

文字列連結をテンプレートへ置き換える

TakiLibで、報告文の形と埋め込む値を分ける

Copyright © 2026 LWP 山中 一弘

本資料は、出典を明記いただければ、商用・非商用を問わず、ご自由に複製・改変・再配布していただけます。なお、著作権表示は改変せず、そのまま記載してご利用くださいますようお願いいたします。

ストーリー

月次報告の文面が、今月も少し変わります。フナさんは数字と文章を照合し、シャケもんは連結の式を修正中。「件数を先に」「担当を先に」と変更が重なり、二人は文章の形と値を分ける方法を考えます。

TakiLib Ver 3.5.4を導入したWindows版Excel VBAを題材にした、説明用の物語です。図と掲載例の結果は原ソースの読解に基づきます。

1 文章を直すたび、連結が増える

①

文章を直すたび、連結が増える

シャケもん

また連結の式を直すのか…

フナさん

今月は、件数を先に書きたいです。

月次報告
連結処理
VBA

月次報告書

件数: 3
担当: フナ
期間: 2024年4月

今のコード（文章と値が混ざっている）

"件数=" & n & "、担当=" & person

固定の文章（変えない部分）
 変わる値（データで入れる部分）

表示されるレポート

件数= 3、担当= フナ

シャケもん

文章の形と、入れる値を分けてみよう。

少しずつ、わかりやすく。

フナさん

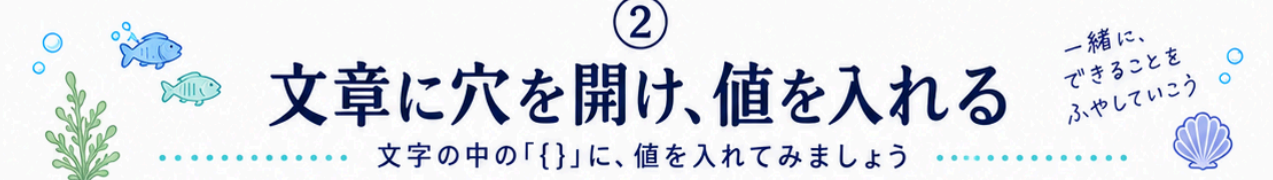
文面を見ながら確認できると助かります。

図1 | 図のnは3、personは「フナ」。文章の形が、値をつなぐ式の中へ入り込んでいる。

2 文章に穴を開け、値を入れる

② 文章に穴を開け、値を入れる

文字の中の「{}」に、値を入れてみましょう



一緒に、
できることを
ふやしていこう



textFormatで、
穴に値を入れるんだ。

テンプレート ({}が穴) 値 結果

件数={}

→

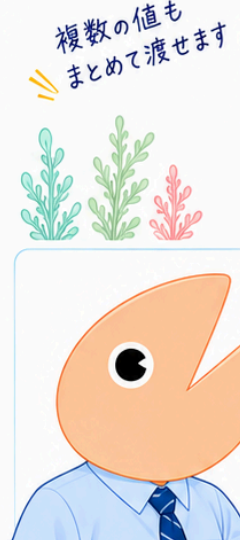
件数=3

使うコード

textFormat("件数={}", 3)



複数の値も
まとめて渡せます



文章の中に、数字の置き場所があるんですね。

値を配列で渡すなら

使うコード

textFormatPacked("件数={}", Array(3))





Packedは、値を配列で
まとめて渡す形だね。

文面の確認は、
私が引き受けます。



いっしょに
よりよいレポートを

..... 小さな工夫で、毎日の仕事をもっと快適に

図2 | textFormatは値を個別の引数で、textFormatPackedは配列で受け取る。

3 並び替えには、番号付きの穴

③

並び替えには、番号付きの穴

値の並びはそのまま、番号を指定できるよ。

担当を先にした版も必要です。

用意する値の対応表

位置	値
0	A
1	B

順番に使う

`{ } / { }` → `A / B`

左から順に
0 → 1 を使う

番号で選ぶ

`{1} / {0}` → `B / A`

指定した番号の値を使う
1 → 0 を使う

0 が A、1 が B。
文面側で選ぶんですね。

番号は 0 から。{ } は使うたびに次へ進む。

図3 | 値の並びを保ったまま、テンプレート側で使う位置を指定できる。

4 番号を使っても、順番は進まない

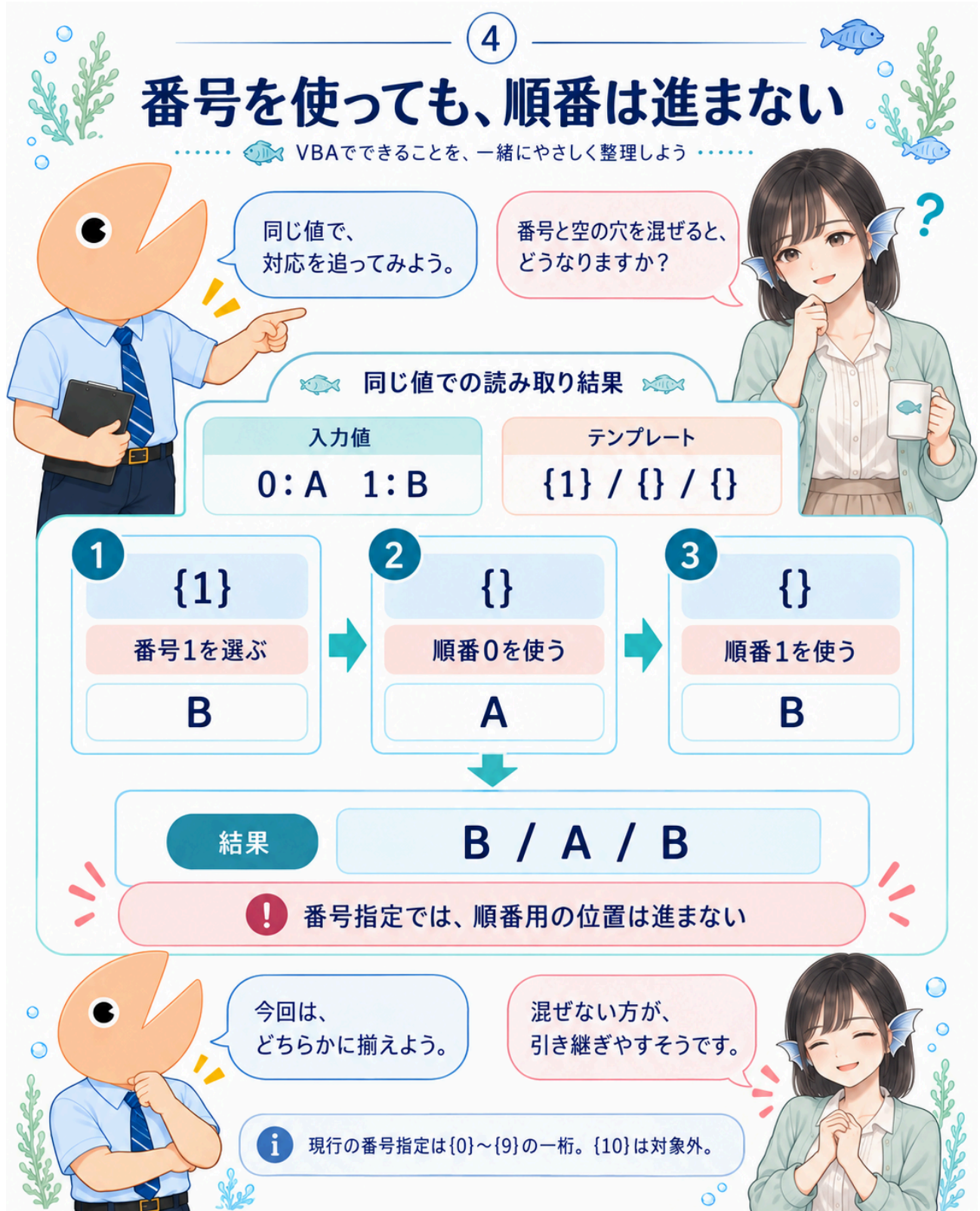


図4 | 混在は可能でも、順番の追跡が必要。番号指定は現行実装では一桁に限られる。

5 文字が返っても、成功とは限らない

⑤

文字が返っても、成功とは限らない

見た目が「文字」でも、処理が正しくできたとは限りません。
出力の内容をしっかりと確認することが大切です。

一つひとつ
確かめながら
進もう



穴が二つなのに、
値は一つだったね。

報告文に、
! #VALUE!が
入っています。



今回のケース

テンプレート
{} : {}

+

値
A のみ

➡

出力結果
A : #VALUE! ...
…はエラー説明の省略



…………… なぜ、このような結果になるの? ……………



戻り値の型：文字列

"abc"

テンプレートに値を埋め込む処理の
戻り値は「文字列」です。
エラーが起きても、#VALUE! などの
エラーメッセージが文字として
返ってきます。

IsErrorだけでは検出できない

IsError

✗

#VALUE!はエラー値ではなく、
「エラーメッセージという文字列」
のため、IsErrorだけでは
検出できません。



穴と値の数を揃えて、
出力も確かめよう。



エラー値の検査だけでは、
見逃すんですね。

💡 出力が期待どおりの内容か、必ず確認する習慣をつけましょう。 🐟

図5 | ここでの#VALUE!は、TakiLibが戻す文字列の一部。図の「…」はエラー説明を省略している。

6 行を貯めても、最後は内容を照合



図6 | textBuildは一回につき整形した一行を追加する。表示やファイルへの出力は別の処理。

まとめ

テンプレートを使うと、報告文の形と埋め込む値を分けられます。ただし、穴と値の対応、引数不足、数値や日付の表記まで自動で正しくなるわけではありません。シャケもんが組み立て方を整え、フナさんが報告としての内容を確認める。その両方が必要です。

今回の仕掛け

textFormatは、受け取った可変長引数を配列へまとめ、textFormatPackedへ渡します。内部では、{}を読むと順番用の位置を進め、{0}～{9}では指定した位置だけを使います。番号指定で順番用の位置は進みません。{10}は十番目を指定する書式としては解釈されません。

値の文字列化にはCStrを使います。引数不足などで値を取得・変換できなかった箇所には、#VALUE!にエラー番号などを添えた文字列が入ります。図5はこの経路の説明であり、Excelのセルが持つError型全般を文字列と呼んでいるわけではありません。IsErrorはエラー値かどうかを判定する関数なので、この文字列を渡してもTrueにはなりません。

textBuildは整形した文字列を配列の末尾へ追加し、同じ文字列を戻り値にもします。改行やファイル出力を自動で行う関数ではありません。また、数値・日付を外部へ渡すときは相手先の書式を決め、必要な整形を済ませてから埋め込みます。テンプレートはCSV・JSON・SQLなどの専用エスケープ処理を代替しません。

今回のソースコードのサンプル

サンプルコード 文面の入れ替えと二行の蓄積

```

01 Public Sub Test_Main()
02     Dim x_lines As Variant, x_line As Variant
03     x_lines = Array()
04
05     Debug.Print textFormat("件数={}, 担当={}", 3, "フナ")
06     Debug.Print textFormat("担当={1}, 件数={0}", 3, "フナ")
07     Debug.Print textFormatPacked("{1} / {} / {}", Array("A", "B"))
08
09     Call textBuild(x_lines, "件数={}", 3)
10     Call textBuild(x_lines, "担当={}", "フナ")
11     For Each x_line In x_lines
12         Debug.Print x_line
13     Next
14 End Sub

```

関数の説明

textFormatの第1引数は文面、第2引数以降は埋め込む値です。textFormatPackedでは、第2引数に値の配列を渡します。どちらも整形した文字列を返します。textBuildは、第1引数の配列へ整形した文字列を追加する点が異なります。

コードの説明

最初の二つのtextFormatは、同じ3と「フナ」を使い、文面の順番だけを変えています。続くtextFormatPackedは、番号と順番を混ぜたときの対応を示します。x_linesにはtextBuildで二行を追加し、For Eachで一行ずつ表示します。

入力・概要

TakiLib Ver 3.5.4本体と依存処理を導入したブックで、別の標準モジュールへ全文を貼り付け、Test_Mainを実行します。同名手続きがある場合は名前を変更してください。入力値はコード内にあり、表示先はVBEのイミディエイトウィンドウです。シートへの書き込みはありません。

結果

原ソースから予測される出力は、順に「件数=3、担当=フナ」「担当=フナ、件数=3」「B / A / B」「件数=3」「担当=フナ」の五行です。今回はこのサンプルをExcelで実行していません。

読むポイント

文章の形を変える箇所と、値を渡す箇所を分けて読みます。番号なしと番号付きの穴は、動きを確かめる例では混在させていますが、保守する文面ではどちらかへ揃えたと対応を追いやすくなります。

出典メモ

- TakiLib Ver 3.5.4、taki3libUTF8.basのtextFormat、textFormatPacked、textBuild。2026年9月13日に元資料TK-053の参照版とSHA-256の一致、関数本体・引数・結果を照合しました。
- [Microsoft Learn: IsError関数](#)。エラー値の判定と、説明文字列との区別を確認しました。
- [Microsoft Learn: データ型変換関数](#)。CStrの戻り値と、日付のシステム書式への依存を確認しました。