C# and .NET Framework
Bài 4: .NET và các lớp cơ bản

Đoàn Quang Minh
minhdqtt@gmail.com
http://www.VTPortal.net
Last update: 28. December 2006

Mục lục

- System.Object
- Xử lý String
- Regular Expression
- Groups of Objects
- Reflection
- Threading

System.Object

- Là lớp cơ bản của C#
  - Nếu không nói gì, một lớp bất kỳ coi như được kế thừa từ Object
- Các phương thức
  - public virutal string ToString()
    - override phương thức này để chuyển một đối tượng thành xâu ký tự.
    - Thường dùng khi kết xuất thông tin về đối tượng.
  - public virutal int GetHashTable()
    - Trả về một giá trị băm của đối tượng
    - Thường dùng khi tạo khoá truy xuất cho đối tượng trong một tập dữ liệu như bảng băm hoặc từ điển.
  - public virutal bool Equals(object obj)
  - public static bool Equals(object objA, object objB)
  - public static bool ReferenceEquals(object objA, object objB)
    - So sánh hai đối tượng

System.Object

- Các phương thức (tiếp)
  - protected virtual void Finalize()
    - Mang ý nghĩa là hàm huỷ, được gọi bởi bộ thu gom rác. Mặc định không thực thi gì.
    - Chỉ override khi cần thiết, ví dụ đóng tập tin.
  - public Type GetType()
    - Trả về kiểu đối tượng, bao gồm lớp cha, các phương thức, thuộc tính,…
  - protected object MemberwiseClone()
    - Copy một đối tượng. Chú ý chỉ copy các tham chiếu bên trong đối tượng

Xử lý String

- Có 2 lớp hay được dùng để xử lý xâu
  - String: xử lý các xâu ký tự
  - StringBuilder: xây dựng một xâu ký tự
- String: chứa các phương thức cơ bản trong việc xử lý xâu ký tự.
  - Compare(): so sánh hai xâu.
  - CompareOrdinal(): so sánh, nhưng không tính đến văn hoá (culture)
  - Format(): định dạng xâu dựa trên biểu thức định dạng và các tham số đầu vào

Xử lý String

- Các phương thức của String (tiếp)
  - IndexOf(), IndexOfAny(), LastIndexOf(), LastIndexOfAny(): tìm kiếm chuỗi ký tự, hoặc một phần chuỗi ký tự trong một xâu cho trước.
  - PadLeft(), PadRight(): điền thêm vào đầu hoặc cuối xâu bởi ký tự cho trước.
  - Replace(): thay thế một mẫu trong xâu bởi một chuỗi ký tự khác.
  - Split(): cắt một xâu thành một tập hợp các xâu con dựa theo một ký tự phân cách cho trước.
  - Substring(): lấy một phần xâu con từ một xâu cho trước.
  - ToLower(), ToUpper(): biến các ký tự trong xâu thành ký tự thường hoặc ký tự hoa.
  - Trim(), TrimEnd(), TrimStart(): xoá các ký tự trắng ở đầu, cuối xâu.
  - Insert(), Remove(): chèn vào, xoá đi một xâu con trong một xâu cho trước.
  - StartsWith(), EndsWith(): kiểm tra xem xâu có bắt đầu, kết thúc bởi một xâu khác.

Xử lý String

- Để xử lý xâu, chúng ta hay dùng các phép toán như so sánh, gán, cộng thêm (+=)
  - Ưu điểm: Các phép toán đơn giản, dễ dùng
  - Nhược điểm: Hiệu suất quản lý bộ nhớ thấp
- Khi cần xây dựng một chuỗi văn bản phức tạp và có độ dài tương đối lớn, chúng ta dùng lớp StringBuilder
  - StringBuilder cho phép nối thêm các xâu mới vào trong một tập hợp các xâu có sẵn mà không cần quá nhiều các thao tác xử lý vùng nhớ
  - Ví dụ:

Xử lý String

- StringBuilder
  - Append(): nối thêm vào đuôi một xâu mới
  - Insert(): chèn vào một vị trí bất kỳ một xâu mới.
  - Remove(): xoá bỏ một xâu con tại vị trí hiện thời
  - ToString(): sau khi xây dựng tập các xâu xong, phương thức này biến đổi tập các xâu thành chuỗi văn bản duy nhất.

Xử lý String

- String.Format: định dạng xâu
  - Giống như hàm printf() của C, phương thức static Format của lớp String cho phép định dạng một chuỗi các tham số theo mẫu cho trước.
  - Cú pháp:

public static string **Format(** string *format*, object *arg0* **);**

Xử lý String

- Chuỗi format chứa một hoặc nhiều các đối tượng cần format, theo mẫu {index[,alignment][:formatString]}
  - index: chỉ số của đối tượng trong danh sách các đối tượng cần format
  - alignment: tuỳ chọn, là độ dài tối thiểu để chứa giá trij chuỗi của đối tượng đã được format
  - formatString: mã format.
  - Ví dụ, string.Format("I have {0,-4:G} computers", x), với x = 2 thì giá trị là "I have 2___ computers"

Biểu thức chính quy (Regular Expression)

- Regular Expression là lớp thực hiện các thao tác liên quan đến biểu thức chính quy:
  - Gồm một tập các ký tự đại diện;
  - Các phương thức phục vụ cho việc tìm kiếm và thay thế;
  - Sử dụng biểu thức chính quy, có thể thực hiện các công việc phức tạp về xử lý chuỗi
    - Kiểm định chuỗi đầu vào theo một tiêu chuẩn nào đó;
    - Định dạng lại chuỗi (thay thế các ký tự không hợp lệ);
    - TÌm kiếm và trích từ chuỗi đầu vào ra những thành phần đặc biệt.

Biểu thức chính quy (Regular Expression)

Ký tự	Ý nghĩa
^	Bắt đầu của chuỗi
$	Kết thúc của chuỗi
.	Tất cả ký tự, ngoại trừ xuống dòng \n
*	Lặp lại 0 lần hoặc nhiều hơn
+	Lặp lại ít nhất 1 lần
?	Lặp lại 0 hoặc 1 lần
\s	Khoảng trắng, bao gồm cả tab
\S	Tất cả các ký tự mà không là khoảng trắng
\b	Kết thúc nhóm
\B	Không kết thúc nhóm

Biểu thức chính quy (Regular Expression)

- Biểu thức chính quy trong C#
  - Nằm trong namespace System.Text.RegularExpression
  - Cung cấp các lớp Regex, Match,…
- Ví dụ
  - Kiểm tra xem 1 chuỗi đầu vào có là số nguyên hay không?

```
Regex re = new Regex(@"\d+");
Match m = re.Match(s);
if (m.Success)
{
    // match is found, s is a number
}
else
{
    // match not found, s isn't a number
}
```

- Bài tập: kiểm tra một chuỗi có biểu diễn một địa chỉ mail hay không?

Group of Objects

- Group of Objects là các lớp trong đó cho phép quản lý một tập hợp các đối tượng có cùng kiểu.
  - ArrayList: tương tự như mảng, nhưng có nhiều tính năng ưu việt: cho phép thêm, chèn, xoá, sắp xếp, tìm kiếm nhị phân,…
  - Collection: đối tượng tập hợp, trên đó có cài đặt giao tiếp IEnumerable cho phép duyệt từng phần tử trong tập hợp.
    - Stack: Cung cấp cơ chế FILO, có 2 phương thức đặc biệt là Push() và Pop()
    - Queue: cơ chế FIFO, có 2 phương thức đặc biệt là Enqueue() và Dequeue()
  - Dictionary: đối tượng từ điển, cung cấp cơ chế tìm kiếm đối tượng thông qua khoá
    - Hashtable: bảng băm, mỗi đối tượng sẽ được đại diện bởi một giá trị băm, gọi là khoá. Tác dụng tăng tốc trong các thao tác tìm kiếm.

Generics

- Gererics cung cấp các lớp cho phép sử lí một tập các đối tượng với kiểu của đối tượng như là tham số đầu vào.
  - List<T>: cho phép thao tác và xử lí một danh sách các đối tượng có kiểu T;
  - Stack<T>: cung cấp cơ chế FILO với kiểu dữ liệu T;
  - Queue<T>: cung cấp cơ chế FIFO với kiểu dữ liệu T;
  - LinkedList<T>: cung cấp một danh sách liên kết đôi xử lí các đối tượng có kiểu T;
  - Dictionary<K,T>: cung cấp một từ điển với kiểu dữ liệu là T, kiểu khóa là K.

Reflection

- Reflection là thuật ngữ chỉ các lớp trong .NET cho phép chúng ta có thể đọc được thông tin về các assembly. Chúng nằm trong namespace System.Reflection
  - Type: cung cấp thông tin về kiểu của một đối tượng. Bao gồm các thông tin như tên kiểu, tên đầy đủ (gồm cả namespace), tên lớp cơ sở,… Ngoài ra, có thể lấy được các thông tin khác như các phương thức, các trường, các sự kiện, các giao tiếp,…
  - Assembly: chứa thông tin về assembly, bao gồm các thông tin như tên assembly, tên công ty, phiên bản,…

Threading

- Các hệ thống thực đều là hệ đa tiến trình:
  - Các ứng dụng trong Windows chạy song song.
  - Với một công việc, thông thường sẽ gồm nhiều công việc nhỏ chạy song song.
  - .NET hỗ trợ lập trình song song dựa vào thread.
- Để tạo nhiều tiến trình chạy song song, chúng ta sẽ tạo ra các Thread
  - Các thuộc tính:
    - Name: tên của thread
    - Priority: mức độ ưu tiên của thread
  - Các phương thức:
    - Start(): khởi động thread
    - Suspend(): tạm ngưng thread
    - Resume(): kích hoạt lại thread đang tạm ngưng
    - Abort(): huỷ (ngắt giữa chừng) thread

Tài liệu tham khảo

- Professional C#, Second Edition
- http://www.asp.net
- http://www.microsoft.com/net/default.mspx
- http://www.codeproject.com
- Địa chỉ download tài liệu http://www.thanglong.edu.vn/nghien-cuu-phat-trien/thang-long/tab.aspx
- Diễn đàn C# & .NET http://www.thanglong.edu.vn/forum/cmd/0/category/hoc-tap-nghien-cuu/dot-net/tab.aspx