**Evaluation Only. Created with Aspose.Words. Copyright 2003-2021 Aspose Pty Ltd.**

**Regex**

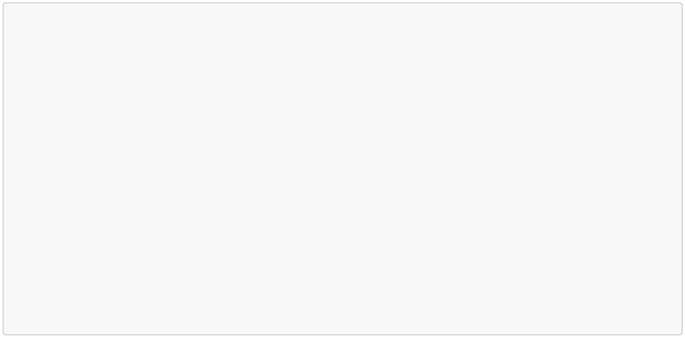


La clase Regex representa el motor de expresiones regulares del .NET. Se puede usar para analizar rápidamente grandes cantidades de texto, buscar patrones de caracteres específicos; para extraer, modificar, reemplazar o eliminar subcadenas de texto; y para agregar las cadenas extraídas a una colección para generar un informe.

Métodos habituales de Regex

* IsMatch : Devuelve True o False si encuentra o no el dato, respectivamente
  + Match : Devuelve el primer resultado encontrado que concuerde con el filtro
* Matches : Devuelve una lista de MatchCollection con todas las coincidencias
  + Replace : Sobre escribe el texto que coincida con el filtro que le hemos mandado
* Split : Crea un array sobre la cadena pasada separando en 1 elemento a medida que coincida con el filtro, igual que en String.Split(), pero mas completo

//Filtro, tiene que contener el ¬V1 o 2 o 3 ...



public const string REGEX\_CUERPO\_VARIABLES = "¬V\\d+";// Tiene que tener un decimal

private string SustituirTokens(string cuerpo, List<string> parametros) { // Declaro el objeto y le indico que tiene que actuar sobre ese filtro Regex regex = new Regex(REGEX\_CUERPO\_VARIABLES);

foreach (var p in parametros)

// Leo una lista de parametros en los cuales tengo que sustituir // el filtro indicado por dichos parametros

cuerpo = regex.Replace(cuerpo, p, 1);

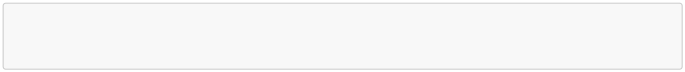
return cuerpo; }

En el ejemplo de arriba:

* cuerpo = una cadena que es la que tiene que tratar y sustituir lo que le corresponda
  + p es el parámetro a sustituir
* 1 es el numero de veces que hay que sustituir ese parámetro

Un ejemplo para encontrar coincidencias en cadenas con expresiones regulares.

Regex.IsMatch(cadena, filtro);



* cadena : La cadena en la que vamos a intentar encontrar coincidencias
  + filtro : Aqui se añade una cadena con las expresiones regulares para realizar el filtro.

Comandos Regex de expresiones regulares

* [] : Aquí van los caracteres que han de coincidir con el texto
  + {} : Se especifíca el número de caracteres máximo al que se va a aplicar el filtro
* () : Se usa para agrupar caracteres
  + ^ : Se usa para marcar el comienzo de un patrón.
    - Por ejemplo. Si ponemos como filtro @"^xyz" y mandamos la cadena "xyz123", como empieza por xyz, devolvería true
* $ : Se usa para marcar el final de un patrón.
  + - * Por ejemplo. Si se pone @"123$" y mandamos la cadena "xyz123", como termina por 123, devolvería true también

Expresiones regulares de ejemplo

* @"[8][0-9]" : Filtra por un carácter que empieza por 8 y otro que vaya del 0 al 9
  + @"[a-zA-Z][0-9]" : El primer carácter empieza por un rango de la a-z o A-Z y otro, un rango del 0-9.
* @"[a-z]{10}" : Busca un texto de a-z con 10 de longitud

