

## Chapter 1 – LINQ Introduction

### Page 5-6

In Listing 1-1 and Listing 1-2 the *Order* type should be a *class* instead of a *struct*. The following are the corrected listings.

#### Listing 1-1

Type declarations with simple relationships

```
public class Customer {
    public string Name;
    public string City;
    public Order[] Orders;
}
public class Order {
    public int Quantity;
    public Product Product;
}
public class Product {
    public int IdProduct;
    public decimal Price;
    public string ProductName;
}
```

#### Listing 1-2

Type declarations with two-way relationships

```
public class Customer {
    public string Name;
    public string City;
    public Order[] Orders;
}
public class Order {
    public int Quantity;
    public Product Product;
    public Customer Customer;
}
public class Product {
    public int IdProduct;
    public decimal Price;
    public string ProductName;
    public Order[] Orders;
}
```

## Page 16

The C# 3.0 syntax for the query contains the keyword *dim* instead of *var*. The following is the right code in C# 3.0.

```
var book =
    new XElement( "Book",
        new XAttribute( "Title", "Introducing LINQ" ),
        from person in team
        where person.Role == "Author"
        select new XElement( "Author", person.Name ) );
```

## Chapter 2 – C# Language Features

### Page 35

The lambda expression definition for *Func* is this:

```
public delegate T Func<T>();
public delegate T Func<A0, T>( A0 arg0 );
public delegate T Func<A0, A1, T>( A0 arg0, A1 arg1 );
public delegate T Func<A0, A1, A2, T>( A0 arg0, A1 arg1, A2 arg2 );
public delegate T Func<A0, A1, A2, A3, T>( A0 arg0, A1 arg1, A2 arg2, A3 arg3 );
```

### Page 44

In the Note at the top of page 44 the last sentence should be "In this way the object is not visible to another thread until it is ~~not~~ completely initialized."

### Page 46

The first sentence of the last paragraph on page 46 should be "The generated class has a public property and an underlying private field for each argument contained in the initializer: ~~its~~ the name and type of the properties are inferred from the object initializer itself"

## Chapter 3 – Visual Basic 9.0 Language Features

### Page 51-52

At the moment the Nullable Types feature is still not implemented. This feature should appear before RTM, probably in Beta 2. The code in Listing 3-1 and in Listing 3-2 does not work with Beta 1.

### Page 59

The collection initializer syntax has been cut from Visual Basic 9.0. It will appear in a later version of Visual Basic. The code in Listing 3-16 does not work neither with the current beta nor with RTM.

### Page 61

At the moment lambda expression syntax is not supported in Visual Basic 9.0 compiler. This feature should appear before RTM, probably in Beta 2. The code in Listing 3-20 and in Listing 3-21 does not work with Beta 1.

### Page 62

At the moment lambda expression syntax is not supported in Visual Basic 9.0 compiler. This feature should appear before RTM, probably in Beta 2. The code corresponding to closures does not work with Beta 1.

### Page 66

The collection initializer syntax has been cut from Visual Basic 9.0. It will appear in a later version of Visual Basic. The code in Listing 3-29 does not work neither with the current beta nor with RTM.

## Chapter 4 – LINQ Syntax Fundamentals

### Page 80

In Listing 4-10 the *orders* variable should be declared as *var* instead of *IEnumerable<Order>*. The following is the corrected listing.

### Listing 4-10

The list of *Quantity* and *IdProduct* of orders made by Italian customers, written with a query expression

```
var orders =  
    from c in customers  
    where c.Country == Countries.Italy  
        from o in c.Orders  
        select new {o.Quantity, o.IdProduct};
```

### Page 87

The last line of execution result is:  
{Month=July, Shipped=False, IdProduct=5, Price=50}