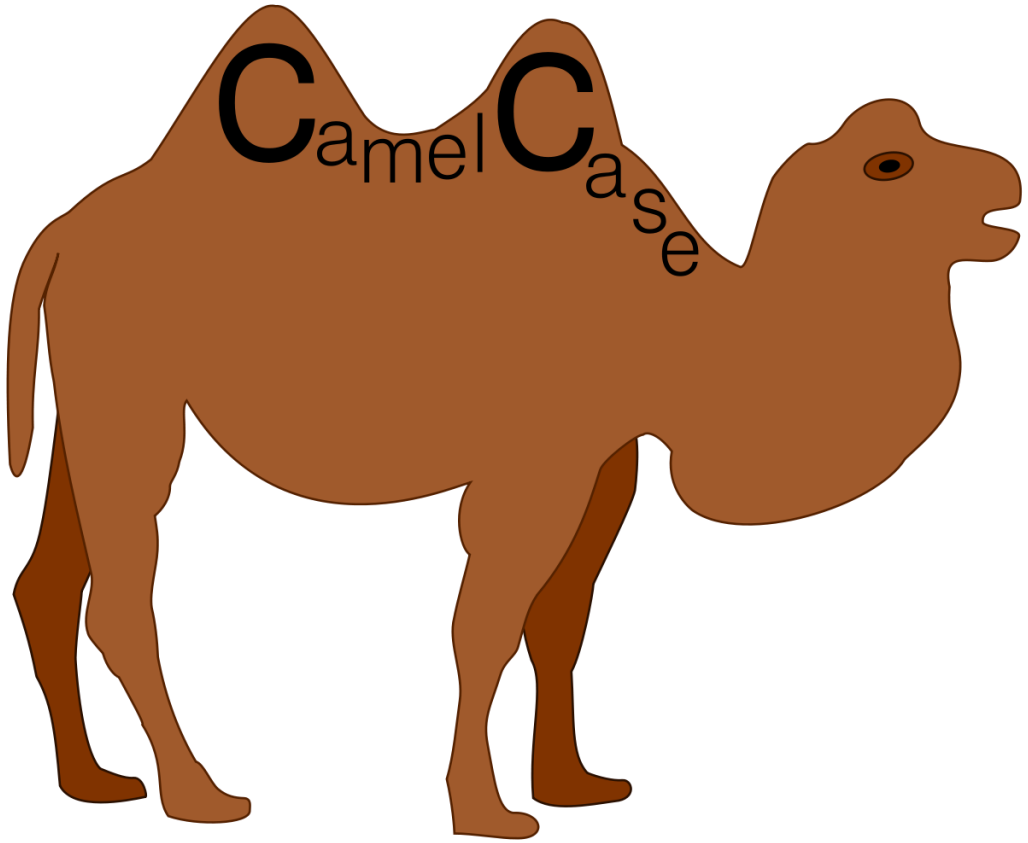
Convenzione dei nomi

in c#

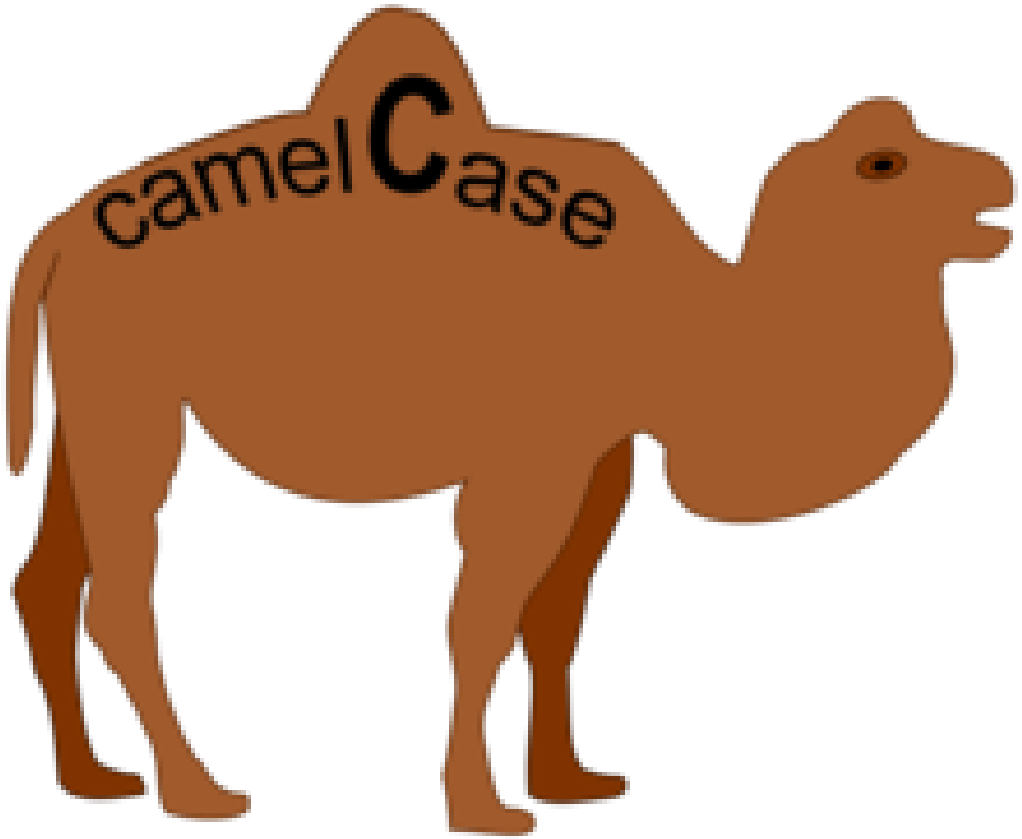
Convenzione dei nomi

- Una serie di regole usate per scegliere i nomi di identificatori: variabili tipi e funzioni ecc.;
- Sono usate per semplificare la lettura del codice
- Accelerare manutenzione e modifica del codice sorgente;
- Consentire correzioni automatiche da parte di tools di correzione del codice;
- Evitare conflitti nel codice appartenente a produttori diversi (namespaces);

Convenzione dei nomi



Convenzione dei nomi



Convenzione dei nomi

Camel case, in genere rappresentato dal termine camelCase è la modalità di scrivere sentenze in cui il secondo nome è rappresentato da un carattere Maiuscolo (eBay).

In realtà ci sono due alternative del Camel case:

- upper camel case
  - Carattere iniziale Maiuscolo noto anche con il nome di Pascal case;
- lower camel case
  - Carattere iniziale minuscolo, conosciuto anche come Dromedary case

In Microsoft il termine camel case si riferisce solo al lower camel case

Convenzione dei nomi

Si usa PascalCasing per i nomi delle classi e dei metodi:

```
public class ClientActivity
{
    public void ClearStatistics()
    {
        //...
    }
    public void CalculateStatistics()
    {
        //...
    }
}
```

Convenzione dei nomi

Si usa camelCasing per i parametri dei metodi e le variabili locali:

```
public class UserLog
{
   public void Add(LogEvent logEvent)
   {
      int itemCount = logEvent.Items.Count;
      // ...
   }
}
```

Convenzione dei nomi

Non usare Hungarian notation o altre modalità di riferimento dei tipi per nominare gli identificatori

```
// Correct
int counter;
string name;

// Avoid
int iCounter;
string strName;
```

Convenzione dei nomi

Non usare caratteri maiuscoli per costanti e variabili di sola lettura:

```
// Correct
public static const string ShippingType =
"DropShip";
// Avoid
public static const string SHIPPINGTYPE =
"DropShip";
```

Convenzione dei nomi

Usa nomi significativi per le variabili. Il seguente esempio usa seattleCustomers per i clienti che sono a Seattle:

```
var seattleCustomers = from cust in customers
where cust.City == "Seattle"
select cust.Name;
```

Convenzione dei nomi

Evitare abbreviazioni. Eccezioni: abbreviazioni comunemente usate come nomi: Id, Xml, Ftp, Uri.

```
// Correct
UserGroup userGroup;
Assignment employeeAssignment;
// Avoid
UserGroup usrGrp;
Assignment empAssignment;
// Exceptions
CustomerId customerId;
XmlDocument xmlDocument;
FtpHelper ftpHelper;
UriPart uriPart;
```

Convenzione dei nomi

Non usare il segno Underscore negli identificatori. Eccezione: si può usare solo con nomi di campi privati:

```
// Correct
public DateTime clientAppointment;
public TimeSpan timeLeft;
// Avoid
public DateTime client_Appointment;
public TimeSpan time_Left;
// Exception
private DateTime _registrationDate;
```

Convenzione dei nomi

Usa nomi o nomi composti per le classi.

```
public class Employee
{
}
public class BusinessLocation
{
}
public class DocumentCollection
{
}
```

Convenzione dei nomi

Usa il prefisso I per le interfacce.

```
public interface IShape
{
}
public interface IShapeCollection
{
}
public interface IGroupable
{
}
```

Convenzione dei nomi

Dare nomi a file sorgente in relazione ai nomi delle classi. Eccezioni: il nome dei file delle partial classes riflette il fine, per esempio designer, generated, etc.

```
Located in Task.cs
public partial class Task
{
}
// Located in Task.generated.cs
public partial class Task
{
}
```

Convenzione dei nomi

Organizzare i nomi dei namespaces con una struttura ben definita:

```
// Examples
namespace Company.Product.Module.SubModule
{
}
namespace Product.Module.Component
{
}
namespace Product.Layer.Module.Group
{
}
```

Convenzione dei nomi

Allineare verticalmente le parentesi:

```
// Correct
class Program
{
   static void Main(string[] args)
   {
      //...
   }
}
```

Convenzione dei nomi

Dichiarare tutte le variabili in alto nelle classi, quelle statiche al primo posto.

```
// Correct
public class Account
{
  public static string BankName;
  public string Number { get; set; }
  public DateTime DateOpened { get; set; }
// Constructor
  public Account()
  {
    // ...
  }
}
```

Convenzione dei nomi

Usare nomi singolari per le enumerazioni.

```
// Correct
public enum Color
{
   Red,
   Green,
   Blue,
   Yellow,
   Magenta,
   Cyan
}
```

Convenzione dei nomi

Non specificare il tipo per una enumerazione o per il suoi valori

```
// Don't
public enum Direction : long
{
   North = 1,
   East = 2,
   South = 3,
   West = 4
}
```

Convenzione dei nomi

Non usare il suffisso Enum nell'enumerazione dei tipi:

```
// Don't
public enum CoinEnum
{
   Penny,
   Nickel,
   Dime,
   Quarter,
   Dollar
}
```

Convenzione dei nomi

Al fine di escludere situazioni di conflitto, non creare nomi di parametri nei metodi o costruttori che differiscono solo per una maiuscola:

```
// Avoid
private void MyFunction(string name, string
Name)
{
   //...
}
```