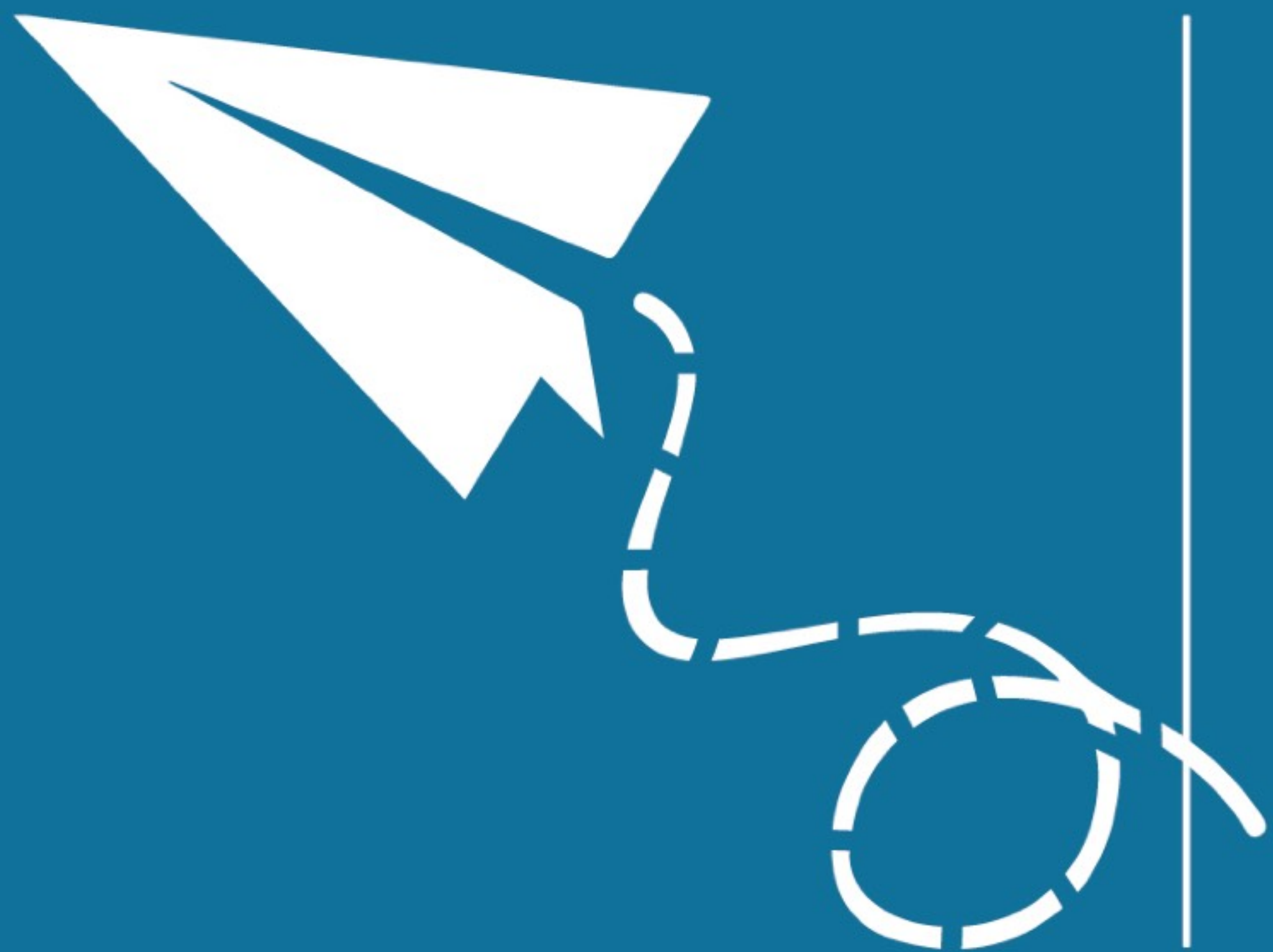




ROME
FILEMAKER®
WEEK
2023

JSON

What is it?

Why does it matter?

Matt Navarre

JSON



Let's Interact

zoom.us

503 616 9422



About Matt



Hey, nice suit ^

- Passionate FileMaker trainer
- Expert FileMaker developer
- Certified in FM versions 7-2023
- Founded 3 companies
- FBA Awards, conferences, podcasts, Meetups



Free Stuff

www.navarre.training

- > SearchResutls 7
 - > All JSON
 - > Faster and easier to customize
- > fmLog 3
 - > Coming soon...
- > Training classes - Champion Track starts in January



Quick Poll

Do you use JSON:

Never

Sometimes

Daily



Two Part Agenda

- > First: JSON Intro
 - > Interactive: Let's write code
 - > Why I love it
- > Second: Experienced JSON Users
 - > Show your stuff
 - > Show Unique Uses
 - > You have 90 seconds



What is JSON?

- > JavaScript Object Notation
- > Structured text formatting
- > “Light weight data interchange format”
- > Human Readable
- > Like XML, but it doesn't suck™



What is JSON?

- > Hierarchical (nested structure)
- > VERY FLEXIBLE
- > FileMaker-y
- > Since Version 16
 - > May 9th, 2017
 - > 2.343 days ago



Why?

Simpler

- > Once you are comfortable using it
- > Just like Case ()
- > And Let ()

Better

- > Flexible
- > Bulletproof



Use Cases

- > Passing Parameters
- > Configurations & Preferences
- > API Integration & Claris Connect
- > Audit Logs
- > Record Snapshot

When?

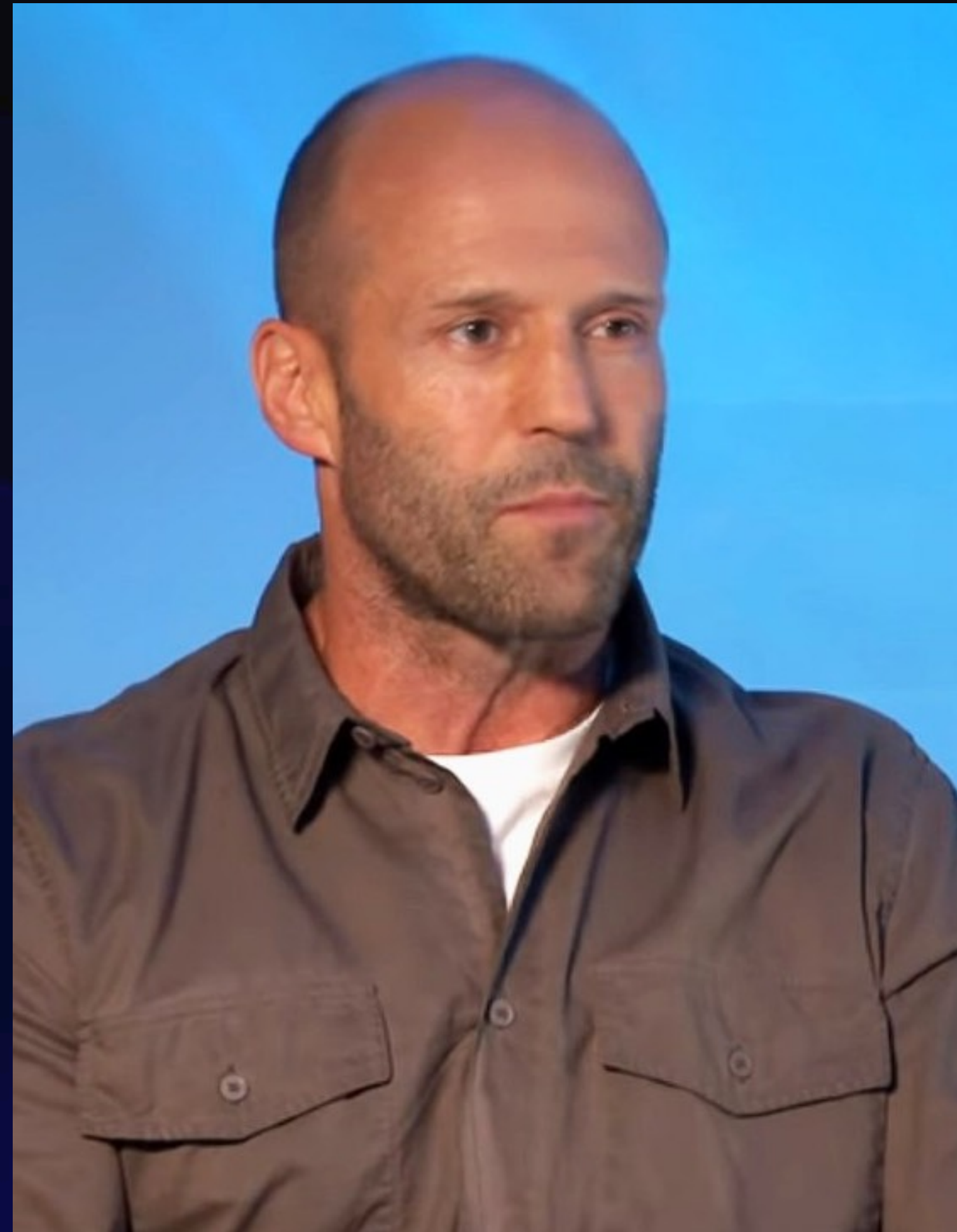


- > Here
- > Today
- > Now

Who?



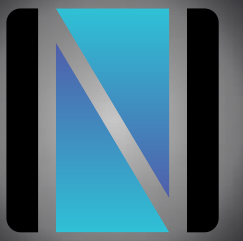
Who?



Who?



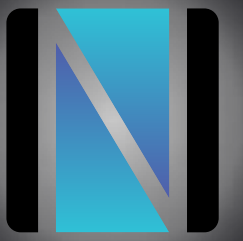
Who?



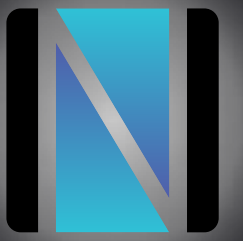
Who?



Who?



Who?



Douglas Crockford



Two Functions

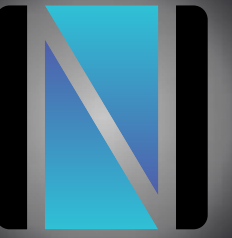
JSONGetElement ()

- > Two arguments: Name and Value

JSONSetElement ()

- > Four arguments — two that matter most

Data Viewer



Data Viewer

Current Watch

Monitor the values of fields, variables, and expressions.

Expression

JSONFormatElements (\$json_in)

JSONFormatElements (\$json_out)

""

JSONFormatElements (JSONSetElement ("" ; ["results.text" ; "Welcome to Quintana Gallery DB" ; JSONString] ; ["results.type" ; "h.

ValueCount (JSONListKeys (\$\$sr_JSON ; "results"))

GetAsTimestamp ((Get (CurrentTimeUTCMilliseconds)) / 1000)

Let ([incoming = Import::Item ; proc = RightWords (incoming ; 1) ; proc = Substitute (proc ; "." ; "/") ; m = Middle (proc ; 4 ; 2) ;...

FieldNames (Get (FileName) ; Get (LayoutName))

Let ([id = ACCT__Account::ID ; user = Get (AccountName) ; type = "Edit record" ; from = Case (IsEmpty (\$\$current_value) ; "[none]"

Let ([id = ACCT__Account::ID ; j = DASH__Board::g_JSON Account Toggle ; keys = JSONListKeys (j ; "") ; included = PatternCount.

Let ([filt = \$filter_recent ; today = Get (CurrentDate) ; yesterday = ">" & today - 1 ; week = ">" & today - 7 ; // this might chang.

Let ([q = "\""; bs = "\"\""; command = "-X GET " ; url = "https://aeroapi.flightaware.com/aeroapi/operators/OMA/flights?" ; head = '

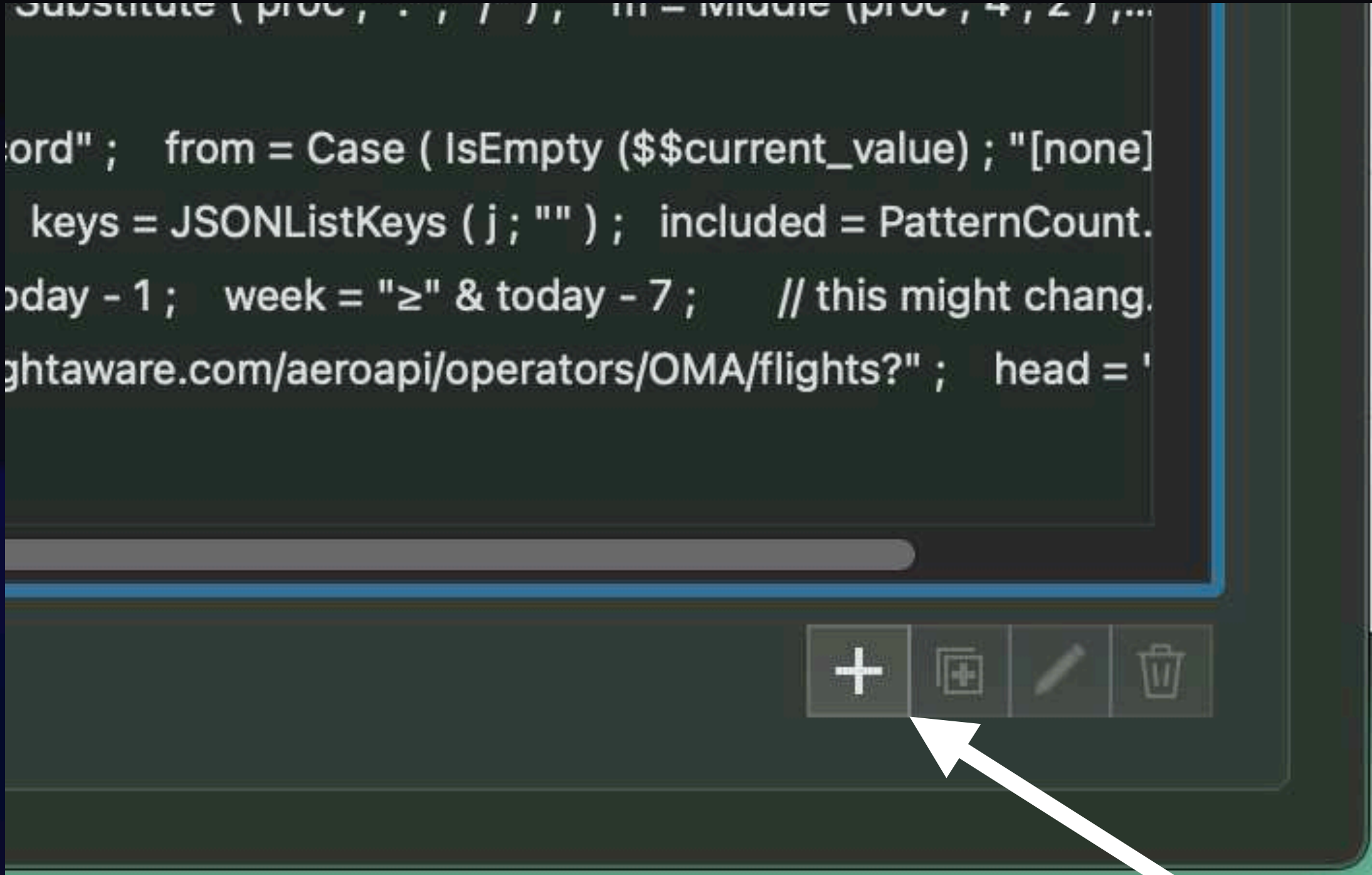
Refresh Values

+

+

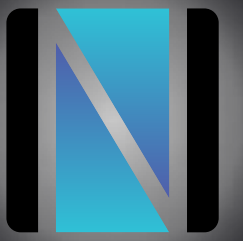


Data Viewer



Click here

Set



FM Defaults

What's important?

My preferred formatting

Expression:

```
JSONSetElement ( json ; keyOrIndexOrPath ; value ; type )
```

```
JSONSetElement ( existing json ; NAME ; VALUE ; JSONString )
```

```
JSONSetElement ( "" ;  
NAME ; VALUE ; JSONString  
)
```



Set : Single

Expression:

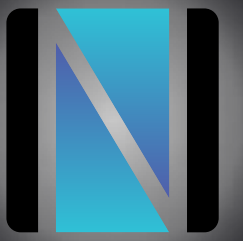
```
JSONSetElement ( "{}" ;  
    "name1" ; "1" ; JSONstring  
)
```

Result:

```
{"name1": "1"}
```

Code Break

String Number Boolean, etc.



Expression:

```
JSONSetElement ( "{}" ;  
  "name1" ; "1" ; JSONstring  
)
```

Result:

```
{"name1": "1"}
```

Expression:

```
JSONSetElement ( "{}" ;  
  "name1" ; "1" ; JSONnumber  
)
```

Result:

```
{"name1": 1}
```

Expression:

```
JSONSetElement ( "{}" ;  
  "name1" ; 1 ; JSONboolean  
)
```

Result:

```
{"name1": true}
```



Punctuation

FileMaker	() [] ;
JSON	{ } , :



Expression:

```
JSONSetElement( "" ;  
  ["media.books[0].title" ; "Moby Dick" ; JSONString] ;  
  ["media.books[0].author" ; "Herman Melville" ; JSONString] ;  
  ["media.books[0].published" ; "1851" ; JSONString] ;  
  ["media.books[1].title" ; "1984" ; JSONString] ;  
  ["media.books[1].author" ; "George Orwell" ; JSONString] ;  
  ["media.books[1].published" ; "1949" ; JSONString] ;  
  
  ["media.movies[0].title" ; "Blade Runner" ; JSONString] ;  
  ["media.movies[0].director" ; "Ridley Scott" ; JSONString] ;  
  ["media.movies[0].release" ; "1982" ; JSONString] ;  
  ["media.movies[0].mainActor" ; "Harrison Ford" ; JSONString] ;  
  ["media.movies[1].title" ; "The Matrix" ; JSONString] ;  
  ["media.movies[1].director" ; "Wachowski Brothers" ; JSONString] ;  
  ["media.movies[1].release" ; "1999" ; JSONString] ;  
  ["media.movies[1].mainActor" ; "Keanu Reeves" ; JSONString] ;  
  
  ["media.actors[0].name" ; "Leonardo DiCaprio" ; JSONString] ;  
  ["media.actors[0].birthdate" ; "1974-11-11" ; JSONString] ;  
  ["media.actors[1].name" ; "Meryl Streep" ; JSONString] ;  
  ["media.actors[1].birthdate" ; "1949-06-22" ; JSONString] ;  
  
  ["preferences.colors.primary" ; "Blue" ; JSONString] ;  
  ["preferences.colors.secondary" ; "Green" ; JSONString] ;  
  ["preferences.colors.tertiary" ; "Red" ; JSONString] ;  
  
  ["metadata.version" ; "1.0" ; JSONString] ;  
  ["metadata.lastUpdated" ; Get(CurrentDate) ; JSONString] ;  
  ["metadata.source.application" ; "FileMaker Pro" ; JSONString]  
)
```

Result:

☒ Automatically evaluate

Evaluate

```
{"media":{"actors":[{"birthdate":"1974-11-11","name":"Leonardo DiCaprio"},  
{"birthdate":"1949-06-22","name":"Meryl Streep"}],"books":[{"author":"Herman  
Melville","published":"1851","title":"Moby Dick"},{"author":"George  
Orwell","published":"1949","title":"1984"}],"movies":[{"director":"Ridley  
Scott","mainActor":"Harrison Ford","release":"1982","title":"Blade Runner"},  
{"director":"Wachowski Brothers","mainActor":"Keanu Reeves","release":"1999","title":"The  
Matrix"}]},"metadata":{"lastUpdated":"9/18/2023","source":{"application":"FileMaker  
Pro"},"version":"1.0"},"preferences":{"colors":
```

Code Break

Just Kidding

We will get there though





We were here

```
Expression:
JSONSetElement ( "" ;
"name1" ; "1" ; JSONString ;
"name2" ; "2" ; JSONString
)|

Result:
```

Why No Workie?



Punctuation Marks

```
Expression:
JSONSetElement ( "" ;
"name1" ; "1" ; JSONString ;
"name2" ; "2" ; JSONString
)|

Result:
```

Nope

```
Expression:
JSONSetElement ( "" ;
    ["name1" ; "1" ; JSONString] ;
    ["name2" ; "2" ; JSONString]
)

Result:
{"name1":"1","name2":"2"}
```

[]

Yup



Real Example

Expression:

```
JSONSetElement ( "{" ;  
  ["layout" ; "Company List" ; JSONString ] ;  
  ["view" ; "Form" ; JSONString ] ;  
  ["sort" ; False ; JSONBoolean ]  
)
```

Result:

```
{"layout":"Company List","sort":false,"view":"Form"}
```

Expression:

```
JSONFormatElements (  
  JSONSetElement ( "{" ;  
    ["layout" ; "Company List" ; JSONString ] ;  
    ["view" ; "Form" ; JSONString ] ;  
    ["sort" ; False ; JSONBoolean ]  
  )  
)
```

Result:

```
{  
  "layout" : "Company List",  
  "sort" : false,  
  "view" : "Form"  
}
```



Define as Object

JSONSetElement ("{}"

- > Defined the element as a JSON object

JSONSetElement ("" ;

- > Let's FileMaker try to figure it out.



Let () us do more

Expression:

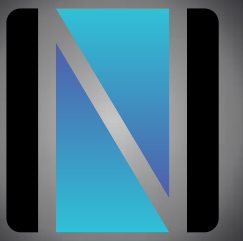
```
Let ($$json =  
  JSONSetElement ( "" ;  
    ["name1" ; "1" ; JSONString] ;  
    ["name2" ; "2" ; JSONString]  
  )  
; $$json )
```

Result:

```
{"name1": "1", "name2": "2"}
```

Code Break

Get



Current

Watch

Displays variables and fields in the current script.

Name	Value	Type
\$\$json	{"name1":"1","name2":"2"}	Global Variable

Expression:

```
JSONGetElement ( $$json ; "name1" )
```

Result:

1

Get



Expression:

```
JSONGetElement ( $$json ; "name1" )
```

Result:

1

Code Break



Get

- > Read All
- > Read Part
- > Read just one value



Other Things you should know

- > Dot Notation
- > Arrays
- > Format Elements
- > Get Element Type
- > List Keys
- > List Values
- > Delete Element

Code Break



Passing Parameters

- > Passing Multiple Parameters
 - > Get (ScriptParameter)
 - > Get (ScriptResult)



More...

- > Audit Logs (fmLog)
- > Record Snapshot
- > API Integration & Claris Connect
- > Configuration & Preferences
- > Dashboard data display
- > SearchResults 7



fmLog

- > Free tool: fmLog
 - > Leverages single field
 - > Yes, I wrote it. But it's free
 - > New version - vastly faster
 - > PSoS
 - > All JSON, all the time
 - > Take it apart to see the dashboard feature



Record Snapshot

```
ExecuteSQL ( “  
    SELECT FieldName  
    FROM FileMaker_Fields  
    WHERE TableName = ?” ; "" ; "" ;  
    Get ( LayoutTableName )  
    )
```



Record Snapshot

```
14 # SET UP LOOP
15 Set Variable [ $count ; Value: ValueCount ( $fieldList ) ]
16 Set Variable [ $i ; Value: 1 ]
17
18 # CREATE JSON OBJECT WITH ALL FIELDS AND VALUES
19 Loop
20     Exit Loop If [ $i > $count ]
21     Set Variable [ $field ; Value: GetValue ( $fieldList ; $i ) ]
22     Set Variable [ $value ; Value: GetField ( $table & "::" & $field ) ]
23     Set Variable [ $record ; Value: JSONSetElement ( $record ; $field ; $value ; JSONString ) ]
24     Set Variable [ $i ; Value: $i + 1 ]
25 End Loop
```

Name: \$json_out

Value: JSONSetElement ("{}" ;
["action" ; "Snapshot" ; JSONString] ;
["record" ; \$record ; JSONObject] // record data
)



API & Claris Connect

- > JSON is the language of the internet



Configuration & Prefs

- > Why? Endless flexibility
 - > Add and change preferences at will
 - > Nothing to change in schema
 - > Nothing to change in layouts
 - > Just Add Features



Dashboard Data Display

- > Store JSON in one field
- > Use Button Bars
 - > The Layout Calculation Widget™
- > Infinite flexibility



SearchResults 7

- > Free tool: SearchResults
 - > Yes, I wrote it. But it's free
 - > New version - vastly faster
 - > PSoS
 - > All JSON, all the time
 - > Take it apart to see the dashboard feature



Settings & Preferences

Metadata about the database is stored in two places:

- > JSON stored in one field
- > One record in the SearchResults table
- > Different records for different user configs
- > Each with endless setting possibilities

AI



A Note about ChatGPT 4

- > Extremely useful for learning anything new
- > Use it to explain JSON functions
- > Finds and corrects code mistakes
- > Great source of new ideas
- > Start a FileMaker conversation. It remembers.
- > Still prone to hallucination



SearchResults 7

- > Search everywhere, consolidate results
- > Preferences are stored in a single field
- > Settings are displayed on a ButtonBar
- > Found sets from multiple tables are stored in one JSON object
- > And it's free



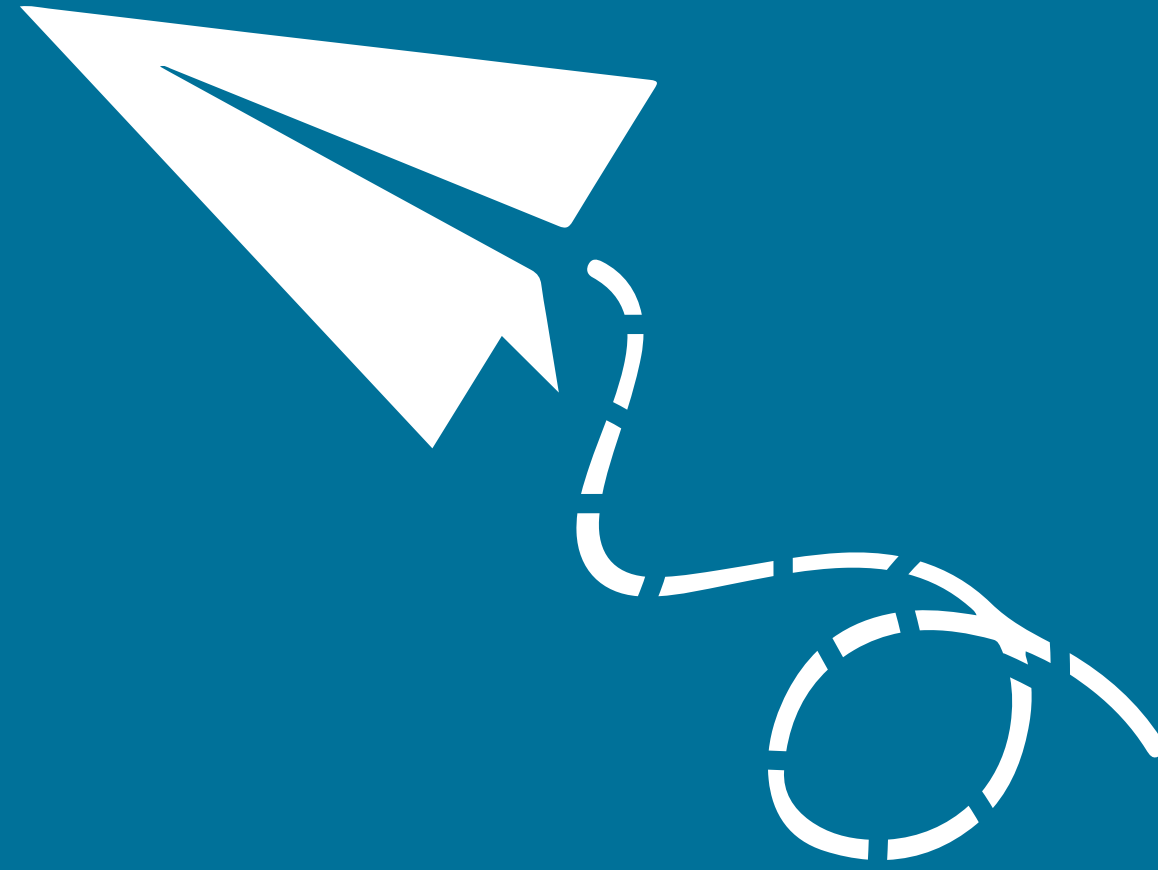
Part Deux

Show Your stuff

Share your screen on Zoom

Talk about the problem & JSON solution

Be brief. Or I'll cut you.



Thank You