Bandiera Roberto – aprile 2024

Come aggiungere la funzione Second() alla libreria LINQ

Con la tecnica degli Extension Methods si possono aggiungere ulteriori funzioni alla interfaccia IEnumerable<T>, ovvero aggiungere ulteriori funzioni alla libreria Linq.
Si ricorda che la libreria Linq è costituita di metodi di estensione per oggetti IEnumerable<T>.

A tal fine si crea una classe statica che funge da contenitore di questi metodi.
Si scrivono le funzioni Second() e SecondOrDefault() del tutto analoghe alle già esistenti First() e FirstOrDefault()

```
// estensione di  LINQ
public static class EstensioneLinq
{
      // funzione che restituisce il secondo elemento della lista
      // se non esiste il secondo elemento si ha un errore run-time
      public static T Second<T>(this IEnumerable<T> lista)
      {
            // ci si sposta al secondo elemento mediante un enumeratore
            var e = lista.GetEnumerator();
            e.MoveNext();
            e.MoveNext();
            return e.Current;
      }

      public static T SecondOrDefault<T>(this IEnumerable<T> lista)
      {
            var e = lista.GetEnumerator();
            if( e.MoveNext() && e.MoveNext() )
            {
                  return e.Current;
            }
            else
            {
                  return  default(T);  // null
            }
      }
}
```

In generale, le funzioni Linq sono componibili, ovvero possono essere chiamate in sequenza, perché esse agiscono su un IEnumerable<T> e restituiscono un IEnumerable<T>.
Per esempio, la funzione Estrai() estrae gli elementi della lista che soddisfano una determinata condizione, esattamente come fa Where()

```
public static IEnumerable<T> Estrai<T>(this IEnumerable<T> lista,
Predicate<T> condizione)
{
      var ris = new List<T>();
      foreach(T elemento in lista)
```

```
      {
            if (condizione(elemento))
            {
                  ris.Add(elemento);
            }
      }
      return ris as IEnumerable<T>;
}
```

La funzione Prendi() si comporta esattamente come la Select() per estrarre un singolo attributo da ciacuno oggetto della lista di partenza.
L'estrazione dell'attributo avviene mediante un "selector" di tipo Func<T, Tresult>

```
public static IEnumerable<TResult> Prendi<T, TResult>(this
IEnumerable<T> lista, Func<T, TResult> selector)
{
      var ris = new List<TResult>();
      var e = lista.GetEnumerator();
      // scorro tutti gli elementi della collezione
      while(e.MoveNext())
      {
            var b = selector.Target;
            var appo = selector( e.Current );
            ris.Add( appo );
      }
      return ris as IEnumerable<TResult>;
}
```

Esempio di utilizzo:

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

public class Persona
{
      public string Nome {get;set;}
      public string Città {get;set;}
      public int Età {get;set;}
}
public class Studente: Persona
{
      public int Voto {get;set;}
}


public class Program
{
```

```
public static void Main()
{
      List<Persona> lista = new List<Persona>();
      lista.Add(new Persona()
            {Nome = "mario", Città="tv", Età = 20});
      lista.Add(new Persona()
            {Nome = "luigi", Città="tv", Età = 12});
      lista.Add(new Persona()
            {Nome = "matteo", Città="pd", Età = 20});
      lista.Add(new Persona()
            {Nome = "franco", Città="pd", Età = 40});
      lista.Add(new Persona()
            {Nome = "gianni", Città="pd", Età = 8});
      lista.Add(new Studente(){Nome = "alfio",
                                    Città="tv",
                                    Età = 15,
                                    Voto= 8});
      lista.Add(new Studente(){Nome = "lara",
                                    Città="tv",
                                    Età = 15,
                                    Voto= 7});

      Persona primo = lista.FirstOrDefault();
      Persona secondo = lista.SecondOrDefault();
      if (primo !=null)
      {
            Console.WriteLine("1) " + primo.Nome);
      }
      else
      {
            Console.WriteLine("la lista è vuota");;
      }
      if(secondo!=null)
      {
            Console.WriteLine("2) " + secondo.Nome);
      }
      else
      {
            Console.WriteLine("non c'è il secondo");
      }

      List<string> ris =
            lista.Estrai(x=>x.Città==”tv”).Prendi(x=>x.Nome)
            .ToList();
      // equivale a lista.Where(x=>x.Città==”tv”).Select(x=>x.Nome)
      // .ToList();
```