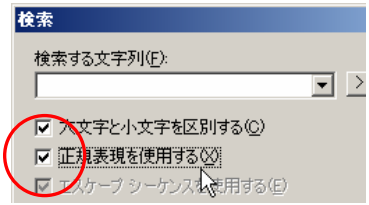
第 5 回：正規表現による検索･置換 (概説編)

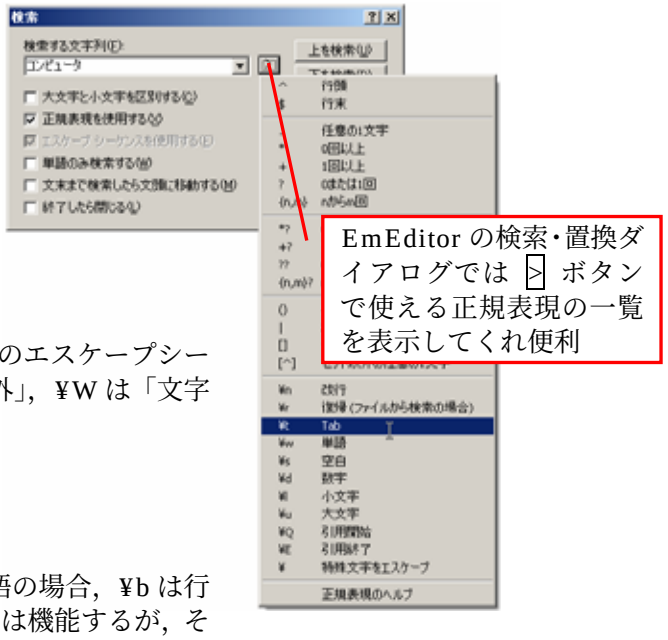
1. 正規表現とは



- テキストを柔軟に検索するために, 特別な意味をもつ文字 (「メタ文字」 metacharacter) を用いて検索パターンを指定する方法。
- EmEditor など正規表現に対応したテキストエディタでは, 検索・置換に正規表現を活用することで, 複雑な検索・置換をおこなうことができる。
- **正規表現が利用可能なソフトウエア：**
  - **テキストエディタ** EmEditor, サクラエディタ, K2 エディタ, 秀丸エディタなど多数 (検索・置換機能)
  - **テキスト検索ツール** grep, kwic 用の各種ツール (検索)
  - **プログラム言語・スクリプト言語** sed, awk, perl, ruby, python, Tcl/Tk, JAVA, Visual Basic, C# 等 (より柔軟なテキストの加工)
  - **その他の各種ソフトウエア** 表計算ソフト ListPad やホームページ作成ソフト Dreamweaver ワープロソフト Word (←実装方法は非常に特殊) など

- ソフトウエアの中で正規表現を処理する部分を「正規表現エンジン」と呼ぶ。アプリケーションがサポートする正規表現の内容は, 正規表現エンジンによって異なる。正規表現の開発の歴史は長く, 正規表現エンジンには独自の改良 (「訛り」) や記述方法のゆれ (「方言」) がある。
- 前回紹介した「ワイルドカード」との違いに注意：* と ? は正規表現では意味が異なる。

2. 正規表現ミニミニツアー



- よく使われるメタ文字
  - 改行： ¥n ないし ¥r¥n (下記 §3. 参照)
  - タブ： ¥t
  - 数字 (0-9)： ¥d [1]
  - スペース類 (スペース, タブなど)： ¥s
  - 文字類 (アルファベットと数字)： ¥w
  - 任意の 1 文字： . (ピリオド)

  ※ ¥の後はすべて小文字で記述すること。大文字のエスケープシーケンス ¥D は「数字以外」, ¥S は「スペース以外」, ¥W は「文字類以外」をそれぞれ表すので注意。

- 特定の区切りを表すメタ文字
  - 行頭： ^ (サーカムフレックス circumflex)
  - 行末： $ (ドル記号)
  - 単語の区切り： ¥b (単語を区切らない日本語の場合, ¥b は行頭や行末, 句読点やハイフン等が入る箇所では機能するが, それ以外の単語の区切りのマッチには使えない。)

[1] EmEditor では全角の数字も¥d に含まれる。半角数字のみにマッチさせる場合は, 文字の範囲を指定する [0-9] (下記参照) などを用いるとよい。

練習１：以下のパターンにマッチする正規表現を記述しなさい。

(1) 行頭の一字下がり	
(2) un で始まる英単語 (検索するのは un の部分だけでＯＫ。run などが検索されないようにすること。)	
(3) less で終わる英単語 (検索するのは less の部分だけでＯＫ。lesson などが検索されないようにすること。)	
(4) 郵便番号 (半角数字と半角スペースを使ったものだけを検索できればＯＫ。)	

練習２：以下のパターンが¥bby¥b **および**¥sby¥s **にマッチするかどうか調べなさい。**

	¥bby¥b	¥sby¥s
(5) 行頭の by		
(6) 行末の by		
(7) by-product		
(8) step-by-step		
(9) one by one		
(10) 以下の文:「～までに」を表す英語は by である。(by の両側にはスペースを入れない)		

- 選択性のあるグループの指定
  - いずれかの文字：四角括弧 [ ] で囲む： [abc] … a か b か c
  - 文字コード順の文字範囲：最初と最後の文字をハイフンで結ぶ： [0-9] … 0 から 9 まで； [1-35] … 1, 2, 3, 5 のいずれか； [a-zA-Z¥-] … 大文字小文字のアルファベット，またはハイフン- (文字としてハイフンを指定する場合には，文字範囲を表す記号と混同されないよう，¥- と記述する。)
  - 指定した文字以外：文字のリストを [^ と] で囲む： [^abc] … a, b, c 以外の 1 字
  - 複数の文字からなるパターンのいずれか：縦棒 (パイプ) | で区切る： one|two|three … one か two か three
  - 検索したい文字列の一部について，パターンを複数列挙する場合，縦棒 | と括弧を使い **(パターン A|パターン B)** のように表す： propert(y|ies) = property, properties
- パターンの連続を表す ？ ＊ ＋
  - パターン 0 回か 1 回： ?
  - パターン 0 回以上： * (アスタリスク)
  - パターン 1 回以上： +

※ 複数の文字からなるパターンの連続は ( ) を使って表す： (pa)+ = pa, papa, papapa…
※ 繰り返しの回数を指定するには中括弧を使う：{ 3 } 3 回以上，{ 2 , 5 } 2 回以上 5 回以下
※ Word の検索・置換機能のメタ文字は，正規表現とは全く異なる。* と ? の意味も異なる (「ワイルドカード」という記法で，* は 0 文字以上を，? は任意の 1 文字を表す)。
※ 任意の長さの文字列：.* (0 文字以上) や .+ (1 文字以上) …「最長一致」する
.*? (0 文字以上) や .+? (1 文字以上) …「最短一致」する (cf. §4)

担当：千葉庄寿 (CHIBA Shoju) e-mail: schiba@reitaku-u.ac.jp

練習３：以下のパターンにマッチする正規表現を記述しなさい。

(5) 英語の冠詞 a, an, the	
(8) 「コンピュータ」と「コンピューター」	
(9) 「短い」と「短かい」と「みじかい」	
(6) 3 桁以上 6 桁以下の数字 (単語として)	
(7) 英語の動詞 make の変化形	
(8) 「ぽたぽた」のような繰り返し表現	
(9) 漢字+する	
(9) as + **任意の 1 単語** + as という英語の表現	
(10) 「～していく」の変化形	

- 正規表現で特別な意味を持つ文字： ^ $. { } [ ] ( ) | * + ? ¥

  ※ 正規表現の中でこれらの文字を「文字として」検索したい場合には,「¥」記号 [2] をつけて表す (「文字リテラル」という)： ¥^ ¥$ ¥. ¥{ ¥} ¥[¥] ¥(¥) ¥| ¥* ¥+ ¥? ¥¥

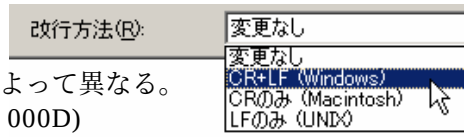
- 「前方参照」の ( ) ：丸括弧 ( ) つきの正規表現を使うと，( ) 内のパターンにマッチした文字列を記憶し，¥1, ¥2, ¥3 …という表記方法で括弧が現れた順番に参照することができる。1 つの正規表現の中で「前方参照」として使えるほか，EmEditor では置換の際，検索文字列や置換後の文字列の中で利用することができる。

  ※ 文字列の一部として，パターンを複数列挙する場合に丸括弧を使うが，その際一致を記憶させたくない場合には **(?:パターンA|パターンB)** と表す。この表記は少なくとも Perl, EmEditor で使えるが，この表記法をサポートしていないソフトウエアもある。

練習４：以下の置換をおこなう正規表現を記述しなさい。

	検索する文字列	置換後の文字列
(5) 「私」を検索し，両側にタブを挿入		
(6) 「私」「わたし」「わたくし」を検索し，両側にタブを挿入		
(7) 「ゝ」を直前の文字に置換		
(8)「ぽたぽた」のような繰り返し表現を検索し，両側にタブを挿入		

[2] エンコード方式やフォントによってはバックスラッシュ (\) など他の記号で表示される場合もある。



3. 改行を表すメタ文字

- テキスト文書の改行記号は，歴史的な理由により OS によって異なる。
  - CR = carriage return (ASCII コード 0D, Unicode 000D)
  - LF = line feed (ASCII コード 0A, Unicode 000A)
    - Windows = CR + LF
    - Macintosh = CR
    - Unix = LF
- 多くの場合，正規表現での改行記号の表現はエスケープシーケンスと同じ ¥n であり，この表現を使えば CR+LF でも CR でも LF でも検索できる。ただし，以下のような例外も。
- **【注意】**EmEditor 4 Professional の「ファイルから検索」で正規表現を使用する場合 CR を ¥r, LF を ¥n と別々に表現するので，正規表現を使って改行を指定する場合，実際のテキストの改行方法にあわせて改行記号を指定する。(「正規表現を使用する」をチェックせず,「エスケープシーケンスを使用する」のみチェックしている場合には，改行記号は¥n でも ¥r¥n でも同じ。) [3]

4. 正規表現利用のコツ

- 正規表現は，パターンを表すブロックの組み合わせによって複雑なパターンを表現できる。ブロック自体は単純であるが，組み合わせ方法は無限にあり，目的にあわせてブロックの組み立てかたを工夫する必要がある。
- 正規表現には「方言」や「訛り」があり，ソフトウエアやプログラム言語によって実装内容が大きく異なる。¥b, ¥w, ¥s, Unicode 文字の表記法など, 確認しておこなう必要がある。
  - 特定の文字エンコード方式にしか対応していない正規表現エンジンがある。例えば，半角英数字以外の文字を文字として正しく処理できないものは [あいうえお] のような使い方ができない (複数の文字列からなるパターンとして (あ|い|う|え|お) と代用記述することで，ある程度使うことができる場合もある)。
  - EmEditor の正規表現は, Perl というプログラミング言語の正規表現に基づいており，今回紹介していない，より高度な正規表現も使用できる。詳しくはヘルプ「正規表現構文」を参照するとよい。

- 「最長一致」の原則：「.*」や「.+」で任意の長さのテキストを表すことができる。しかし，このような表現は検索に時間が長くかかるばかりか，できるだけ長く一致しようとする特性のため，意図していないパターンを検索してしまうことがある。

  誤： ¥b.*s¥b (任意の長さで最後が s の単語)：予定外の場所まで一致してしまう
  正： ¥b¥w*s¥b または ¥b.*?s¥b (.*? で「最短一致」を表す)：正しく単語ごとに一致する。最短一致の *? が使えない場合には，代用表現として「スペース以外の文字の連続」を表す [^␣]* (␣はスペースを表す)を使い，[^␣]*s¥b などと記述しても同様の効果がある。

5. 参考文献：正規表現について

- IDEA・C (2001) 『正規表現の達人』 ソフトバンク.
- 大名力「正規表現によるテキスト検索」オンライン資料 URL: http://infosys.gsid.nagoya-u.ac.jp/~ohna/re/index.html
- 中尾浩 他 (2002) 『コーパス言語学の技法Ⅰ：テキスト処理入門』夏目書房
  同 (2004) 『コーパス言語学の技法Ⅱ：言語データの収集とコーパスの構築』夏目書房
- Friedl, Jeffrey E. F. (歌代和正監訳, 2003) 『詳説正規表現』第２版 オライリー・ジャパン

※ プログラミング言語 Perl の入門書には，たいてい正規表現の詳しい解説が含まれる。

---

[3] この動作は，旧バージョン (EmEditor 3) と異なるので注意。