**Data-manipulation**

|  |  |
| --- | --- |
| Ý chính | Nội dung chi tiết |
| Data manipulation  (thao tác dữ liệu) | Bất cứ lúc nào văn bản cần được ghi lại, xử lý, gửi giữa các ứng dụng hoặc hiển thị, chuỗi sẽ có ích：  + Lấy trạng thái của một hoạt động  + Trích xuất dữ liệu từ văn bản lớn  + Hiển thị thông tin đến người dùng |
| Phương thức | * **Concat** : nối chuỗi   Ex: String.concat (varname1,varname2)   * **Contains**: kiểm tra chuỗi chỉ định có xuất hiện trong chuỗi   Ex: VarName.Contains (“text”)   * **Format**: chuyển đổi giá trị của object sang strings ( và chèn chúng vào văn bản khác )   String.Format (“{0} is {1}”, VarNam1, VarName2)   * **IndexOf**: Trả về chỉ mục dựa trên vị trí trong một chuỗi   VarName1.IndexOf (“a”)   * **Join**: Nối các phần tử trong tập hợp và hiển thi chúng dưới dạng chuỗi   String.Join (“|”,CollVarName1)   * **Replace:** thay thế tất cả các chuỗi con trong một chuỗi   VarName.Replace(“original”,”replaced”)   * **Split:** Tách một chuỗi thành chuỗi con bằng dấu phân tách đã cho   VarName.Split(“|”c)(index)   * **Substring:** Trích xuất một chuỗi con từ chuỗi sử dụng chỉ số bắt đầu và độ dài của chuỗi   VarName1.Substring (startIndex,length)  vbCr để chuyển sang dòng tiếp theo |
| List | * Dùng để thu thập và lưu trữ số lượng hóa đơn đáp ứng các tiêu chí nhất định   Có 4 thao tác với list:   * Add to collection * Remove from Collection * Exists in Collection * Clear Collection * Tạo list: new List (of String) from {item,item,item}   Nếu không có item thì new List(of String)   * Nối list: Enumerable.Concat(list1.AsEnumerable,list2.AsEnumerable).ToList * Sắp xếp list: Invoke Method   Sử dụng for Each, chuyển đổi thành ProperCase sử dụng StrConv(item,VBStrConv.ProperCase) |
| Dictionaries | Các thao tác với Dictionaries   * Thêm và xóa mỗi key, value * Truy xuất giá trị được liên kết với 1 khóa * Chỉ định lại các giá trị mới cho khóa có sẵn   Chỉ định:   * Lưu trữ chi tiết cấu hình hoặc thông tin * Lưu trữ chức danh công việc hoặc các thông tin liên quan khác của nhân viên * Phương thức * Khởi tạo * Thêm: VarName(Key,Value) * Xóa: VarName.Remove(Key) * Truy xuất   VarName.Item(Key) trả về mục của key đó  VarName.Count trả về giá trị int32 của số của mục distionary  VarName.ContainsKey(Key): Kiểm tra xem mục có trong distionary hay không và trả về true/false  VarName.TryGetValue(Key,Value): Kiểm tra xem mục có trong distionaries hay không và trả về giá trị true/false kèm theo mục tìm thấy |
| Regex | * Các ứng dụng điển hình của RegEx:   + Xác thực đầu vào  + Phân tích cú pháp chuỗi  + Gom dữ liệu  + Thao tác chuỗi   * Truy xuất các văn bản theo một chuẩn nhất định:   + Truy xuất số điện thoại với các số đầu cố định  + Thu thập tên con đường từ văn bản lớn   * Phương thức:   + **Matches**: Tìm kiếm một chuỗi đầu vào cho tất cả các lần xuất hiện và trả về tất cả các kết quả phù hợp thành công  + **IsMatch**: Cho biết liệu biểu thức chính quy được chỉ định có tìm thấy khớp trong chuỗi đầu vào được chỉ định hay không  + **Replace**: Thay thế các chuỗi khớp với một mẫu biểu thức chính quy bằng một chuỗi thay thế được chỉ định |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Tổng kết**:

……………...…………………………...…………………………...…………………………...…………………………...…………………………...…………………………...…………………………...…………………………...…………………………...…………………………...………………………..