straneh
- Deque
  - Double-ended queue

# Dvojna vrsta – IDeque<T>

- **CreateDeque**<T>: IDeque<T>
- **CreateBoundedDeque**<T>(size): IDeque<T>
- **CreateEvictingDeque**<T>(size): IDeque<T>

# Sklad / Vrsta / Dvojna vrsta – učinkovitost

| | |
|---|---|
| **Dostop** | O(**n**) |
| **Iskanje** | O(**n**) |
| **Vstavljanje** | O(**1**) *tipično*, O(**n**) *najslabše* |
| **Brisanje** | O(**1**) |

# Kompleksne strukture

# Razpršene tabele

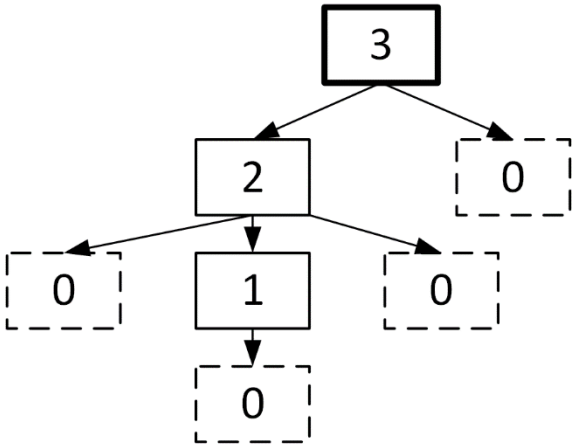
- Tabela
  - Integer → T
- Razpršena tabela
  - T → T1
  - T → f(T): Integer → T1
    - Razpršilna funkcija
- System.Hash
- Spring.Hash, Spring.HashTable

# Razpršene tabele – učinkovitost

| | |
|---|---|
| **Dostop** | O(**1**) |
| **Iskanje** | O(**1**) |
| **Vstavljanje** | O(**1**) |
| **Brisanje** | O(**1**) |

# Drevesa

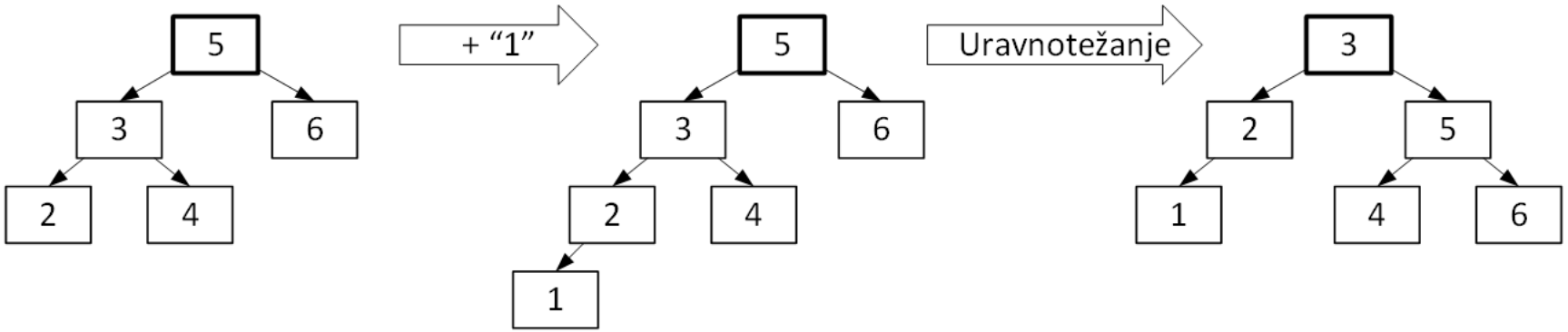
- Rekurzivna struktura
- Drevo je:
  - Prazno drevo ali
  - Vozlišče, ki kaže na eno ali več dreves
- Koren, vozlišče, list
- Višina
- Nivo



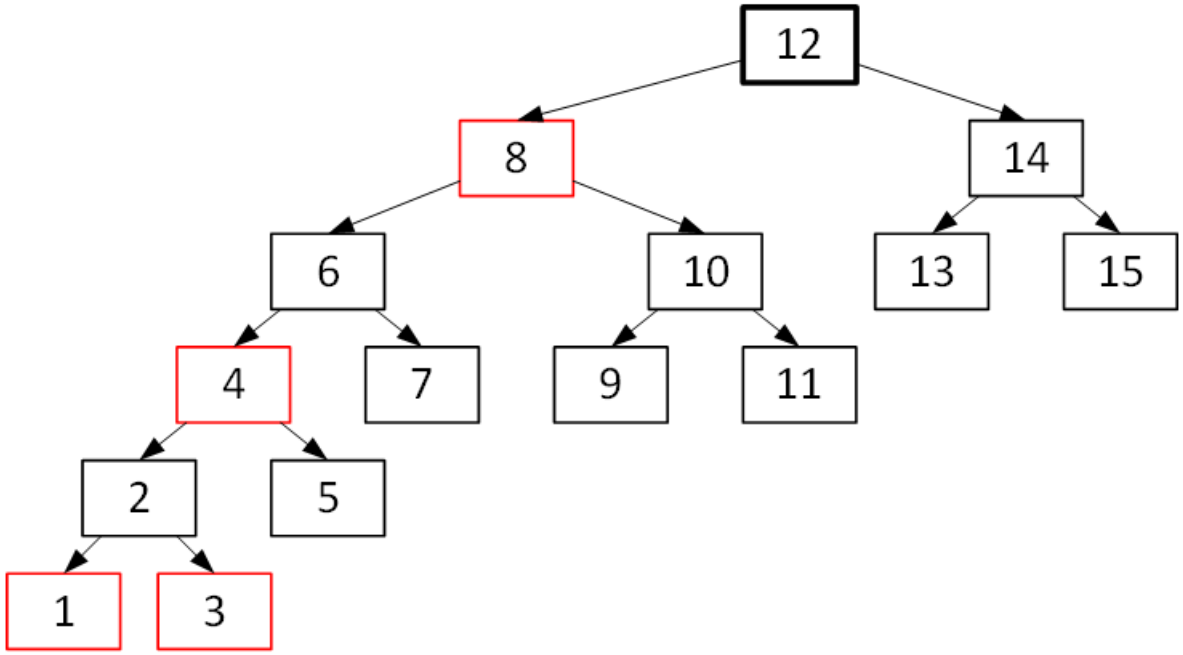
# Drevesa – IBinaryTree<T>, <K,V>

- Count
- Height
- Root
- Add
- Delete
- Exists
- Find
- Clear
- GetEnumerator
- ToArray

# Uravnotežena drevesa



# Rdeče-črna drevesa



# Rdeče-črna drevesa – IRedBlackTree<T>, <K,V>

- FindNode
- DeleteNode

- Spring.Collection.Trees

# Uravnotežena drevesa – učinkovitost

| | |
|---|---|
| **Dostop** | O(**log n**) |
| **Iskanje** | O(**log n**) |
| **Vstavljanje** | O(**log n**) |
| **Brisanje** | O(**log n**) |

# Množica

- Zbirka elementov brez neposrednega dostopa
- Vsaka vrednost je v zbirki največ enkrat

# Množica – ISet<T>

- ExceptWith
- IntersectWith
- UnionWith
- IsSubsetOf
- IsSupersetOf
- SetEquals
- Overlaps
- TrimExcess
- Capacity

# Množica – IOrderedSet<T>

- IndexOf
- Items[]

# Množica – tovarne

- **CreateSet**<T>: IOrderedSet<T>
  - razpršena tabela
- **CreateSortedSet**<T>: ISet<T>
  - *drevo*

# Vreča

- Zbirka elementov brez neposrednega dostopa
- Vsaka vrednost je lahko v zbirki večkrat

# Vreča – IMultiSet<T>

- Add
- Remove
- SetEquals
- OrderedByCount
- Entries
- Items
- ItemCount[item:T]

# Vreča – tovarne

- **CreateMultiSet**<T>: IMultiSet<T>
  - razpršena tabela
- **CreateSortedMultiSet**<T>: IMultiSet<T>
  - drevo

# Slovar

- Preslikava podatkov (ključev) v druge podatke (vrednosti)
- Vsak ključ je lahko vsebovan samo enkrat
- Dostop do ključev je hiter, do vrednosti počasen

# Slovar – IMap<K,V>

- [Try]Add
- Remove[Range]
- Extract
- Contains
- ContainsKey
- ContainsValue
- Keys
- Values
- OnKeyChanged
- OnValueChanged

# Slovar – IDictionary<K,V>

- *AddOrSetValue*
- Extract
- GetValueOrDefault
- TryExtract
- TryUpdateValue
- AsReadOnly
- TrimExcess
- Capacity
- **Items[]**

# Slovar – IOrderedDictionary<K,V>

- IndexOf
- Items[]

# Slovar – tovarne

- **CreateDictionary**<K, V>: IOrderedDictionary<K, V>
  - razpršena tabela
- **CreateSortedDictionary**<K, V>: IDictionary<K, V>
  - drevo

# Dvosmerni slovar

- Preslikava ključev v vrednosti in vrednosti v ključe
- Hiter dostop tudi do vrednosti
- Vsaka vrednost je lahko prisotna samo enkrat

# Dvosmerni slovar – IBidiDictionary<K,V>

- Inverse

# Dvosmerni slovar – tovarne

- **CreateBidiDictionary**<K, V>: IBidiDictionary<K, V>
  - razpršena tabela

# MultiMap

- Preslikava podatkov (ključev) v zbirko podatkov (vrednosti)
- Vrednosti so lahko organizirane na različne načine

# MultiMap – IMultiMap<K,V>

- Add
- AddRange
- Extract
- TryGetValues
- AsReadOnly
- Items

# MultiMap – tovarne

- **CreateMultiMap**<K, V>: IMultiMap<K, V>
  - razpršena tabela / seznam
- **CreateHashMultiMap**<K, V>:IMultiMap<TKey, TValue>
  - razpršena tabela / razpršena tabela
- **CreateTreeMultiMap**<K, V>: IMultiMap<K, V>
  - razpršena tabela / drevo

# MultiMap – tovarne

- **CreateSortedMultiMap**<K, V>: IMultiMap<K, V>
  - drevo / seznam
- **CreateSortedHashMultiMap**<K, V>:  IMultiMap<K, V>
  - drevo / razpršena tabela
- **CreateSortedTreeMultiMap**<K, V>: IMultiMap<K, V>
  - drevo / drevo

# Čas za vprašanja