

# DBPublisher/i

---

Database Publishing Plug-ins for InDesign

**活用ガイド**

2010年 3月 1日 第1版発行

## 発行

---

株式会社リンクス

## 著作

---

正規表現の手引き： 基礎編、実例編本文及びサンプル原案、付録：市川せうぞー  
実例編サンプルデータ：株式会社リンクス

## 商標

---

DBPublisher は株式会社リンクスの登録商標です。

Adobe および InDesign は、Adobe Systems Incorporated（アドビシステムズ社）の米国およびその他の国における登録商標または商標です。

その他の製品名は、各社の商標または登録商標です。

# 正規表現の手引き

---

市川せうぞー

## はじめに

読者の中には「正規表現」という言葉を初めて聞く方もいらっしゃるかもしれません。正規表現はとても便利なテキスト検索方法の一種です。「正規表現は難しい」などと言う人がいますがそれは誤解です。基礎をきちんと理解すれば、決して難しいものではありません。この小冊子は、正規表現を初めて学ぶ人のために書きました。

この小冊子は前半と後半に大きく分かれています。まず、前半の基礎編で正規表現の基本的な考え方について学びます。次に実例編でいくつかの作例を見ながら正規表現を学習できます。実例編はチュートリアルとして実際にやってみると理解が深まるでしょう。巻末には参考書籍や主な正規表現一覧を掲載してあります。

筆者と正規表現の付き合いはもう20年近くになります。その間に使っているコンピュータもアプリケーションもまったく変わってしまいました。正規表現は息の長いスキルです。正規表現のパワフルな魅力をひとりでも多くの方に知っていただきたいと思います。

## 正規表現とは何か？

「はじめに」で述べた通り、正規表現はテキスト検索方法の一種です。通常のテキスト検索にくらべて強力なテキスト検索方法が可能です。多くのテキストエディタをはじめ、たくさんアプリケーションで使えます。InDesignではCS3から「GREP」という名前で実装されました（InDesign CS4では「正規表現」に改正）。DBPublisher/iではバージョン1.5のテキスト変換機能で正規表現が使えるようになりました。

正規表現は **regex**（regular expressionの略）と呼ばれることもあります。しかし、ワイルドカード検索やあいまい検索などとは異なることに注意してください。正規表現の大きな特徴は「**メタ文字**」と呼ばれる記号を使って、文字を集合的に扱えることです。

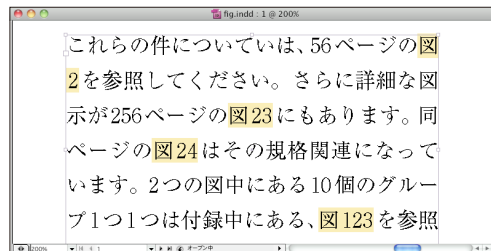
## 正規表現で何ができる？

例えば、本文中の「図1」から「図9」を検索したいとしましょう。通常のテキスト検索ですと「図1」を検索して、次に「図2」を検索して、さらに「図3」を検索して……、合計9回も検索しなくてはなりません。「図1」から「図99」までならどうでしょう？ それだけで多くの時間がかかってしまいます。

こんな時、正規表現を使えばたった1回の検索で「図1」から「図99」までを検索できます。

```
図\d+
```

たったこれだけで図番号を持つテキストすべてが検索できます。これが正規表現です。通常のテキスト検索が「1対1」検索なのに比べて、正規表現検索は「**1対多**」検索が可能です。



## メタ文字とは？

先ほどの検索クエリは「\d+」という見慣れない書き方をしています。これが**メタ文字**です。すなわち、「\d」が0から9までの半角数字を表し、「+」記号がその1つ以上の連続を表しています。ですから、「\d+」は数字の連続を意味しており、文字列「1」にも「50」にも「999」にもマッチします。

メタ文字にはたくさんの種類があります\*1。また、アプリケーションによって使える正規表現の種類が少し違います。こうしたことが、正規表現を難しく感じさせているようです。

しかし、正規表現を使うのにメタ文字をすべて覚える必要はありません。必要に応じてリファレンスを参照しながら、ひとつづつ身につけていけばよいからです。正規表現のメタ文字は大別すると以下の4種類に分類できます。

- 文字を表すメタ文字
- 位置を表すメタ文字
- 繰り返しを表すメタ文字
- その他のメタ文字

これからこの4つの種類の代表的なメタ文字について学習しましょう。

## メタ文字を攻略する

### 文字を表すメタ文字

最初にご紹介するメタ文字は「.」（ドット）です。これは**任意の1文字**を表します。どんな文字にでもマッチするという意味です。例えば、

お.さん

と検索すると、「お父さん」にも「お母さん」にも「お姉さん」にも「おじさん」にもマッチします。

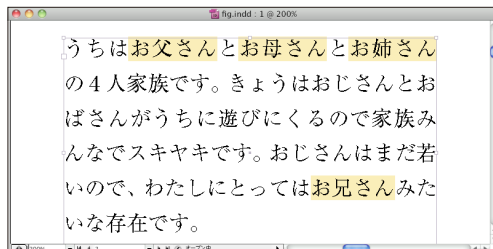
しかし、「.」はあまりにも広くマッチしすぎてしまいますので、「えがおでさんぽ」の「おでさん」にもマッチしてしまうでしょう。この場合、家族にだけ正しくマッチさせるには

お[父母姉兄]さん

とする必要があります。「[ ]」（ブラケット）に囲まれた文字にだけマッチさせます。これを「**文**

\*1 InDesign CS3 で使用できる正規表現一覧は [http://www.seuzo.jp/st/Other/InDesign\\_regex.html](http://www.seuzo.jp/st/Other/InDesign_regex.html) を参照してください。

文字クラス」といいます。「[〇一二三四五六七八九]」と書けば、漢数字にだけマッチします。



文字クラスでは、連続した文字コードの範囲を表すことができます。たとえば、数字の0から9までは文字コードが連続していますから「[0-9]」と書くことができます。文字コードの最初の文字と最後の文字を「-」（ハイフン）で繋げば、その文字コード範囲すべての文字を書いたことと同じになります。同様にアルファベットの大文字は「[A-Z]」と書けますし、ひらがなは「[あ-ん]」と表現できます。ちなみに、ひらがな以外を検索したいときは、文字クラスの最初に「^」（キャレット）を付けて「[^あ-ん]」と表現できます。これを「否定文字クラス」といいます。

文字クラスでコード範囲を指定する際、注意することがひとつあります。正規表現を使うアプリケーションによって、どの文字コードを使うかが違ってきます。たとえば、InDesignではユニコードを採用していますから、一般的な漢字の範囲は「[一-龠]」になります。シフトJISを採用しているアプリケーションなら、「[亜-熙]」と書かなければなりません。文字クラスについて、もうちょっと詳しく知りたい方は<http://d.hatena.ne.jp/seuzo/20090309/1236525090>を参照してください。

文字を表すメタ文字の中で、ほかによく使うメタ文字として、「\d」（数字の0～9を表す）、「\t」（タブ文字を表す）、「\r」（改行文字を表す、\nを使うアプリケーションもある。InDesignでは\nは強制改行を表す）などがあります。ひとつづつ覚えていきましょう。

## 位置を表すメタ文字

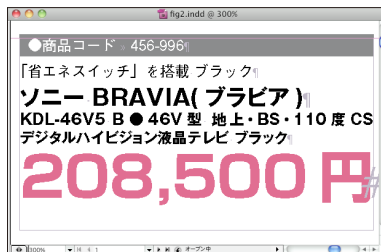
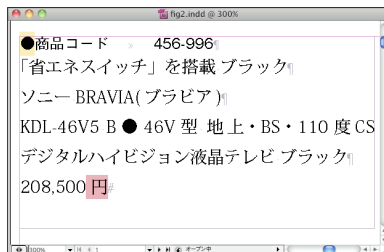
位置を表すメタ文字の代表は「^」（キャレット）と「\$」（ドルマーク）です。「^」は行頭を表し、「\$」は行末を表しています。例えば、カタログの製品番号のテキストが「●」で始まっているとしたら

^●

と書けば、製品番号のテキストがどこにあるのか検索できます。同様に、価格テキストの行末が「円」で終わっていれば

円\$

と書けば、価格を検索できるでしょう。



ほかにも、英単語に正しくマッチさせるために単語の境界を表す「\b」も位置を表すメタ文字です。例えば、「rice rises more than thrice in price」という英文中で正しく「rice」という単語にだけマッチさせるには

```
\brice\b
```

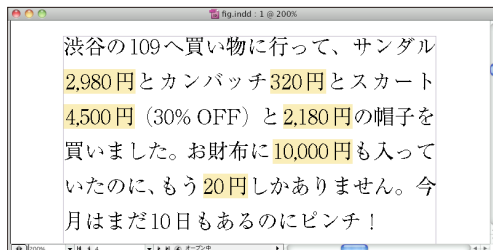
と検索してください。これを応用して、「rice\b」とだけ書けば、後方一致で検索することも可能です。

## 繰り返しを表すメタ文字

例えば、「198円」や「21,000円」のような価格を検索したい時、桁数によって何度も検索するのは面倒です。数字と位取りのカンマが何桁かあって、最後に「円」が付くテキストを検索したい場合は

```
[\d,]+円
```

と検索すれば、すべての価格テキストにマッチします。これは、「+」（プラス）が直前の文字が1回以上繰り返しを表しているからです。「キター+!!」と正規表現を書けば、「キター!!」にも「キター————!!」にもマッチします。



次にご紹介するのは、直前の文字が0回以上繰り返しているのを表す「\*」（アスタリスク）です。「0回も1回も変わらないんじゃないの？」と思うかもしれません。例えば、データベースなどではフィールドが空欄の場合もあります。「ペット:猫と犬」と書かれているフィールドもあれば、「ペッ

ト：」と空欄になっているフィールドもあるかもしれません。こうした場合にも

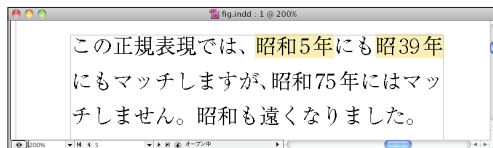
```
ペット：.*
```

と検索すれば、どちらのテキストにもマッチします。

例えば、「昭和20年」といった年号と「昭20年」などと略書きされた年号が混在する時、どちらも検索したいとします。この時「昭和\*\d+年」と書くのは感心しません。「\*」や「+」は欲張りにマッチしすぎる傾向がありますから、知らず知らずのうちに誤検索の原因になります。直前の文字が0回または1回しかない時は「?」（クエスチョンマーク）を使って

```
昭和?[1-6]?\d年
```

と書くようにすると、より正確に検索できます。

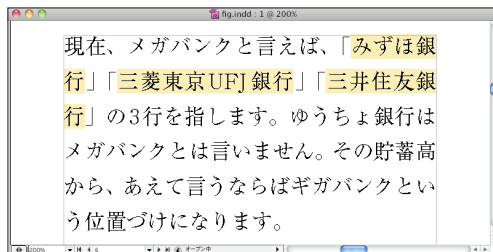


## その他のメタ文字

正規表現にはいくつかの選択肢にマッチする書き方があります。例えば、「みずほ銀行」「三菱東京UFJ銀行」「三井住友銀行」のいずれかにマッチさせたい場合は「|」（パイプライン：縦棒）を使って

```
(みずほ|三菱東京UFJ|三井住友)銀行
```

と検索します。「( )」（パーレン）でグループ化して、「|」で区切られた選択肢を持ったテキストにマッチします。



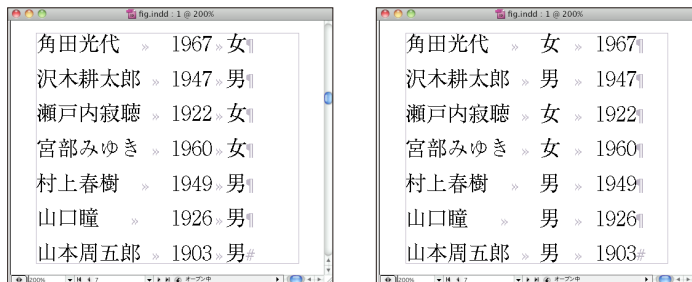
「( )」を使ったグループ化には、さらに面白い使い方があります。「( )」内にマッチしたテキストは、テキスト置換時に再利用できます。例えば、タブ区切りテキストのカラム順を入れ替えた時などにも使用できます。「市川せうぞー <tab>1964<tab>男」の2カラム目と3カラム目を入れ替えたいなら

```
^(^[\t\r\n]*)\t([\t\r\n]*)\t([\t\r\n]*)
```

と検索して、置換時に

```
$1\t$3\t$2
```

とすると、最初のテキストが「市川せうぞー <tab>男<tab>1964」となります。パーレン内にマッチしたグループは、左から「\$1」「\$2」「\$3」として利用できます。これを後方参照といいます。ちなみに「([\t\r\n]\*)」は「タブと改行文字以外が0文字以上のグループ」を意味しており、タブ区切りテキストを扱う時の常套句として頻繁に利用しますのでそのまま覚えてください。



正規表現には、いくつかのオプションを付けることができます。代表的なオプションは「(?i)」で、これはアルファベットの大きい文字と小さい文字を同一視するオプションです。例えば、

```
(?i)mac
```

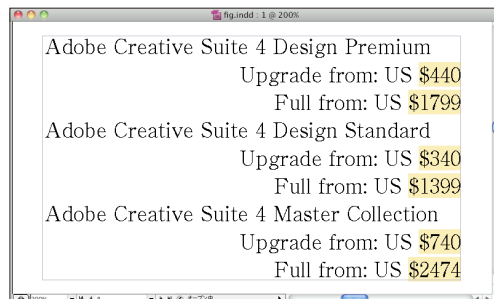
と書けば、「mac」にも「Mac」や「MAC」にもマッチします。大文字と小文字を区別していないのがわかります。

## メタ文字をそのものを表すには

メタ文字はテキストで書かれています。ですから、「This price is \$20.」のように通常のテキストとして出現する可能性があります。この場合、価格部分を検索するために「\$\d+」と書いても実際にはマッチしません。メタ文字をそのものを表すには、メタ文字の前に「\」（バックスラッシュ）を置いて

```
\$\\d+
```

と書く必要があります。「\\」そのものを表すためには「\\\'と書かなければなりません。これをメタ文字のエスケープと呼びます。



## 陥りやすい罠

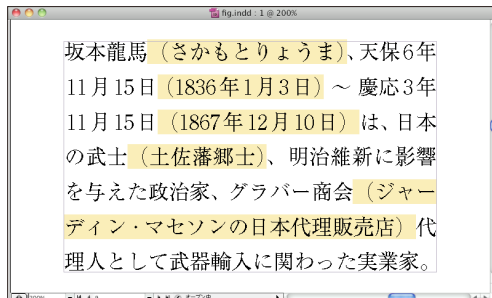
正規表現には陥りやすい罠がいくつかあります。たとえば、テキスト中の「(~)」で囲まれた部分を検索したい時、

```
(.+)
```

と書いてしまいたくなるかもしれません。確かに、1行の中に「(~)」で囲まれた部分が1カ所しか出現しない場合は、うまく検索できているように見えます。しかし、「今日(2/22)は猫の日(にゃんにゃんデー)です」のように括弧が2回以上出現する場合は、「(2/22)は猫の日(にゃんにゃんデー)」がマッチしてしまいます。これは「.+」が欲張りにマッチしてしまうのが原因です。これを**最左最長一致**といいます。これを回避するためには、

```
(.+?)
```

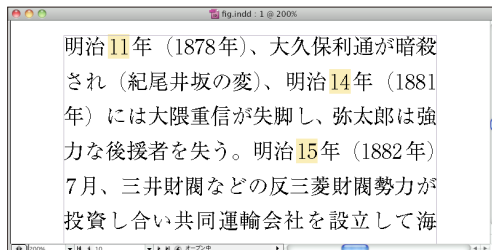
としてください。「+?」や「\*?」は**非欲張り型**の繰り返しを表します。



2桁の数字だけを検索したい時、繰り返しの回数を指定する正規表現を使って「\d{2}」などと書いてもあまり意味はありません。「123」という3桁数字のテキストの「12」部分にマッチが成功してしまうからです。2桁数字にだけマッチさせるには、**後読み**と**先読み**を使用して

```
(?<![\d,\.\.])\d\d(?![\d,\.\.])
```

と書かなければなりません。



## 正規表現の考え方

いままでいくつかの正規表現の種類と用法を見てきました。リファレンスに比べてまだまだ数が少ないと思われた方もいらっしゃるかもしれませんが、しかし、普段の仕事で使う正規表現では、これだけのメタ文字が使えるればほぼ90%以上を学習したことになります。

では、メタ文字を完全に使えるれば正規表現が使えるのでしょうか？ メタ文字さえ理解すれば正規表現が使えると考えている初心者は、なかなか正規表現の使い方が上達しません。それは木を見て森を見ていないからです。例えば、東京の電話番号を検索したい時、「03-\d{4}-\d{4}」と書いたとします。たしかに、この正規表現は東京の電話番号にもマッチするでしょう。しかし、こんなテキストにもマッチします。「会員番号：0003-1234-567890」。これは電話番号ではありません。電話番号の前にコロンやスペースで区切って「電話」や「TEL」が付いているのに気がつくかどうか大きなポイントです。

```
(電話|TEL\.?)[: ■ ■]?03-\d{4}-\d{4}(?!\\d)
```

ここまで書いたならば、電話番号でないものにマッチする可能性はほぼないでしょう。検索対象のテキスト全体を見渡し、自分の書いた正規表現がどんなテキストにマッチする可能性があるのか？ どんなテキストにマッチしないのかを想像し、見極める力が必要です。正規表現の最も重要な点は**テキストのモデル化**です。メタ文字をあれこれ選ぶ前に、検索したいテキストがどんな形をしているか、どんなパターンなのかを知る必要があります。「**包丁を選ぶ前に材料を見よ!**」ってことです。よいモデルは正しい検索と置換で大きな力を与えてくれますが、悪いモデルは誤検索と誤置換の迷路に迷い込んでしまいます。

## InDesignの正規表現とDBPublisher/iとの違い

正規表現が使えるアプリケーションはたくさんあります。アプリケーションはそれぞれ自分用の正規表現エンジンを持っており、正規表現検索を実現しています。DBPublisher/iはInDesignと同じ正規表現エンジンを使用していますが、一部動作が異なります。ですから、InDesign上で普段お使いの正規表現であってもDBPublisher/iでの検索結果や置換結果は異なる場合があります。例えば、正規表現ダイアログ中の「カナを区別」と「全角・半角を区別」などのオプションの指定はできません。カナと全角・半角は区別されます。\\d、\\s、\\u、\\lなどは全角にもマッチします。[]や|を使用して文字クラスを明示的に指定すれば、これらの違いの影響を受けにくくなります。またその他にも若干の動作の違いがある可能性があります。

# 実例編

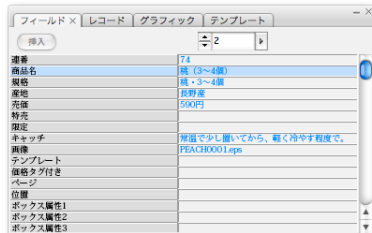
※DBPublisher/iの正規表現検索置換の操作方法については、ユーザーガイドの「テキスト変換」を参照してください。

## 括弧内の文字を小さくする

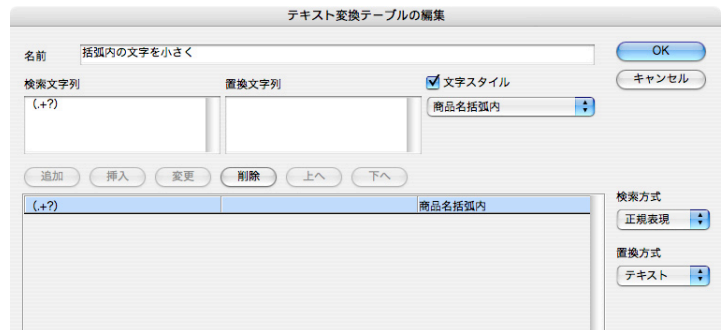
ひとつのフィールド内に「桃（3～4個）」のように、括弧内の文字だけ文字サイズの異なるテキストが混在している場合があります。このような時、わざわざフィールドを分けることなく「（3～4個）」の部分を正規表現でマッチさせて、特定の文字スタイルを適用してみましょう。使用する正規表現はこのようなになります（括弧は全角）。

(.+?)

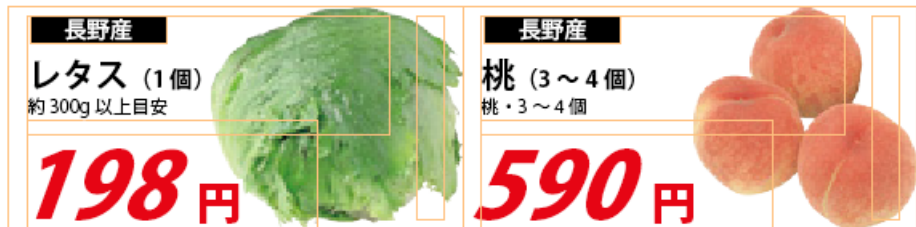
マッチした部分に文字スタイルを適用すればよいだけなので、「置換文字列」は空欄のままでも構いません。「文字スタイル」チェックボックスをオンにして、適用したい文字スタイルを選んでおきます。



▲DBPublisher/iの「フィールド」パネル。「商品名」フィールドに括弧で囲まれたテキストがある。



▲DBPublisher/iの「テキスト変換テーブルの編集」ダイアログ。正規表現にマッチしたテキストに対して「商品名括弧内」文字スタイルを適用する設定になっている。



▲組版結果。

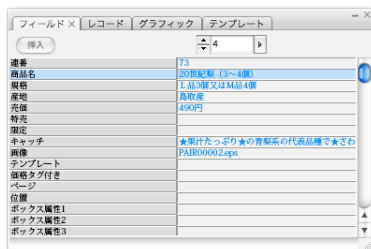
## ★～★に囲まれていたら太字にする

ひとつのフィールド内で太字にしたいテキストに対して、「★～★」でマーキングしてあるテキストがあるとします。前述の「括弧内文字を小さくする」の場合ではマッチしたすべてのテキストが対象でしたが、今回の場合は「★」は不要になります。このような場合、半角の「( )」でグループ化した部分だけを、後方参照の「\$1」を使って取り出すようにします。

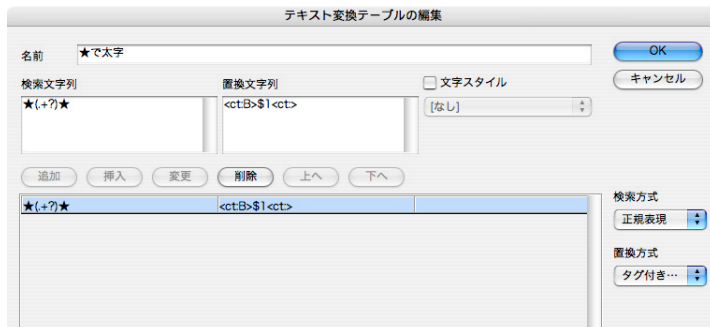
検索： ★(.+?)★

置換： <ct:B>\$1<ct:>

このとき、「置換方式」ポップアップメニューから「タグ付きテキスト」を選んでおくと、置換文字列の中にInDesignタグを含められます。InDesignタグとは、InDesignのテキスト属性をすべてタグとして表現する方法です（InDesignタグの詳細は、InDesignのインストーラに含まれる「タグ付きテキスト.pdf」を参照してください）。InDesignタグ「<ct:B>～<ct:>」は、フォントファミリーの「B」を適用するという意味です。



▲DBPublisher/iの「フィールド」パネル。「キャッチ」フィールドに★～★で囲まれたテキストがある。



▲DBPublisher/iの「テキスト変換テーブルの編集」ダイアログ。正規表現にマッチしたテキストをグループ化してタグを付与している。



▲組版結果。

## 価格だけに特定の文字スタイルを適用する

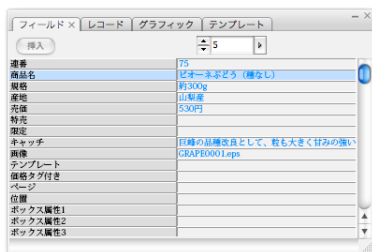
価格だけに特定の文字スタイルを適用するには、正規表現の先読みを使ってください。先読みを使うと、最後に「円」のついた数字にマッチしますが、「円」には文字スタイルが適用されません。

(`([\d,]+)(?=円)`)

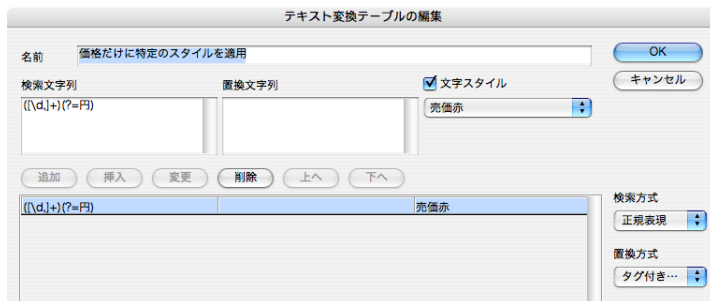
数字の最後の桁と「円」の間をカーニングで詰めたければ、InDesign タグと併用します。

検索：`([\d,]+)(\d)(?=円)`

置換：`$1<ck:-180>$2<ck:>`



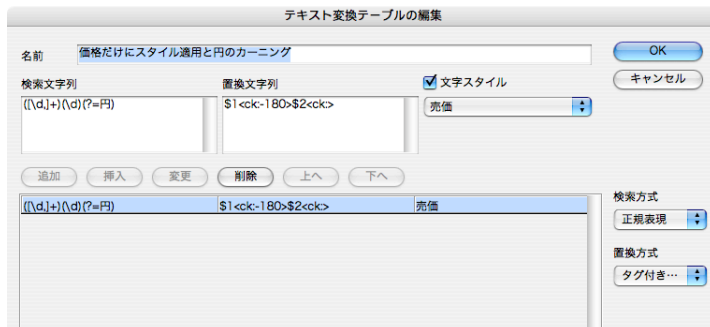
▲DBPublisher/iの「フィールド」パネル。



▲DBPublisher/iの「テキスト変換テーブルの編集」ダイアログ。先読みを使うことで、「円」には文字スタイルが適用されない。



▲組版結果。



▲InDesign タグと文字スタイルの併用。

## 1～4桁数字の字形変換

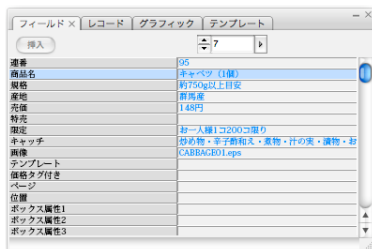
縦組みのテキスト中で、桁数に応じて数字の字形を変えたい場合、否定後読みと否定先読みを使用して検索し、InDesign タグで挟んで置換してください。たとえば、1桁の数字を全角字形にするには、下記のように設定してください。

検索： (?<![\d/.,]) \d (?![\d/.,])

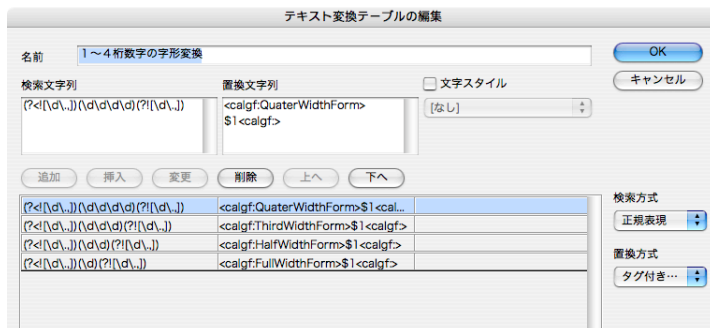
置換： <calgf:FullWidthForm>\$1<calgf:>

同様に2桁～4桁まで桁を増やすには、検索の中央部分にある「\d」を桁数分増やします。置換側では、2桁の半角字形が「<calgf:HalfWidthForm>\$1<calgf:>」、3桁の三分字形が「<calgf:ThirdWidthForm>\$1<calgf:>」、4桁の四分字形が「<calgf:QuaterWidthForm>\$1<calgf:>」となります。

DBPublisher/i では、それぞれのテキストフィールドに対して、このように複数回の検索置換を行います。



▲DBPublisher/iの「フィールド」パネル。「限定」フィールドに1桁～4桁の数字がある。



▲DBPublisher/iの「テキスト変換テーブルの編集」ダイアログ。4つの変換テーブルを上から順に実行する。



▲組版結果。

## ルビ処理

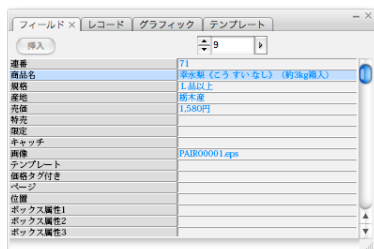
テキストにルビを振りたい場合も InDesign タグが使えます。青空文庫のルビ形式\*1で、テキストが「親文字《おや も じ》」となっていれば、下記のように検索置換してください（|は全角）。

検索： |?([一-龠]+)《(.+?)》

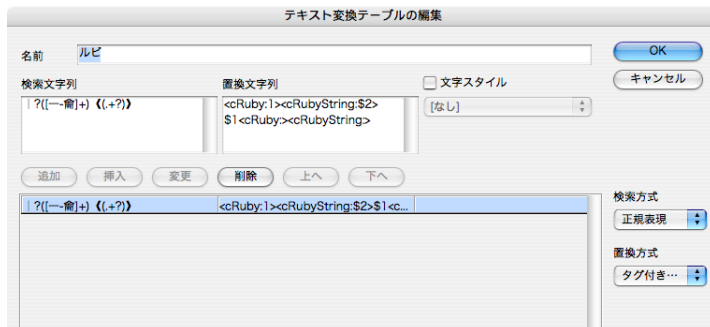
置換： <cRuby:1><cRubyString:\$2>\$1<cRuby:><cRubyString:>

InDesign 上でルビを入力するのと同様、ルビをスペースで区切っておけば、モノルビとして正しく親文字の上にルビが振られます。「親文字《おやもじ》」ではなく、「親文字《おや も じ》」としてください。

漢字の途中からルビを振りたい場合、ルビ開始の合図として全角の「|」を挿入し「親|文字《も じ》」とすれば、「親文字」のようにルビが振られます。



▲DBPublisher/iの「フィールド」パネル。「商品名」フィールドにルビが入力されている。



▲DBPublisher/iの「テキスト変換テーブルの編集」ダイアログ。InDesign タグを使用してルビを実現する。



▲組版結果。

\* 1 青空文庫のルビ形式については、[http://www.aozora.gr.jp/KOSAKU/MANUAL\\_2.html#ruby](http://www.aozora.gr.jp/KOSAKU/MANUAL_2.html#ruby) を参照してください。本例は親文字が漢字の場合のみの対応となります。漢字以外にルビを振るケースに挑戦してみてください。

## 参考書籍

- 『詳説 正規表現 第3版』  
Jeffrey E.F. Friedl 著、株式会社ロングテール／長尾高弘 訳、オライリージャパン刊、2008年、ISBN-13: 978-4873113593
- 『正規表現の達人 第2版』  
IDEAC 著、ソフトバンククリエイティブ刊、2005年、ISBN-13: 978-4797330816
- 『入門 正規表現 ～検索・置換・テキスト処理に強くなる!』  
岩谷宏 著、技術評論社刊、2008年、ISBN-13: 978-4774134048

## 参考Webサイト

- 『regex - 名もないテクノ手』  
<http://d.hatena.ne.jp/seuzo/archive?word=%2A%5Bregex%5D>
- 『はじめての正規表現』  
<http://higashizm.sakura.ne.jp/reg/>
- 『正規表現メモ』  
<http://www.kt.rim.or.jp/~kbk/regex/regex.html>
- 『Regular Expression(Riue ちゃんの正規表現講座) - Index』  
<http://www.sixnine.net/regexp/>
- 『正規表現パズル』  
<http://oraclesqlpuzzle.hp.infoseek.co.jp/regex/>
- 『正規表現クイズ』  
<http://aastory.info/reg/>
- 『HiFi Regex Tester - Live JavaScript Regular Expression Tester』  
<http://www.gethifi.com/tools/regex>
- 『正規表現まとめサイト - エンタープライズ | マイコミジャーナル』  
<http://journal.mycom.co.jp/news/2009/06/05/046/index.html>

## 正規表現の主なメタ文字一覧

詳細な一覧は [http://www.seuzo.jp/st/Other/InDesign\\_regex.html](http://www.seuzo.jp/st/Other/InDesign_regex.html) を参照してください。

### ▼文字を表す

正規表現	意味	検索のみ	用例・備考
.	任意の1文字	●	改行文字を含まない
\t	タブ文字		
\r	改行文字		
\n	強制改行		InDesign 上での shift+Return と同じです。
\d	数字	●	半角数字と全角数字の両方にマッチします。半角数字のみにマッチさせたい場合は [0-9] とします。
\s	ホワイトスペース	●	[ <code> </code> <code>\t</code> <code>\r</code> <code>\n</code> ] と同じです。全角スペースを含みます。全角スペースを含めたくない場合は [ <code> </code> <code>\t</code> <code>\r</code> <code>\n</code> ] とします。EM スペースなどにはマッチしません。
\u	任意の大文字	●	半角と全角の両方の大文字にマッチします。半角のみにマッチさせたい場合は [A-Z] とします。
\l	任意の小文字	●	半角と全角の両方の小文字にマッチします。半角のみにマッチさせたい場合は [a-z] とします。
\\	(文字) バックスラッシュ		メタ文字をエスケープする時にバックスラッシュを使用します。一般的に正規表現では、文字前に置いた \ は次の文字を表します (\ を無視します)。「\あ」は「あ」を表します。
\x{hex-num}	コードポイント hex-num (16進数) の文字	●	文字「あ」は「\x{3042}」。ASCII 文字のみ \xnum でも表せます (文字「A」は「\x41」)

### ▼位置を表す (アンカー)、後読み／先読み

正規表現	意味	検索のみ	用例・備考
^	(位置) 段落の始まり	●	
\$	(位置) 段落の終わり	●	
\<	(位置) 単語の始まり	●	
\>	(位置) 単語の終わり	●	
\b	(位置) 単語の境界	●	
(?<= )	肯定後読み	●	たとえば、正規表現「(?<=東京都)港区」と書いた時、「東京都港区」の中の「港区」にはマッチするが、「名古屋市港区」にはマッチしません。

正規表現	意味	検索のみ	用例・備考
(?! )	否定後読み	●	たとえば、正規表現「(?<! 東京都) 港区」と書いた時、「東京都港区」にだけマッチしません。「名古屋市港区」や「大阪市港区」の中の「港区」にはマッチします。
(?= )	肯定先読み	●	たとえば、正規表現「東京都(?=港区)」と書いた時、「東京都港区」の中の「東京都」にはマッチするが、「東京都練馬区」や「東京都杉並区」にはマッチしません。
(?! )	否定先読み	●	たとえば、正規表現「東京都(?! 港区)」と書いた時、「東京都港区」にだけマッチしません。「東京都練馬区」や「東京都杉並区」の中の「東京都」にはマッチします。

## ▼繰り返し

正規表現	意味	検索のみ	用例・備考
?	直前の文字が0回または1回	●	たとえば、正規表現「https?://」と書いた時、「http://」と「https://」マッチします。また、正規表現「Mac(intosh)?」と書いた時、「Macintosh」と「Mac」にマッチします。
*	直前の文字が0回以上	●	たとえば、正規表現「\s*\$」と書いた時、空行とスペースのみが0個以上の行がマッチします。
+	直前の文字が1回以上	●	たとえば、正規表現「cool+!」と書いた時、「cool!」にも「coooooooooooooooooo!」にもマッチします。あまりクールな例ではありませんが……
*?	直前の文字が0回以上（最小一致）	●	たとえば、CSVの二重引用符（空要素もありうる）のような文字列「"hoge hoge"」に対して★マークを入れるような場合、検索フィールドに正規表現「"(. *?)"」と書き、置換フィールドに「"★\$1"」とすれば文字列の先頭に「★」が付与できます。
+?	直前の文字が1回以上（最小一致）	●	たとえば、文字列「<p><a href="http://www.seuzo.jp/">せうぞー</a></p>」に対して、正規表現「<. +?>」と書くと文字列すべてにマッチしてしまいます。最初のタグ「<p>」にマッチさせたい場合は「<[<?>+>」または「<. +?>」と書きましょう。非欲張りマッチ、最小マッチ。
{n}	直前の文字が少なくともn回	●	このとき、必ずしもn回かっきりでないことに注意。たとえば「123456」という数字に対して「\d{2}」とすると、最初の「12」にマッチします。すなわちこの場合、「\d{2}」が常に2桁の数値を表しているわけではありません。

正規表現	意味	検索のみ	用例・備考
{n,}	直前の文字がn回以上	●	
{n,m}	直前の文字がn回以上m回以内	●	必ずしもm回以内に納まっているものだけがマッチするわけではありません。直前の文字がm回以上あったとしても、少なくともm回まではマッチします。

## ▼文字クラス、グループ、選択

正規表現	意味	検索のみ	用例・備考
[ ]	文字クラス	●	たとえば、正規表現「[abc]」と書いた時、文字「a」または「b」または「c」にマッチします。同じ意味で「[a-c]」と範囲を指定できます（この時の範囲はUnicode順）。さらに「[\^abc]」は「abc」以外の文字にマッチします。この時、気をつけなければならないのは、[^\^abc]は改行文字も含んでしまうことです。「.」が改行文字を含まないことに慣れてしまうと、ついついやってしまいがちなミス。また漢字の範囲は[一-龠]、ひらがなは[あ-n]になります。
( )	グループマーキング	●	グループを使用する目的は主に2つあります。ひとつは検索文字列をキャプチャして、置換文字の「\$1～\$9」で参照させること。もうひとつは、選択「 」を使用するときを選択境界をはっきりさせることです。たとえば、タブ区切りの列の順序を入れ替えたい時、文字列「hoge<tab>fuga<tab>puyo」に対して、検索フィールドに正規表現「(.+?)\t(.+?)\t(.+?)\$」と書き、置換フィールドに「\$3\t\$2\t\$1」と書けば、結果は「puyo<tab>fuga<tab>hoge」となります。
(?: )	グループのキャプチャをしない	●	文字列をグループ化するとき、グループ化された文字列をバッファにキャプチャしません。たとえば、正規表現「(?:.+?)\t(?:.+?)\t(?:.+?)\$」と書いた時、3番目のグループは「\$2」で参照されます。
\$0	マッチした文字列すべて		置換フィールドでのみ使用可能です。グループ化の有無にかかわらず、マッチした文字列すべてを参照できます。
\$1～\$9	グループ化された文字列（左から1～9）		置換フィールドでのみ使用可能です。文字列「いろはにほへとちり」に対して、正規表現（検索フィールド）に「^(.)(.)(.)(.)(.)(.)(.)(.)(.)(.)(.)」と書き、置換フィールドに「\$9\$8\$7\$6\$5\$4\$3\$2\$1」と書けば、置換結果は「りちとへほにはろい」になります。番号は左から順に1～9番が振られ、グループが入れ子になっているときは起こし括弧の順となります。

正規表現	意味	検索のみ	用例・備考
\1～\9	検索フィールドでのグループの再利用	●	検索フィールド内で、正規表現「(hoge)\1」と書いた時、文字列「hogehoge」にマッチします。 \1はマッチ文字列そのものであり、グループ内に書かれた正規表現が再度展開するわけではありません。つまり、文字列「hoge1hoge2」に対して、正規表現「(hoge\d)\1」はマッチしません。この時、文字列「hoge1hoge1」にはマッチします。
	選択	●	たとえば、正規表現「hoge fuga」は、「hoge」と「fuga」にマッチします。1月～12月は「(1[0-2] [1-9])月」と表現できます。

## ▼オプション

正規表現	意味	検索のみ	用例・備考
(?i)	大文字と小文字を区別しない-オン	●	「(?i)abc」と書くと、「abc」にも「ABC」にも「aBc」にもマッチします。
(?m)	複数行-オン	●	デフォルト。フィールドテキスト内に改行文字(\r)または強制改行(\n)が含まれる場合に、2行目以降にもマッチします。「^」はフィールドの先頭および各行の「行頭」に、「\$」は各行の「行末」とフィールドの末尾にそれぞれマッチします。マルチラインモード。
(?-m)	複数行-オフ	●	フィールドテキスト内に改行文字(\r)または強制改行(\n)が含まれる場合に、2行目以降にはマッチしません。シングルラインモード。
(?s)	単一行-オン	●	「.」が改行文字(\r)と強制改行(\n)を含むようになります。「^.+」と書けば、改行を含むフィールドテキスト全体にマッチします。
(?-s)	単一行-オフ	●	デフォルト。「.」は改行文字(\r)と強制改行(\n)のいずれも含みません。
(?x)	スペース文字を無視-オン	●	正規表現中に書かれたスペース文字を無視します。正規表現「(?x)ab c」は文字列「abc」にマッチします。
(?-x)	スペース文字を無視-オフ	●	デフォルト。