Claris

Claris FileMaker

JSON関数

事始め

- その2 -

OCT. 2020

VER. 1.1

Claris

目次

JSON形式を活用する(1) 1
JSON形式を活用する(2) 7
その２のまとめ 14

はじめに

このドキュメントは、2020 年 10 月実施の Web セミナー「はじめての JSON 関数 その 2」（Claris 法人営業本部）の内容をベースとしています。

本 Web セミナーの録画は次の Web サイトで視聴できます。

https://content.claris.com/ja_recorded-webinars

なお、本ドキュメントは、本ドキュメントの前編「JSON 関数事始め - その 1 -」に記載している JSON 形式の基本的な知識と FileMaker の基本的な JSON 関数の使い方を学習していることを前提としています。もし、初めて JSON 形式を学習する方で、まだ「その 1 」を読んでない方は、ぜひ「その 1」を先にご一読ください。

「JSON 関数事始め その 1」と同「その 2」、そしてそれぞれに対応する Web セミナーを併せてご利用になって、Claris® FileMaker® プラットフォームにおける JSON 形式のデータおよび JSON 関数についての理解をさらに深めてください。

JSON形式を活用する(1)

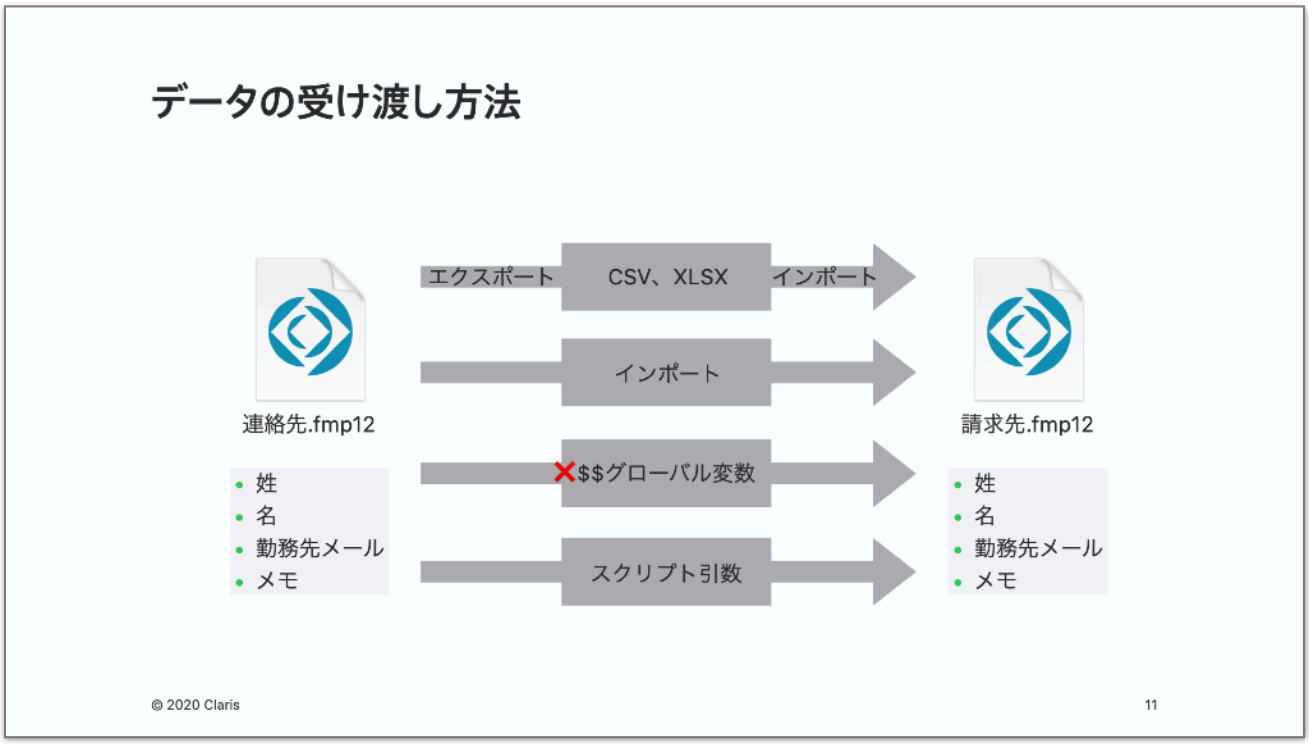
カスタム App の中で JSON 形式のデータをさらに活用してみましょう。

スクリプト変数

例えば、顧客の連絡先を管理するカスタム App から請求先を管理するカスタム App に、顧客の姓、名、勤務先メール、メモといったデータを渡したい場合、どうしますか？

FileMaker を使い始めて間もないときは、まず、連絡先 App で渡したいデータを CSV 形式や Microsoft Excel 形式の外部ファイルにエクスポートし、それを請求先 App でインポートする、といった方法を考えるかもしれません。これは、ユーザが直接 FileMaker を操作して実行することもできますし、スクリプトによって実行することもできます。異なるカスタム App 間で確実にデータの受け渡しができ、実際、よく見かける手法です。

しかし、この方法は、何百、何千件といった、ある程度まとまった量のデータを受け渡すときは良いのですが、1 件分の顧客連絡先を受け渡したいときにもいちいち外部ファイルを作って受け渡すのはちょっと大げさな気がしますね。また、外部ファイルを介することによって処理のオーバーヘッドが生まれますし、ファイルシステムに関係する思いも寄らないエラーが起きてしまうこともあります。



逆に、必要なデータを連絡先 App からインポートするスクリプトを請求先 App に作成して処理することも考えられます。この場合は、FileMaker だけで完結するので一見うまくいきそうですが、連絡先 App のテーブルで対象レコードを絞り込む必要があるので、そのスクリプトの作成は案外手間がかかります。

では、連絡先データをグローバル変数を使って受け渡す方法はどうでしょうか。この方法も FileMaker だけで完結し、かつ、グローバル変数を使って値を読み書きするだけなので簡単に実現できそうですが、残念ながらグローバル変数は、それを宣言したカスタム App ファイルの中だけで有効なもので、異なるファイルからは参照することはできません。

では、どうしたら簡単かつ確実にデータを受け渡すことができるでしょうか。

このような場合は、スクリプト引数を利用することができます。スクリプト引数は、テキストや数字をスクリプトに渡します。異なるカスタム App のスクリプト間でも、[スクリプトを実行] スクリプトステップ（送り側）と Get (スクリプト引数) 関数（受け側）を使って値を受け渡しすることができます。

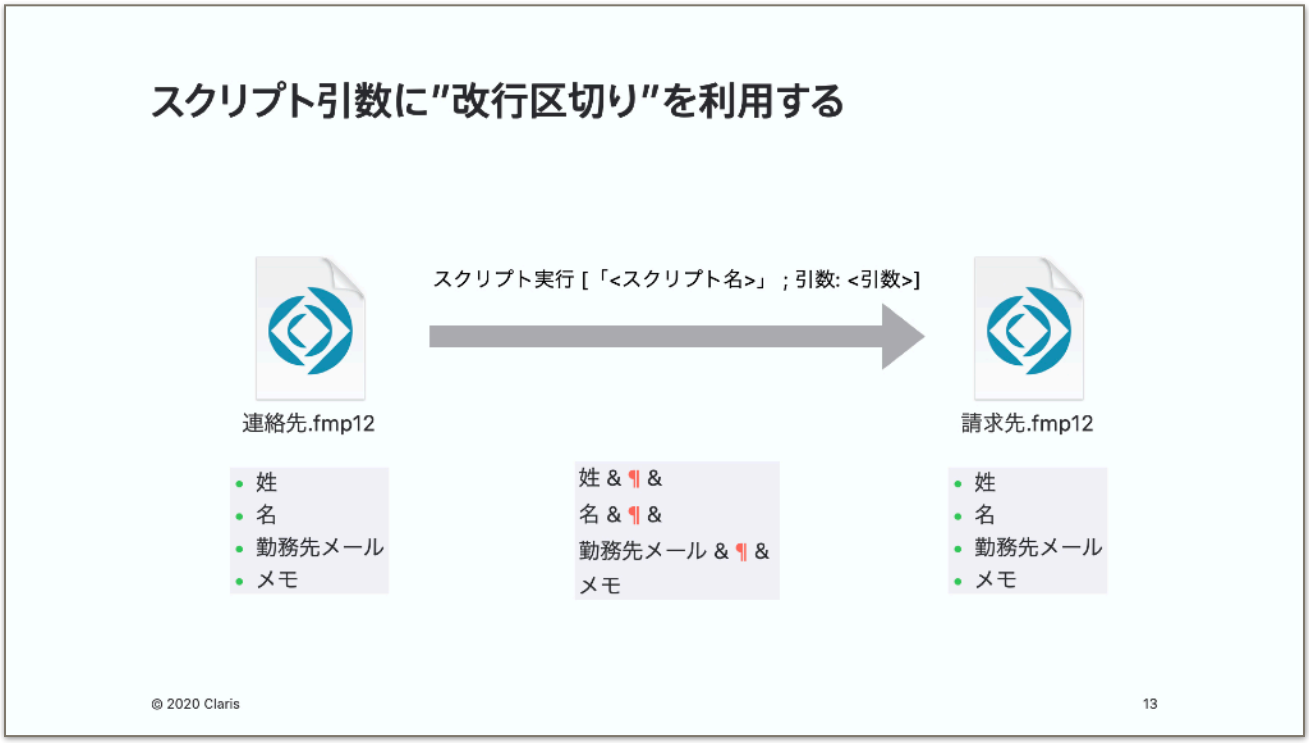
ただし、1 つのスクリプトには 1 つのスクリプト引数しか渡せません。

スクリプト引数で複数の値を渡す

それでは、スクリプト引数で、顧客連絡先の姓、名、勤務先メール、メモといった複数の値を渡すにはどうしたらいいでしょうか。

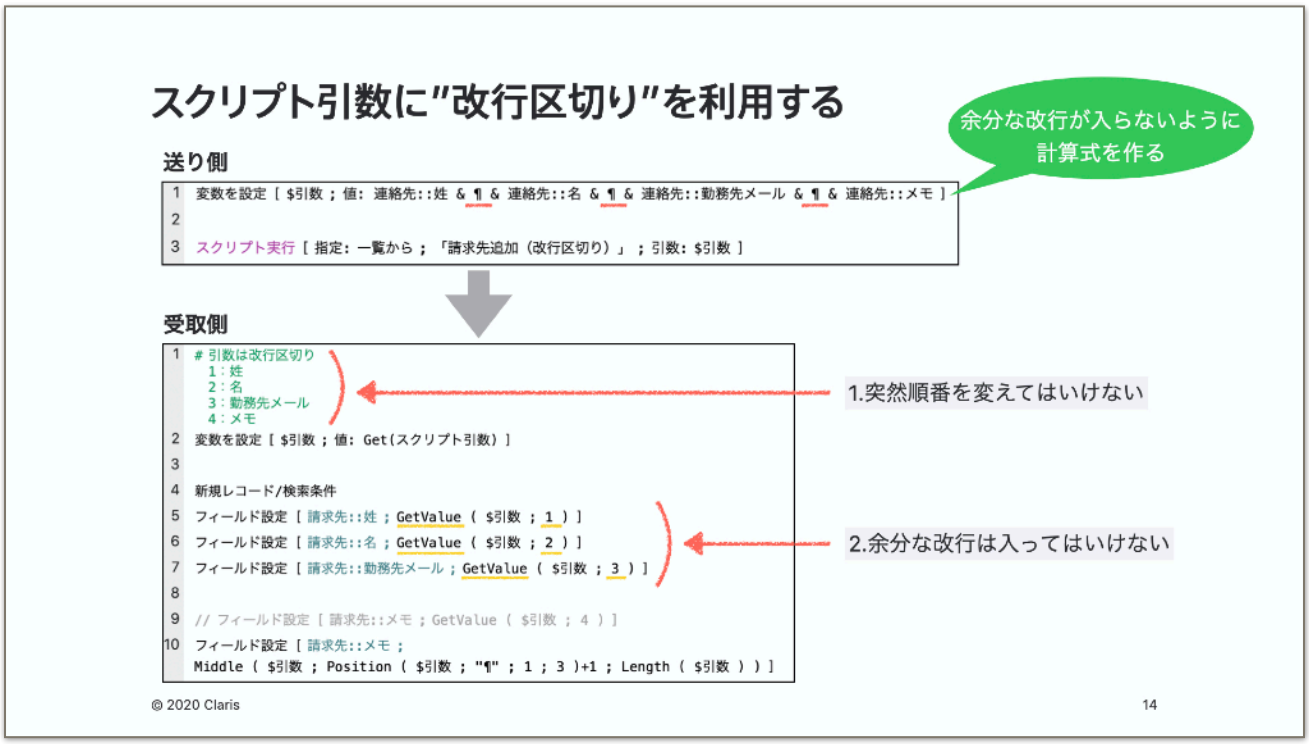
最も一般的な手法の一つは、改行区切りのテキストを使うものです。

これは、スクリプトに渡したい複数の値の間に改行文字（¶）を FileMaker の List 関数あるいは手動で挿入し、１つのテキストとしてスクリプト引数に設定する方法です。受け取ったスクリプトでは、GetValue 関数を使用して目的の値を取り出して使用します。

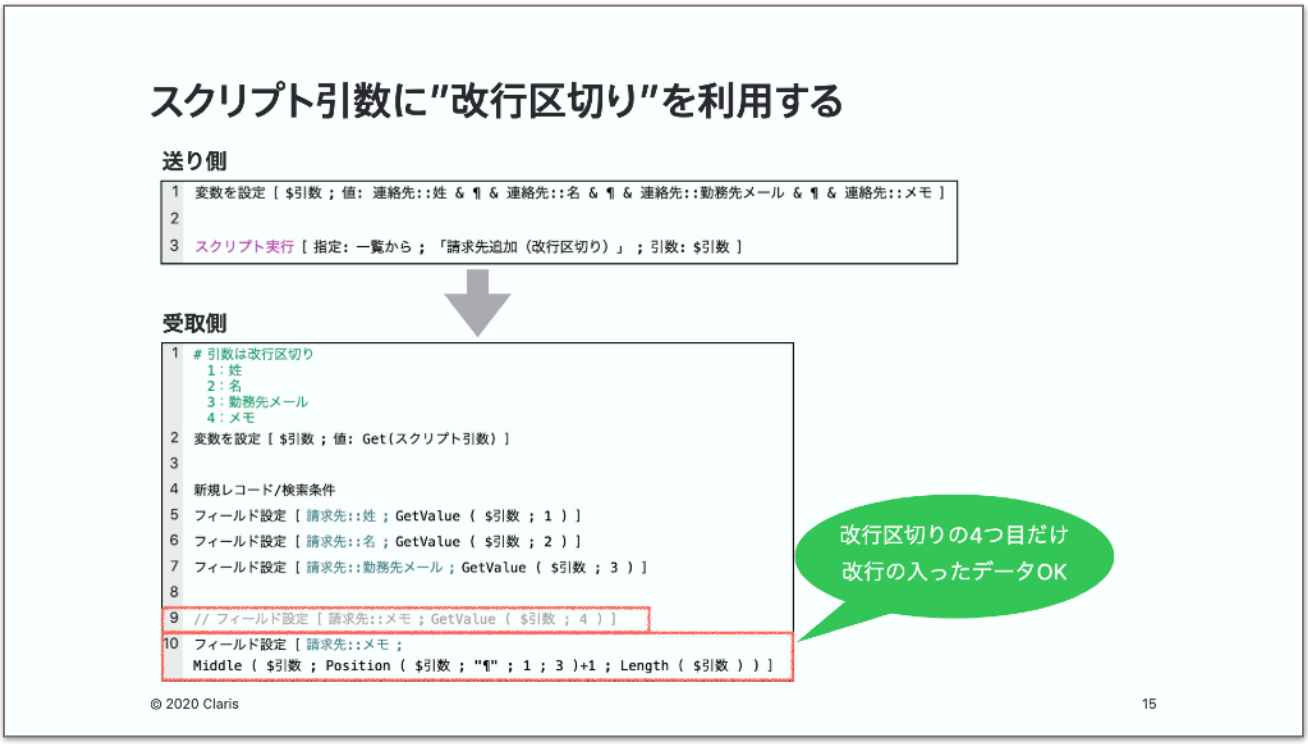


ただしこの手法では、スクリプト引数を作る側（ユーザあるいはスクリプト）も受け取って使用するスクリプトでも、値の種類と順番を同じルールで解釈する必要があります。「姓、名、勤務先メール、メモ」の順と決めたら、「名、姓、電話番号、勤務先メール、メモ」のように、順番を変えたり新しいデータを追加したりすることはできません。

また、値の中に改行が含まれていると改行以降は次の値と解釈されてしまうので、値の中には改行が入らないようにする必要があります。



例えば「メモ」フィールドのように改行が含まれることが想定されているデータであれば、それをスクリプト引数の最後に置き、「O個目の値の次からテキストの最後までを１つの値として取得する」というイレギュラーな方法も考えられます。しかし、この方法は改行を含むデータが１つの場合にしか利用できません。



それならば、改行文字ではなく他の文字を区切り子（区切り文字）として使う、といった方法も考えられますが、区切り子としてデータに決して現れない文字を選択する必要があり、また、処理自体も煩雑になってきます。

これに対して、名前と値のペアを使う手法も提案されています。これは、カスタム関数を使って、名前と値のペアの集合（辞書）を１つのテキストとして作成し、名前を使って自由に値を取り出したり編集したりするというものです。ただ、この手法はカスタム関数で実現されているので、利用するためには必要なカスタム関数をすべてのカスタム App ファイルに取り込む必要があります。また、FileMaker の標準機能ではないので、このようなカスタム関数自体の存在を知らないユーザも数多くいます。

JSON形式で複数の値を渡す

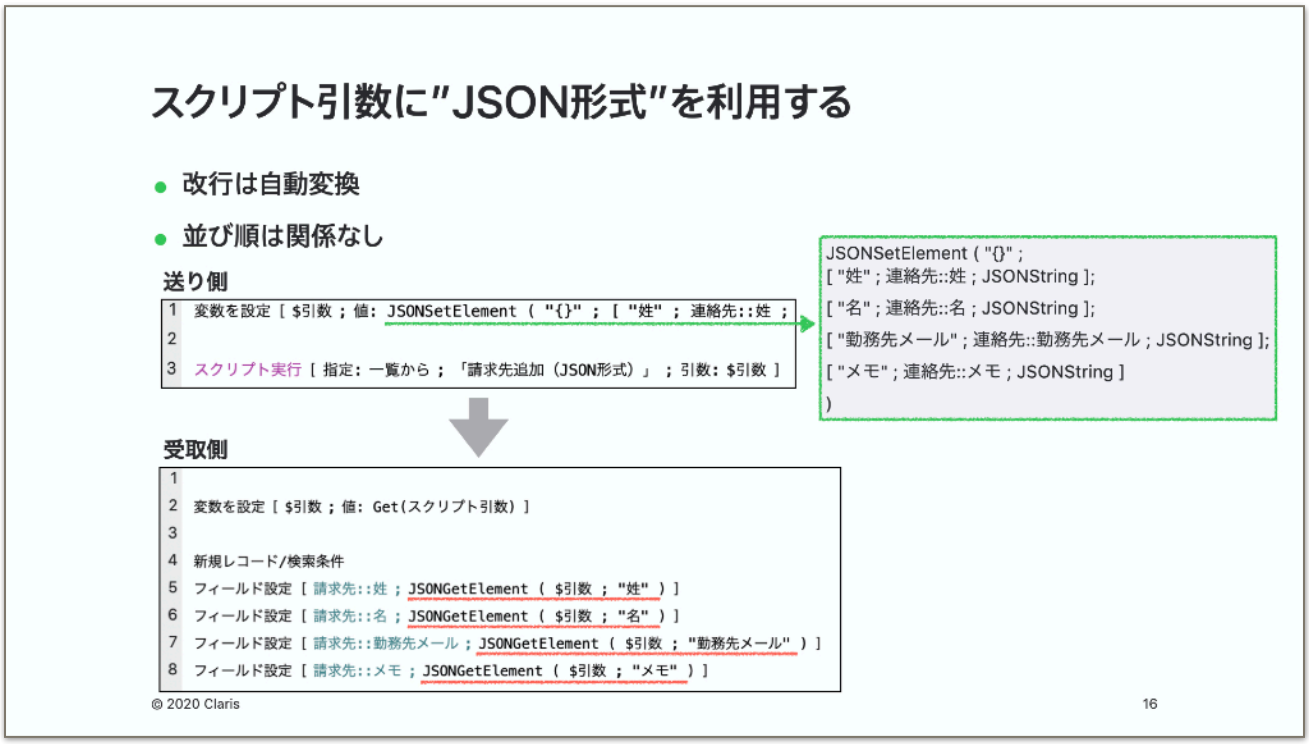
ではここで、スクリプト引数に JSON 形式を利用することを考えてみましょう。JSON 形式のデータは１つの文字列なので、複数の要素（引数データ）を持った JSON データをスクリプト引数にそのまま渡すことができます。

まず、スクリプト引数として渡す JSON データを生成するには、前編の「JSON 関数事始め - その1 -」でも紹介した JSONSetElement 関数を使います。図の例では、「姓」、「名」、「勤務先

メール」そして「メモ」をキーとして要素を設定しています。このとき、各要素の値に改行が含まれても問題ありません。

スクリプト引数を受け取ったスクリプトでは、同じく「その 1」で紹介した JSONGetElement 関数を使ってそれぞれのキーに対応する値を取得します。このとき、開発者はスクリプト引数の中で各データがどのような順番で並んでいるかをまったく気にする必要はありません。

このように、スクリプト引数として JSON 形式のデータを利用することによって、FileMaker の標準機能だけを使い、開発者が意識しなければならないことを大幅に減らせるだけでなく、実装方法も簡潔にすることができます。



なお、ここでは複数のカスタム App ファイル間でスクリプト引数を使う場合についてご紹介しましたが、1 つのカスタム App ファイルの中で利用することもあります。1 つのカスタム App の中では、複数の値を受け渡すために個々に変数を設定することがありますが、JSON 形式のデータを使うことによって、変数を個々に定義する手間が省けるというメリットがあります。

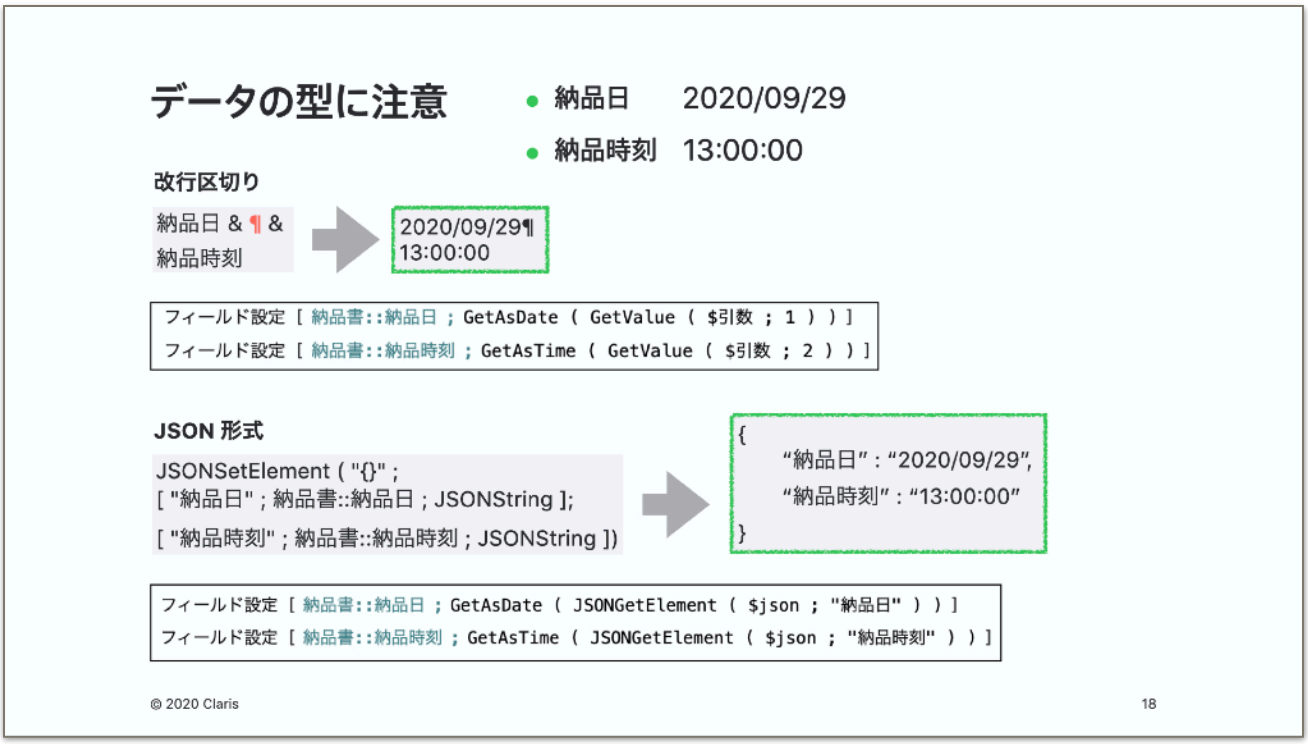
JSON形式とデータタイプ

次に、「納品日」、「納品時刻」のような日付と時刻データを JSON 形式で受け渡す場合を考えてみます。

前編の「JSON 関数事始め - その 1 -」で紹介したように、JSON には FileMaker の日付、時刻、タイムスタンプなどのフィールドタイプに対応する型がないので、このようなデータは文字列（JSONString）として扱います。ところが、受け側のスクリプトは人間と違って、この文字列が

日付データなのか、時刻データなのか、ただのテキストなのか、そのままでは常に正しく解釈することはできません（たまに正しく解釈されることもあります）。

このため、JSON データを受け取ったスクリプトでは、JSONGetElement 関数で値を取り出した後、GetAsDate 関数や GetAsTime 関数を使って値を対応するフィールドタイプに変換する必要があります。



また、数値を JSONNumber タイプを指定して JSON データを生成すると 19 桁以上の数は精度が失われるため、代わりに JSONString タイプを使用することがあります。この場合は、JSON データから値を取得した後に GetAsNumber 関数を使って FileMaker の数字に変換することを忘れないようにしましょう。

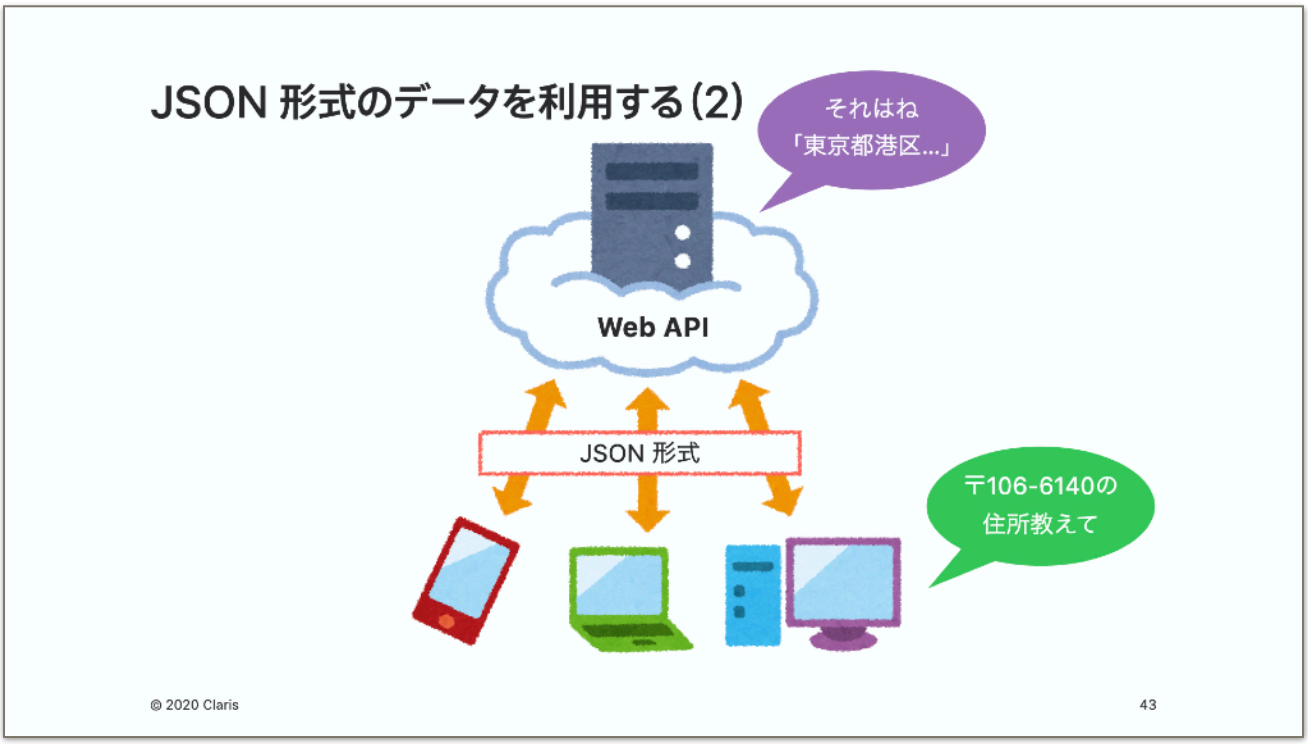
JSON 形式のデータをスクリプト引数として使う方法については、FileMaker Master Book 中級編の「11.5.5 複数の引数」でも取り上げていて、演習を交えて学習することができます。ぜひこちらも参照してください。

JSON形式を活用する(2)

Web API から取得したデータをさらに活用してみましょう。

Web APIから複数のデータを取得する

前編の「JSON 関数事始め - その 1 -」では、郵便番号による住所検索 API[1] を使って、指定した郵便番号に対応する1 つの住所を取り出しました。

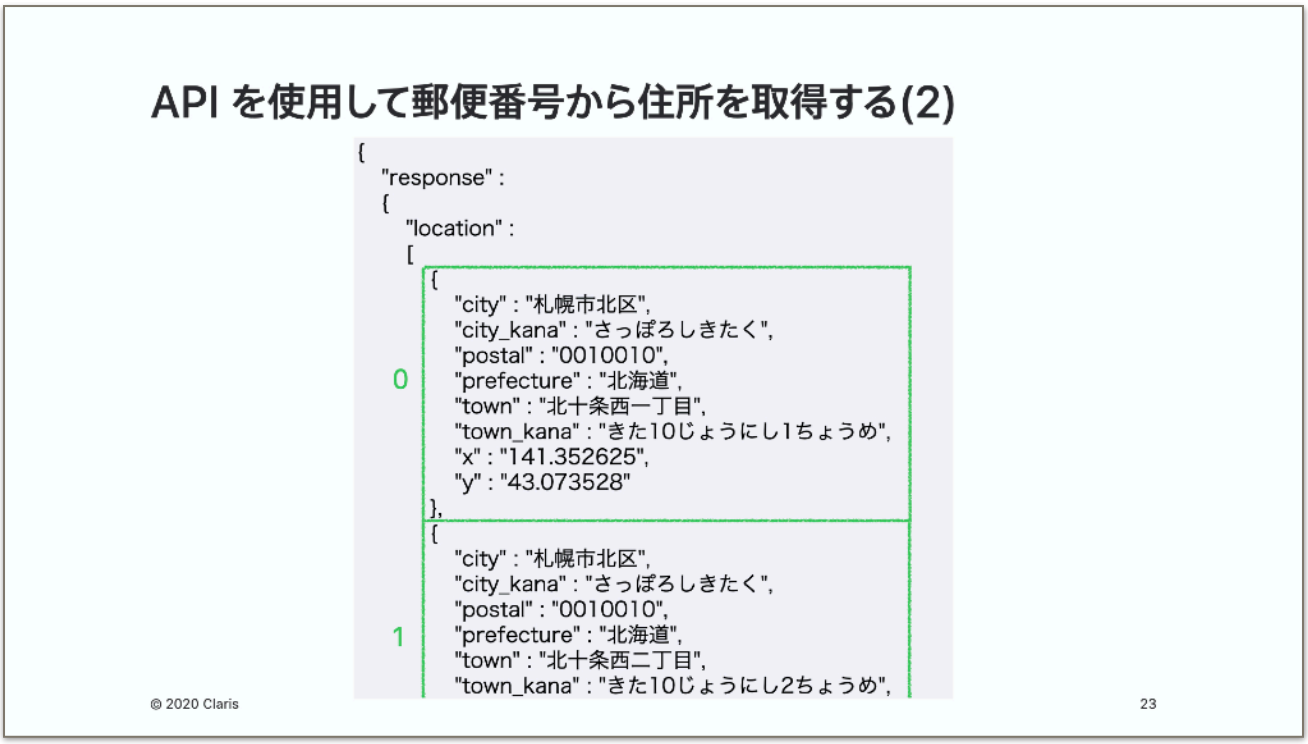


今回は、1 つの郵便番号に対して複数の住所が対応する場合を考えます。

例えば、北海道札幌市の「001-0010」という郵便番号には複数の住所が対応しています。1 つの郵便番号に対応する住所が 1 つだけであればそのまま処理を進められますが、複数個ある場合にはユーザに目的の住所を選択してもらう必要があります。

---

[1] 郵便番号／住所／緯度経度データサービス Heart Rails Geo API
郵便番号による住所検索 API:　https://geoapi.heartrails.com/api.html#postal

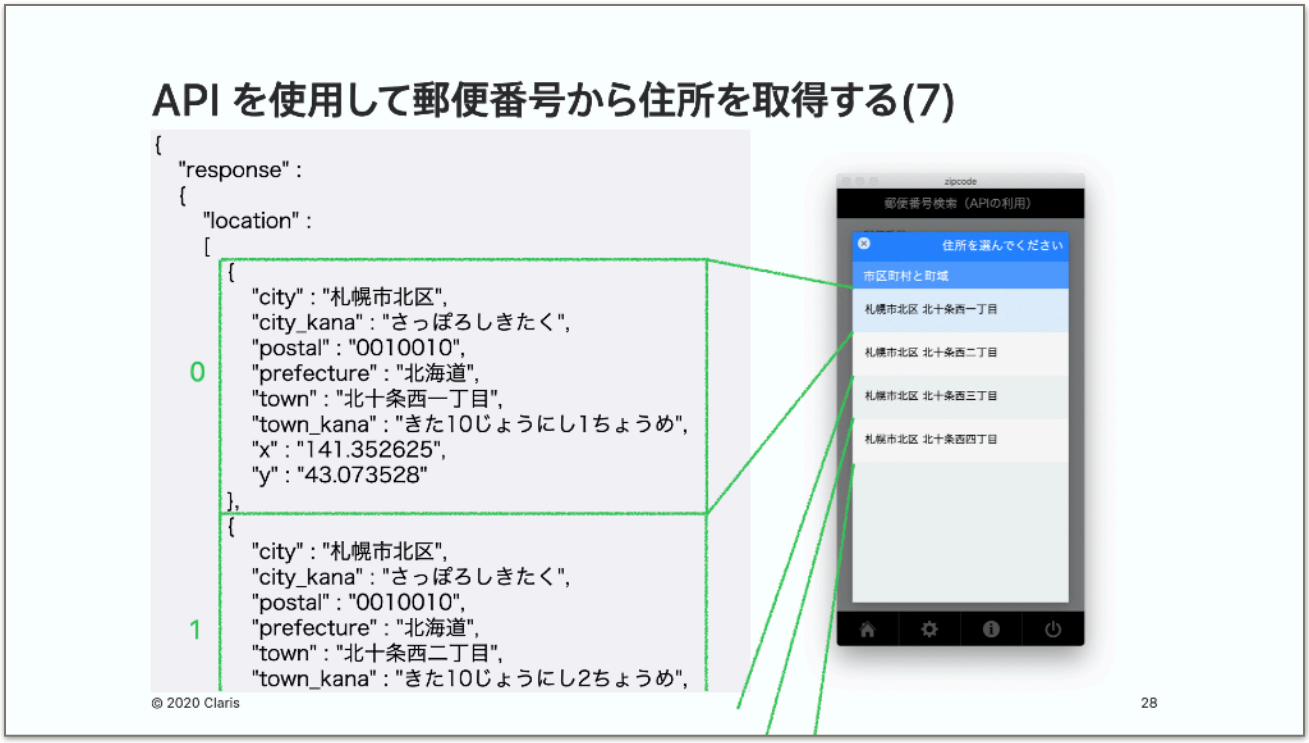


実際のアプリのイメージは次のようになります。

まず最初にユーザが郵便番号欄に郵便番号を入力して虫眼鏡ボタンを押す（ステップ 1 ）と、スクリプトが起動し、入力値に問題がないかチェックした後、Web API にリクエストを投げます。そして、Web API から返ってきた住所データが 1 つだけの場合は、そのデータを「都道府県」、「市区町村」、「町域」フィールドにそれぞれ入力します。

一方、Web API から返ってきた住所データが複数個ある場合は、サブスクリプトを起動して複数の住所データのリストを作成し、それをカードウインドウに表示してユーザに選択を促します（ステップ 2 ）。ユーザがリスト中の住所を 1 つ選択すると、そのデータを「都道府県」、「市区町村」、「町域」フィールドにそれぞれ入力して処理を終了します（ステップ 3 ）。

この処理の一番のポイントは、ステップ 2 の住所のリストを作成するところです。ここでは、Web API から返ってきた JSON オブジェクトの中から住所データを一つ一つ取り出して、住所名のリストにする必要があります。具体的は、「response」オブジェクトの中の「location」オブジェクトに JSON 配列として格納されている値（JSON オブジェクト）を順に取り出していきます。



これを Loop による繰り返し処理で実現するために、次にご紹介する JSON 関数を利用します。

JSONListKeys関数とJSONListValues関数

FileMaker の JSON 関数には、JSON 形式からデータを取り出す関数として、JSONGetElement 関数の他に、JSONListKeys 関数とJSONListValues 関数があります。

構文

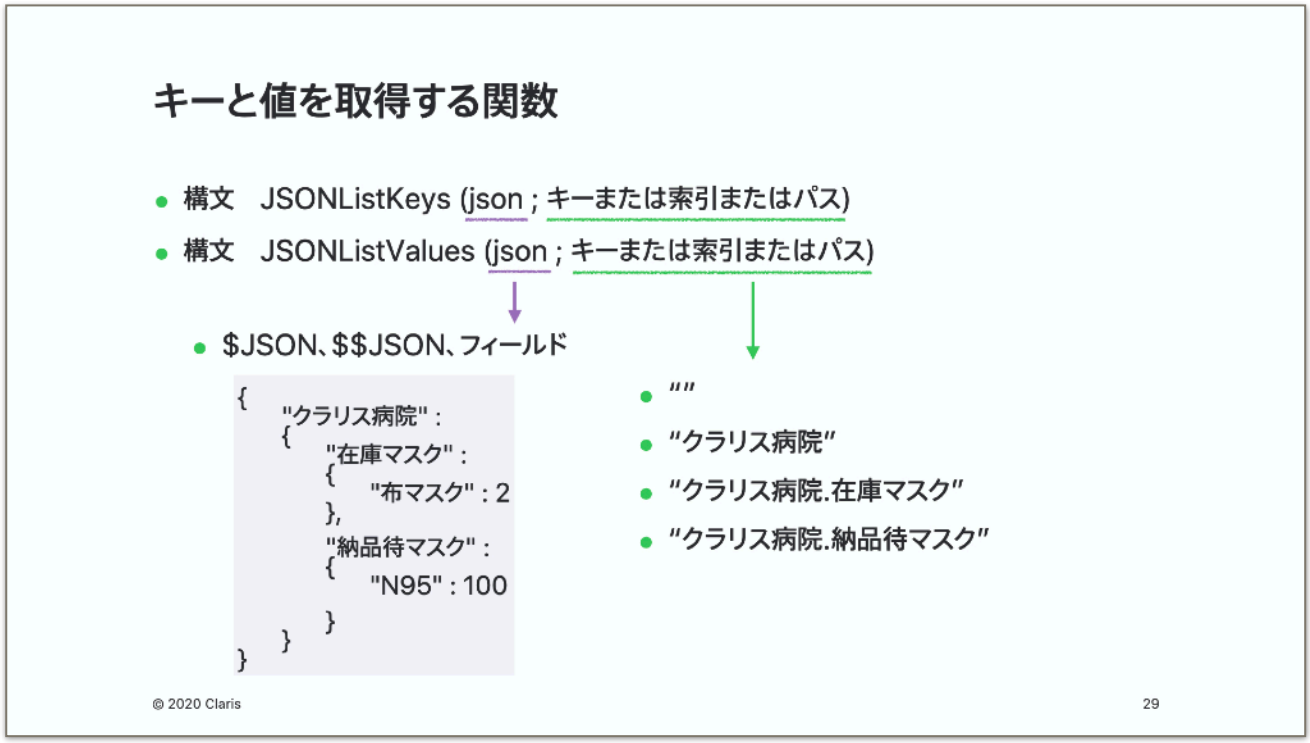
JSONListKeys (json ; キーまたは索引またはパス)

構文

JSONListValues (json ; キーまたは索引またはパス)

JSONListKeys 関数は、第 1 引数で指定された JSON オブジェクトから、第 2 引数で指定されたキーまたは索引またはパスに対応するオブジェクト名（キー）または配列索引の一覧を返します。

JSONListValues 関数は、第 1 引数で指定された JSON オブジェクトから、第 2 引数で指定されたキーまたは索引またはパスに対応する値の一覧を返します。

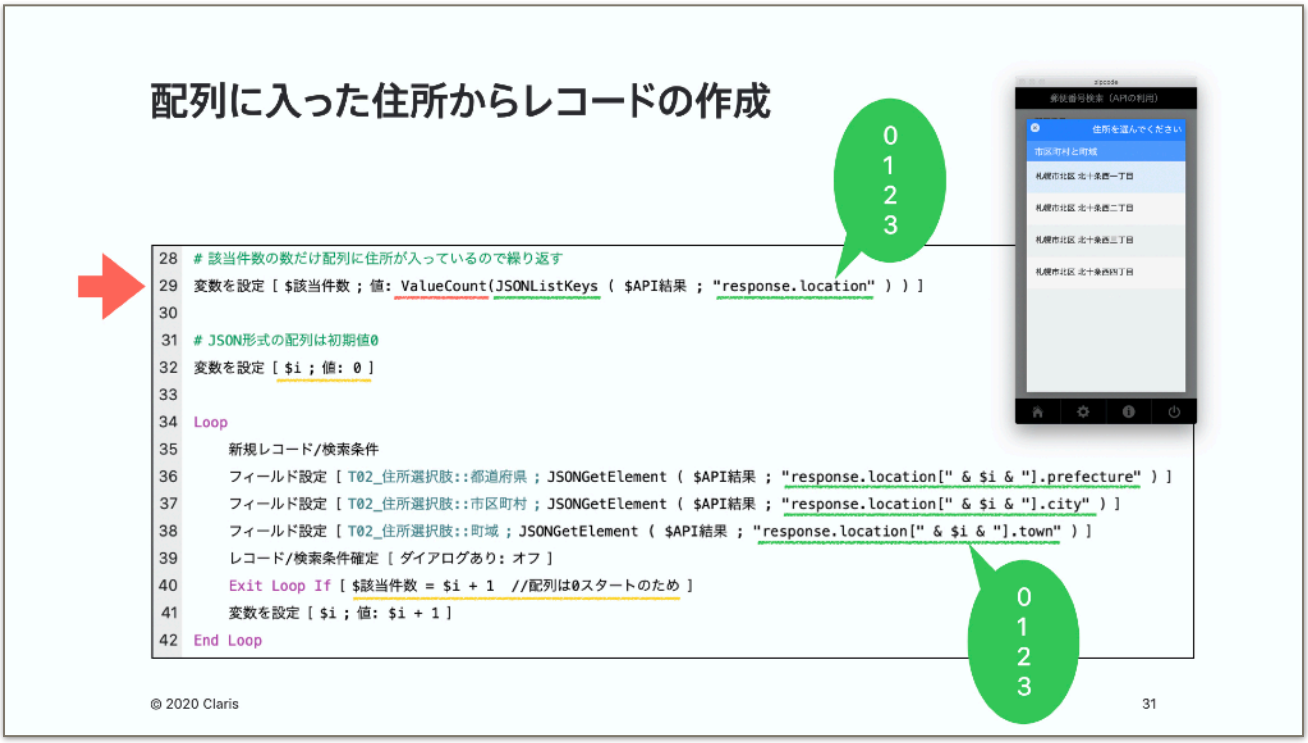


それぞれの関数の利用例について、詳しくはFileMaker Pro ヘルプを参照してください。

ここでは、これらの関数を利用することで、JSON データの中に要素が何個含まれるかを調べます。

Loopを使ってJSON配列要素を取り出す

実際に複数の住所データを取り出すスクリプトを見てみましょう。



このスクリプトではまず、JSONListKeys 関数を使って「response.location」に含まれる要素の数を ValueCount 関数で数えています。「response.location」は JSON 配列なので、JSONListKeys 関数は、配列のすべての要素の索引（0, 1, 2, ...）を取り出し、改行区切りの文字列として返します。この例では 4 つの住所データが含まれているので「0¶1¶2¶3」が JSONListKeys 関数から返されて ValueCount 関数の引数として渡され、最終的にこの式によってローカル変数「$該当件数」には「4」という数字、すなわち配列要素の数が設定されます。

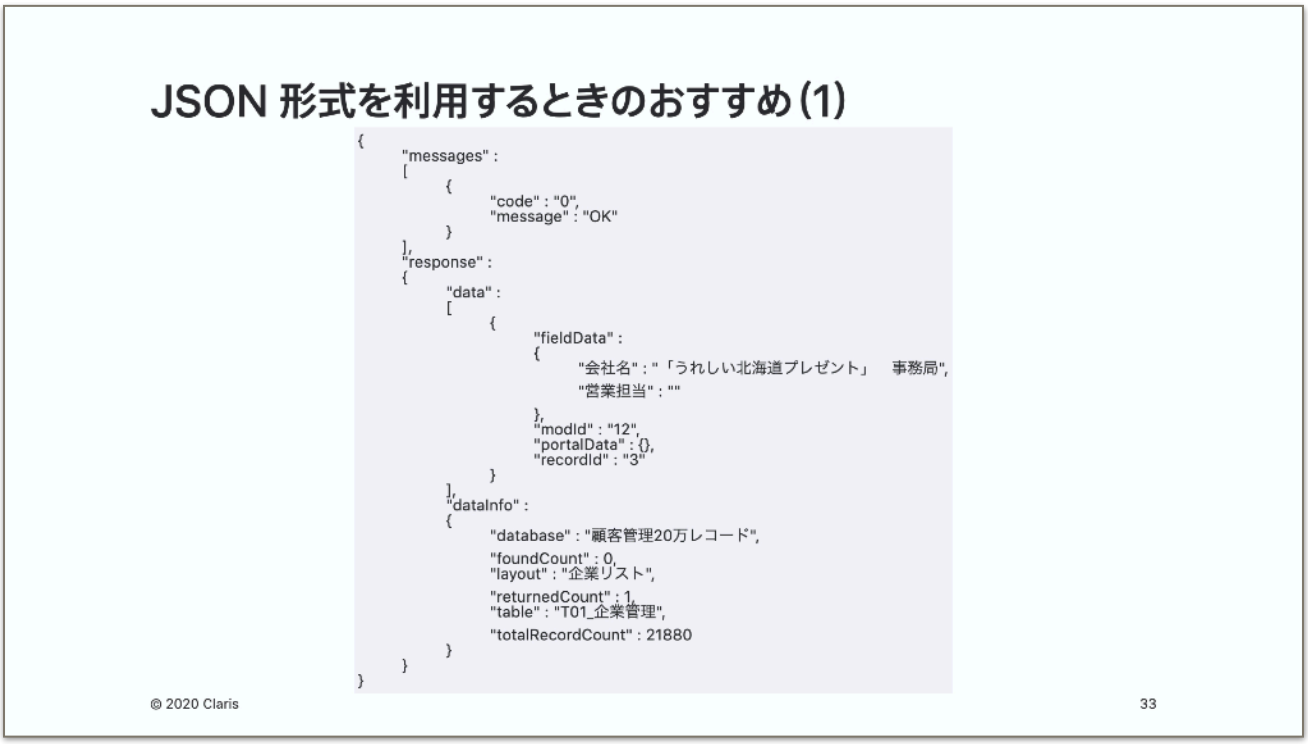
Loop の中では、JSON 配列中のそれぞれの JSON オブジェクトから、市区町村（"city"）と町域名（"town"）に対応する値を取り出して対応するフィールドに設定します。JSON 配列の各配列要素には配列の索引（ $i ）を指定してアクセスしますが、このとき、JSON 配列の索引は「0」から始まるので、ローカル変数「$i」の最初の値として「0」を設定することに注意してください。

Loop の終了条件として Exit Loop If スクリプトステップに指定する条件は、最初に JSONListKeys 関数を使って設定した「$該当件数」、すなわち、配列要素の数です。現在のローカル変数「$i」の値は、それまで処理した実際の配列要素の数よりも 1 だけ小さい（「$i」の初期値が「0」のため）ので、プラス 1 した値が「$該当件数」と同じになったら Loop を抜けます。

JSON形式のデータを利用するときのおすすめ

JSON 形式のデータは、配列や入れ子を使って複雑な構造を簡単に記述することができます。ただ、JSON データの構造が複雑になってくると、キーの名前や、どの要素がどのキーの下にあるかといった論理構造を覚えておいてスクリプトを書くことが難しくなってきます。

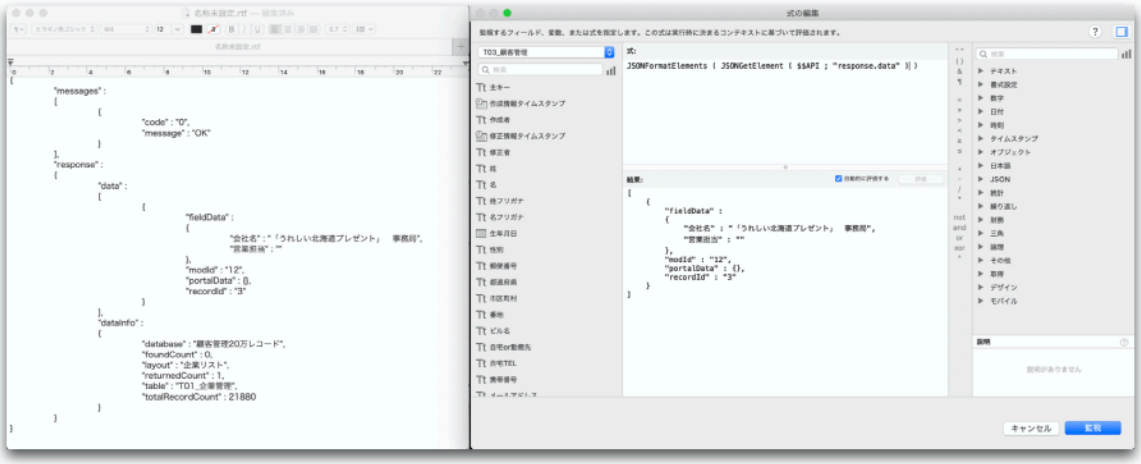
例えば、次の一見複雑な JSON データも、これまで見てきた JSON 関数を使って順番に処理すれば何のことはないのですが、スクリプトを記述するときにこの構造を覚えておくのはなかなか大変です。



そのような場合には、JSON 形式のデータを一旦 JSONFormatElements 関数で整形し、そのテキストをメモ帳やテキストエディタに取り出しておくことをお勧めします。それをデータビューアと並べて置き、データビューアで JSONGetElement 関数を使ったとき、あるいは、JSONListKeys 関数を使ったときにどういった結果が返ってくるかを見てみると、JSON データの構造がとてもよくわかります。

簡単な方法なので、ぜひ試してみてください。

JSON 形式を利用するときのおすすめ(2)



© 2020 Claris 34

この他、JSONFormatElements 関数で整形したデータを、スクリプトの中にコメントとして入れておくこともお勧めです。これによって、今のスクリプトの中でどのような JSON データを扱っているのかを目で確認しながら JSON 関数を使ったスクリプトを書くことができます。

JSON 形式を利用するときのおすすめ(3)

```
# 完成イメージ
#{
        "基本情報" :
        [
            {
                "fフィールド名" : "f3",
                "項目名" : "No"
            },
            {
                "fフィールド名" : "f2",
                "項目名" : "案件番号"
            }
        ],
        "導入予定" :
        [
            {
                "fフィールド名" : "f16",
                "項目名" : "導入予定"
            }
        ]
    }
```

© 2020 Claris 35

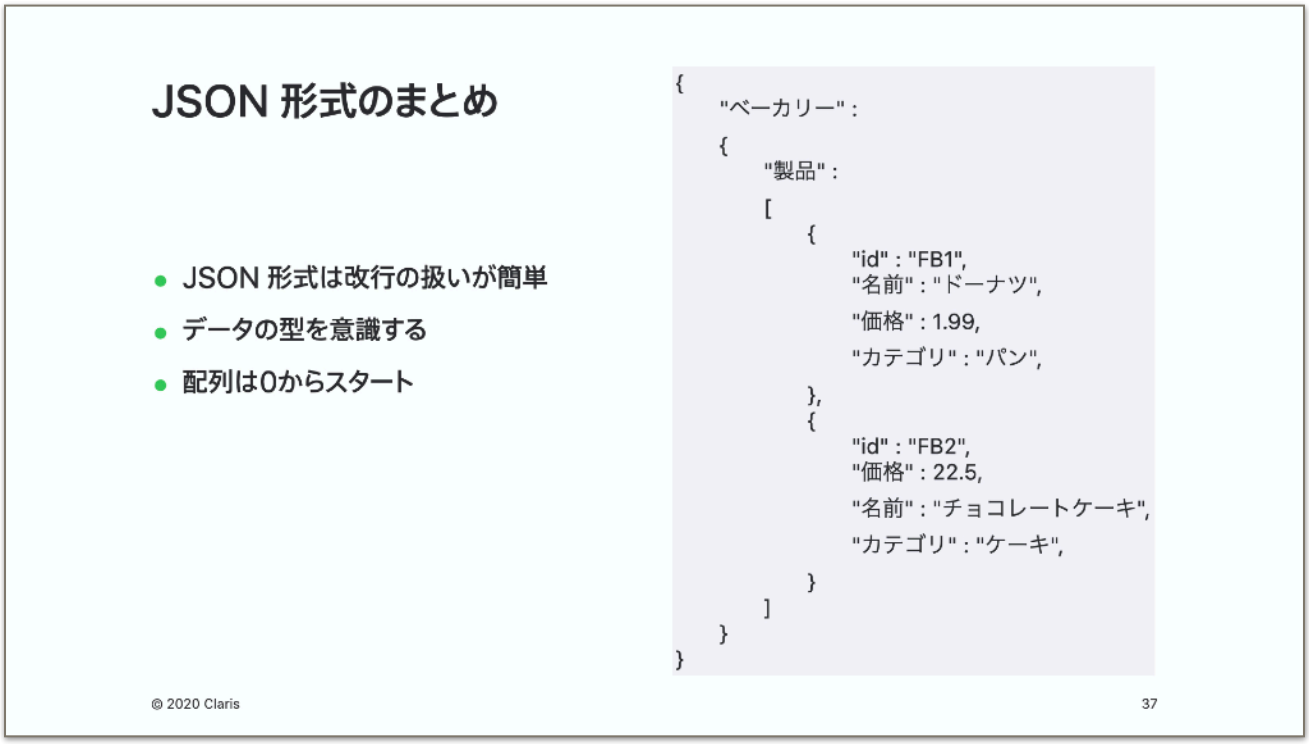
その２のまとめ

本編では、前編の「JSON 関数事始め - その 1 -」に引き続き、FileMaker で JSON 形式のデータをさらに活用するための方法をみてきました。

FileMaker のスクリプト引数に JSON 形式のデータを利用すると、複数の情報（値）を渡す場合に値の順序を意識する必要がないだけでなく、改行を含むテキストをそのまま渡すことができるため、従来の改行区切りの文字列で渡す方法に比べてデータの受け渡しがとても簡単になります。

一方、JSON と FileMaker では、利用可能なデータタイプが異なったり、仕様が異なったりするので、JSON 形式のデータを使ってデータ交換をする場合には、送り側と受け側のタイプを常に意識する必要があります。FileMaker 側で JSON データから取り出した値を利用する場合には、“GetAs” で始まる関数を利用して適切なタイプに変換することを忘れないようにしましょう。

また、JSON 配列は値（JSON オブジェクト含む）を順序付けて管理することができて便利ですが、配列の索引は「0」から始まることに注意が必要です。特に Loop で繰り返し処理するときなど、配列要素を適切な範囲を処理するためには、これは重要なポイントです。



前編の「JSON 関数事始め - その 1 -」および本編の 2 回にわたり、FileMaker の JSON 関数とその利用方法についてご紹介してきました。JSON 形式のデータを使い慣れてない方には、まだピンとこないところもあるかもしれません。しかし、今後、FileMaker を使って様々なカスタム App を作成していくにつれ、JSON 形式のデータを使う機会がどんどん増えていくはずです。ここで紹介した JSON 関数の使い方や tips などを参考にして、どんどん JSON データを活用していってください。

参考資料

1. FileMaker Pro 19 ヘルプ
   https://help.claris.com/ja/pro-help/#page/FMP_Help%2Findex.html

2. 「FileMaker 16 の新機能『JSON 関数』」、株式会社イエスウィキャン、May 2017.
   https://ywc.com/filemaker/?p=3908

3. Krueger, S., "JSON, You Deserve [Arrays]," Claris Engage 2020, Sept. 2020.
   https://youtu.be/ZpyMs9_VJ7w

4. Kos, M., "FileMaker JSON Serialization and Deserialization Gotchas," Soliant Consulting, Inc., Aug. 2019.
   https://www.soliantconsulting.com/blog/filemaker-json-serialization/
   （邦訳）nomfjmt, 「FileMaker <<>> JSON データ変換時の注意点」、
   https://notonlyfilemaker.com/2019/09/filemaker-json-serialization/ (Not Only FileMaker)

5. Krueger, S. A., "FileMaker 16's New {JSON} Functions," LuminFire Brilliant Solutions, Sept. 2017.
   https://luminfire.com/2017/09/30/filemaker-16s-new-json-functions/

6. Introducing JSON
   https://www.json.org/

7. 「FileMaker JSON 関数事始め - その 1 - 」, Claris International Inc., Sep. 2020.
   https://downloads.claris.com/kk/downloads/pdf/JA_FileMaker_JSON_1.pdf

8. Claris オンデマンド Web セミナー「はじめての JSON 関数　その 2」, Claris International Inc.
   https://downloads.claris.com/kk/video/webseminar/20201006_fm19_json_2.mp4
   https://youtu.be/eA1dPoUXWBA