Sovelto

Kokoelmat

Kokoelmat

- Kokoelmat ovat olio-ohjelmoinnin dynaamisia tietorakenteita
- Rakenne on erittäin dynaaminen
  - Kasvaa automaattisesti kun siihen lisätään tietoa
  - Välistä voidaan poistaa viitteitä lennosta
- ArrayList on kokoelma object – tyyppisiä viittauksia
- Nimiavaruus: System.Collections
- HUOM: System.Collections-nimiavaruuden kokoelmia **ei** pitäisi enää käyttää vaan vastaavia generics-kokoelmia: List<T>

```
ArrayList lista = new ArrayList();
lista.Add("Eka");
lista.Add("Toka");
lista.Add("Kolmas");

foreach(string s in lista)
{
  Console.WriteLine(s);
}
```

Kokoelmien tyypit - perustietorakenteita

- List on dynaaminen taulukko, alkiot ovat indeksoitavissa kokonaisluvulla
  - lisäys: Add ja Insert. Poisto: Remove ja RemoveAt
- Dictionary on ns. hajatustaulukko (HashTable), viittaa osin tekniseen toteutukseen ja siihen että arvon talletuksessa määritellään avain millä arvoon viitataan, avaimen pitää olla yksilöllinen
- Queue on jono, FIFO-tyypinen tietorakenne (First-In-First-Out). Jonon alkuun ja loppuun voi lisätä ja poistaa alkioita ja jono voidaan käytä kokonaan läpi.
- Stack on pino jossa on kolme toimintoa: Push, Pop ja Peek.
  - Push 'painaa' pinon päällimmäiseksi alkio
  - Pop ponnauttaa pinon päällimmäisen alkion pois pinosta
  - Peek kurkistaa päällimmäisen arvon

Generics - kokoelmat

- Generics – kokoelmille voidaan tarkasti määrittää käytettävä tietotyyppi
- Ne ovat tyyppiturvallisia
- Nimiavaruus: System.Collections.Generic
- Esim. lista, johon voidaan laittaa vain henkilöitä

```
List<Henkilö> lista = new List<Henkilö>();
lista.Add(new Henkilö("Anna"));
lista.Add(new Henkilö("Saara"));

foreach(Henkilö h in lista)
{
 Console.WriteLine(h.Nimi);
}
```

```
public class Henkilö
 {
   public string Nimi { get; set;}
   public Henkilö(string nimi)
   {
     Nimi = nimi;
   }
 }
```

ReadOnlyCollection

- Kokoelma voidaan kapseloida luokan sisään siten, että sitä voidaan vain lukea
- ReadOnlyCollection – luokka on tarkoitettu tähän käyttötarkoitukseen
- Luokan konstruktorille annetaan viite varsinaisen tiedon sisältävään kokoelmaluokkaan
- Nimiavaruus: System.Collections.ObjectModel

ReadOnlyCollection

```
HenkilöLista lista = new HenkilöLista();
lista.Add(new Henkilö("Kari"));

foreach(Henkilö h in lista.Henkilöt){
      Console.WriteLine(h.Nimi);
}
```

```
public class HenkilöLista  {
  public ReadOnlyCollection<Henkilö> Henkilöt;

  private List<Henkilö> lista = new List<Henkilö>();

  public HenkilöLista(){
    Henkilöt = new ReadOnlyCollection<Henkilö>(lista);
  }

  public void Add(Henkilö h){
    lista.Add(h);
  }
 }
```

Wanhat System.Collection –kokoelmat (älä käytä näitä)

- ArrayList
  - Dynaaminen taulukko ilman avainnusta (indeksi on järjestysnumero)
- HashTable
  - Avainnettu kokoelma (vrt. Collection) perustuen avainten HashCodeen.
- SortedList
  - Avainnettu kokoelma, jota pidetään avaimen mukaisessa järjestyksessä
- Stack, Queue
  - Pino ja jono- tyyppinen jäsenten käsittely

- Näihin saatat törmätä googlaamalla mutta älä käytä omassa koodissa, ovat 'vanhentuneita' mutta toki toimivat siitä huolimatta

Generics-kokoelmat

- System.Collections.Generic nimiavaruus
- Käytä aina näitä, ovat tyyppi- ja boxing -turvallisia

Generic	Concrete
List<T>	ArrayList
Dictionary<Tkey, Titem>	Hashtable
SortedDictionary<Tkey, Titem>	SortedList
Stack<T>	Stack
Queue<T>	Queue

System.Collections.ObjectModel

Luokka	Kuvaus	Versio
Collection<T>	Kantaluokka geneerisille kokoelmaluokille	2.0
KeyedCollection<Tkey, TItem>	Kantaluokka avainnetuille kokoelmaluokille	2.0
ObservableCollection<T>	Kantaluokka kokoelmille, lähettää tapahtuman CollectionChanged kun kokoelman sisältö muuttuu, mukana vasta Frameworkin versiossa 3.0	3.0
ReadOnlyCollection<T>	ReadOnly-kokoelma, käytetään yhdessä List-luokan kanssa	2.0
ReadOnlyObservableCollection<T>		3.0

- Huomaa myös nimiavaruudessa System.Collections.Concurrent olevat säieturvalliset kokoelmaluokat

Extension Methods

- Staattisen metodin (vain metodit, ei property tai indekseri) kutsuminen olion kautta
- Olio, jonka kautta kutsu tehdään, on ensimmäinen parametri, extension ”enabloidaan” this-määreellä (metodilla voi olla muitakin parametreja)
- Extension-metodin nimiavaruus on esiteltävä using -lauseella, kutsussa ei voi käyttää nimiavaruutta

```
string s = "Tämä on lause.";
//kutsu ”wanhalla” tavalla
StringExtenderit.SanojenLkm(s));
//Extension Method -kutsutapa
s.SanojenLkm();
```

```
static class StringExtenderit{
  static public int SanojenLkm(this string s) {
     string[ ] sanat = s.Trim().Split(new char[] { ',', ' ', '.' });
     return sanat.Length;
  }
}
```

Extension Methods ja kokoelmaluokat

- Generics-kokoelmaluokkien käsittelyyn on runsaasti Extension-metodeja

```
List<int> lista = new List<int>();
for (int i = 0; i < 10; i++) {
    lista.Add(i);
}

int max = lista.Max();
int min = lista.Min();
double ka = lista.Average();
int summa = lista.Sum();
```