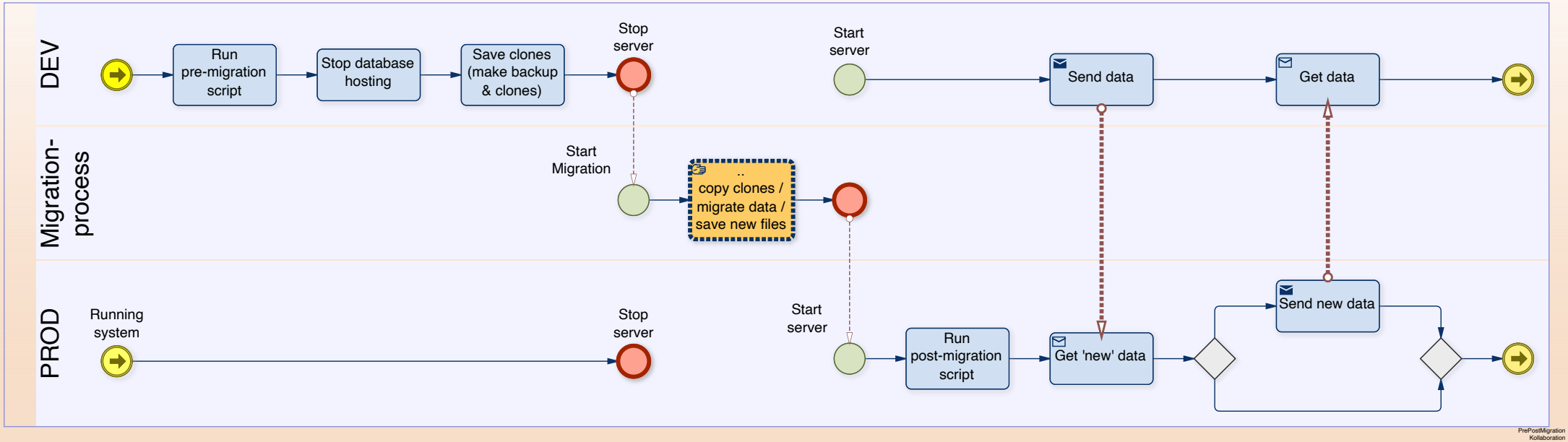
Pre- & Post-Migration Data-Transfer & -Manipulation

Heider

Michael Heider GmbH



Motivation

Heider

Michael Heider GmbH

With data migration, we are faced with the task of integrating it into a process with preparation and follow-up.

Motivation

Michael Heider GmbH

Before the migration

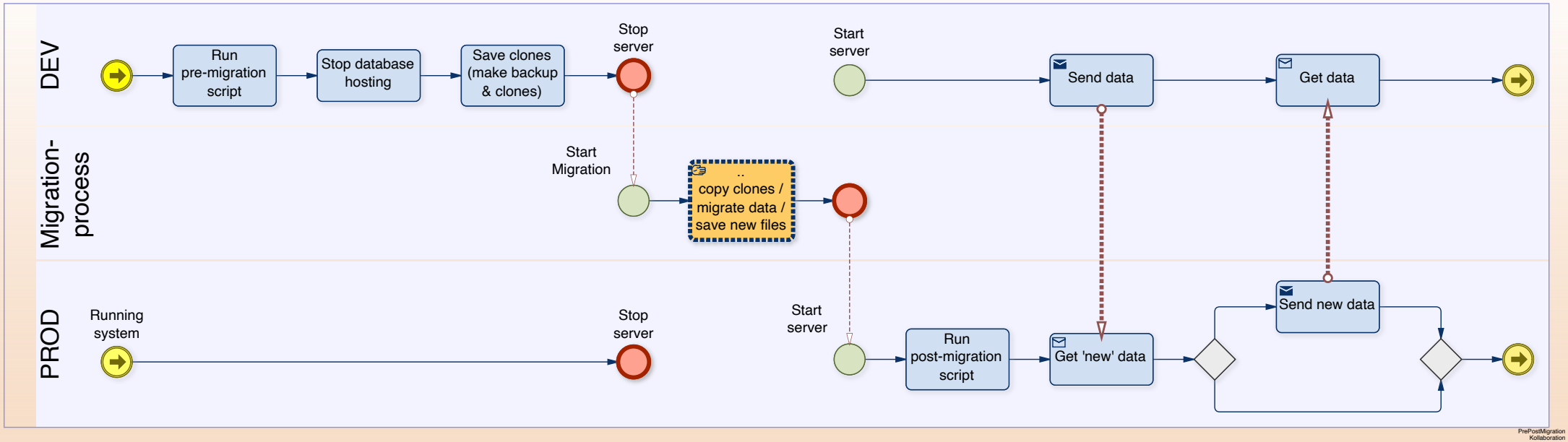
- Data must be prepared (via script).

After the migration

- Data must be post-processed (via script).
- Transfer an entire table contents from the development environment to the production environment
- Individual data must be transferred from the development environment to the production environment
  - e.g. for newly introduced settings fields
- Data must be set and updated manually after the migration

Process

Heider
Michael Heider GmbH



Motivation

- Multiple people can perform the migration.
- All modules have data that needs to be transferred from the development environment to the production environment.
- The database consists of multiple files, and not all files in the database are always migrated together.
- Different developers are responsible for different modules (or files).
- => Communication is needed about which files to include in the next migration.
- => Communication is required about what needs to be considered for each file.
- After migration, documentation must be created to show what has happened.

Motivation

➔ Goals

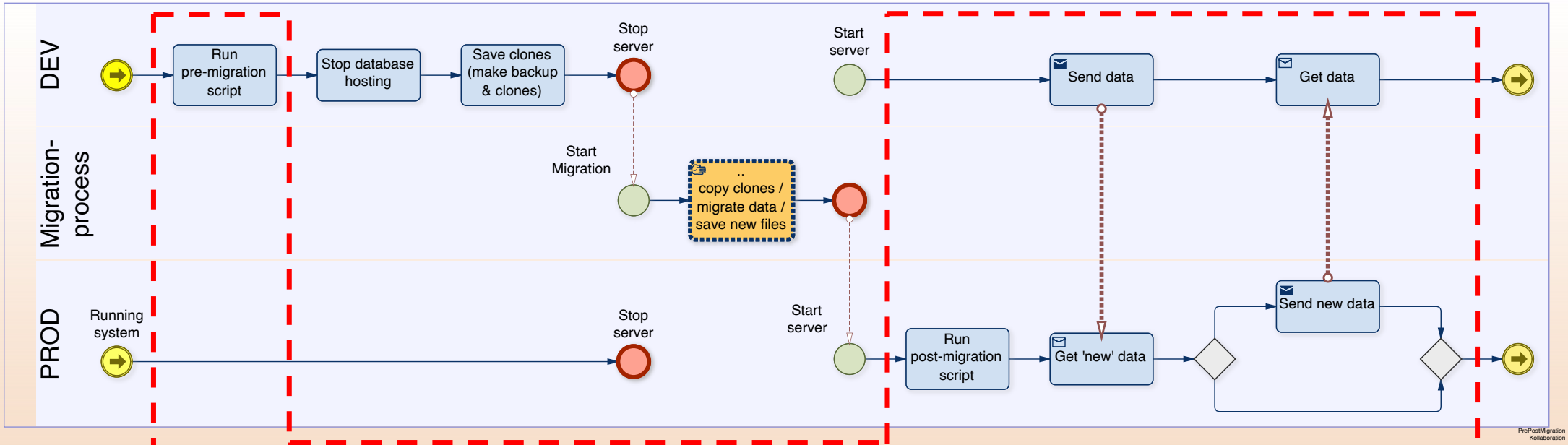
- Central preparation & execution & documentation
- NO connections between files from development environment to production environment (and vice versa)!!!

➔ Implementation

- REST API as a solution approach

Process

Heider
Michael Heider GmbH



Goal:
Automated process for a variable number of files, tables and fields.
Medium-term preparation and documentation of what is to be done during the next migration.
Afterwards: documentation of what has been done.

Implementation / First ideas

Commands

- Execute Script
- Get records / Get data
- Send records / Send data
- Delete records
- Comment (to the outline)

Execution

- Set command on / off
  ("To be run on next migration")
- Sort Order
- Person responsible for the command

Organisation

- Process
  - Pre-Migration
  - Post-Migration
- Environment
  - Development
  - Production
- Item
  - File
  - Layout (=Table)
  - Fieldname
  - (Data / Find-Query)

Implementation / First ideas

Michael Heider GmbH

Achieved goal

- Everyone can set for "their" modules which steps should be carried out before and after a migration.
- This setting can be changed for each migration.
- Migration can be carried out without further consultation.
- Automated process flow.
- Automated documentation of the processes

Implementation / What also was needed

Michael Heider GmbH

Commands necessary

- **Admin API**
  - Start Data APIs
  - Stop Data API
- Client: Script-Call via **fmp-Protocoll**
  - Activate Account with Data API privileges.
  - Deactivate Account with Data API privileges

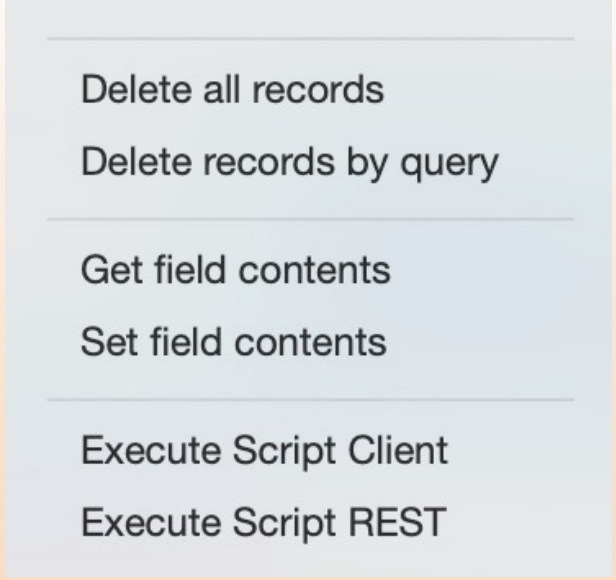
(For saftey reasons the Data API isn‘t always on and the accounts for using Data API should be inactive while not used!)

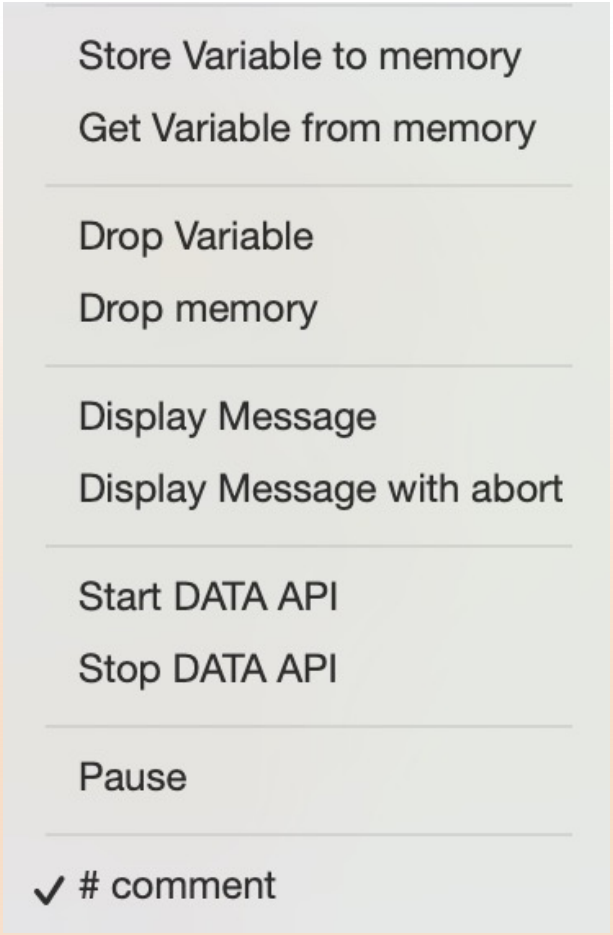
Commands ‚comfort‘

- Store data to memory
  - As JSON in Tool-File
- Get Data from memory
  - As JSON in Tool-File
- Display message
- Display message with abort
  - (If Function could not be executed)
- Pause

Implementation

Michael Heider GmbH





Examples

Michael Heider GmbH

Transfer all data of a table from the development environment to the production environment.

	PrePost	Activ	Sort	Environment	Command	Database	Layout	Name			Contents
▶	Post		100		# comment	Transfer all records in table "Protokoll" from development to production					
▶	Post	✓	110	Development	Get all records	heiProtokoll	REST__Protokoll		**; put to**	**Variable**	$$records
▶	Post	✓	120	Production	Delete all records	heiProtokoll	REST__Protokoll				
▶	Post	✓	130	Production	Send records	heiProtokoll	REST__Protokoll		**; get from**	**Variable**	$$records
▶	Post	✓	140		Drop Variable						$$records

Examples

Michael Heider GmbH

Transfer certain data records from a table from the development environment to the production environment

	PrePost	Activ	Sort	Environment	Command	Database	Layout	Name			Contents		FindQuery	
▶	Post		200		# comment	Replace or insert specific records in table "Protokoll" from development to production								
▶	Post	✓	210	Development	Get records by query	heiProtokoll	REST__Protokoll		; put to	Variable	$$records	; with Query	{"query":	JSON
▶	Post	✓	220	Production	Delete records by query	heiProtokoll	REST__Protokoll					; with Query	{"query":	JSON
▶	Post	✓	230	Production	Send records	heiProtokoll	REST__Protokoll		; get from	Variable	$$records			
▶	Post	✓	240		Drop Variable						$$records			

Examples

Michael Heider GmbH

Transfer content from a field in a table from the development environment to the production environment.

PrePost	Activ	Sort	Environment	Command	Database	Layout	Name			Contents		FindQuery	
Post		500		# comment	Replace specific field contents in table "Protokoll" in FIRST FOUND RECORD with entries from development to production								
Post	✓	510	Development	Get field contents	heiProtokoll	REST__Protokoll	Projekt	; put to	Variable	$$result	; with Query	{"query":	JSON
Post	✓	520	Production	Set field contents	heiProtokoll	REST__Protokoll	Projekt	; get from	Variable	$$result	; with Query	{"query":	JSON
Post	✓	530		Drop Variable						$$result			

Examples

Michael Heider GmbH

Overwrite the content of a field in the production environment with manual content.

PrePost	Activ	Sort	Environment	Command	Database	Layout	Name			Contents		FindQuery	
Post		600		# comment	Replace specific field contents in table "Protokoll" in FIRST FOUND RECORD with custom text								
Post	✓	620	Production	Set field contents	heiProtokoll	REST__Protokoll	Projekt	; get from	Custom	TestByHeider	; with Query	{"query":	JSON

Examples

Michael Heider GmbH

Start API & Script-calls

PrePost	Activ	Sort	Environment	Command	Database	Layout	Name		Contents	FindQuery
Post	✓	10		# comment	**Start PreMigration: REST-API on**					
Post	✓	20	Production	Start DATA API						
Post	✓	30	Development	Start DATA API						
Post	✓	40		# comment	**Enable all accounts & Test all databases.**					
Post	✓	50	Production	Execute Script Client	GDBS_MILSEELS		REST_MigrationMaster	**with Parameter**	{"sp":	JSON
Post	✓	60	Development	Execute Script Client	GDBS_MILSEELS		REST_MigrationMaster	**with Parameter**	{"sp":	JSON
Post	✓	70	Production	Execute Script REST	GDBS_MILSEELS		REST_MigrationMaster	**with Parameter**	{"sp":{"task":"Test"}}	JSON
Post	✓	80	Development	Execute Script REST	GDBS_MILSEELS		REST_MigrationMaster	**with Parameter**	{"sp":{"task":"Test"}}	JSON
Post	✓	90		Display Message with abort					Wenn eine Fehlermeldung	
Post	✓	100	Production	Execute Script REST	GDBS_MILSEELS		PostMigrationMaster	**with Parameter**		JSON
Post	✓	110		Pause					2	

Toolbox

Michael Heider GmbH

Long-term storage of content in the tool

- Writing data to a memory data set from a variable
- Reading data from a memory data set and writing it to a variable
- Delete memory data set

Post		1500	# comment	Storage				
Post	✓	1510	Store Variable to memory		records	<= memory | Variable =>	$$records	
Post	✓	1520	Get Variable from memory		records	<= memory | Variable =>	$$records	
Post	✓	1530	Drop memory		records			

Overwrite table, save some values

Heider

Michael Heider GmbH

PrePost	Activ	Sort	Environment	Command	Database	Layout	Name			Contents		FindQuery		Owner	TimeStampLastRun
Post	✓	2120	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailAbsender	; put to	Variable	$$mailAbsender	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2140	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailBCC	; put to	Variable	$$mailBCC	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2160	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailCC	; put to	Variable	$$mailCC	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2180	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailCCsendenWennNaehe	; put to	Variable	$$mailTage	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2200	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailText_Bestellung	; put to	Variable	$$mailText01	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2220	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailText_Reservierung	; put to	Variable	$$mailText02	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2240	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailText_Storno	; put to	Variable	$$MailText03	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2260	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	UhrzeitPfoertnerObligatoris	; put to	Variable	$$pfoertner	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2280	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	ZeitraumAnzeigeDringlich	; put to	Variable	$$zeit01	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2300	Production	Get field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	ZeitraumMailDringlich	; put to	Variable	$$Zeit02	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2320	Development	Get all records	GDBS_Mod_Raumplanu	REST___COLLECTION		; put to	Variable	$$data				hei	
Post	✓	2330	Production	Delete all records	GDBS_Mod_Raumplanu	REST___COLLECTION								hei	
Post	✓	2340	Production	Send records	GDBS_Mod_Raumplanu	REST___COLLECTION		; get from	Variable	$$data				hei	
Post	✓	2380		Drop Variable						$$data				hei	
Post	✓	2400	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailAbsender	; get from	Variable	$$mailAbsender	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2420		Drop variable						$$mailAbsender					
Post	✓	2440	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailBCC	; get from	Variable	$$mailBCC	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2460		Drop variable						$$mailBCC					
Post	✓	2480	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailCC	; get from	Variable	$$mailCC	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2500		Drop variable						$$mailCC					
Post	✓	2520	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailCCsendenWennNaehe	; get from	Variable	$$mailTage	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2540		Drop variable						$$mailTage					
Post	✓	2560	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailText_Bestellung	; get from	Variable	$$mailText01	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2580		Drop variable						$$mailText01					
Post	✓	2600	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailText_Reservierung	; get from	Variable	$$mailText02	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2620		Drop variable						$$mailText02					
Post	✓	2640	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	MailText_Storno	; get from	Variable	$$MailText03	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2660		Drop variable						$$MailText03				hei	
Post	✓	2680	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	UhrzeitPfoertnerObligatoris	; get from	Variable	$$pfoertner	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2700		Drop variable						$$pfoertner				hei	
Post	✓	2720	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	ZeitraumAnzeigeDringlich	; get from	Variable	$$zeit01	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2740		Drop variable						$$zeit01				hei	
Post	✓	2760	Production	Set field contents	GDBS_Mod_Raumplanu	REST___COLLECTION	ZeitraumMailDringlich	; get from	Variable	$$Zeit02	; with Query	{"query":[{"_a1":"1"}]}	JSON	hei	
Post	✓	2780		Drop variable						$$Zeit02				hei	

"Memory" for test purposes

Heider
Michael Heider GmbH

Before the test

	PrePost	Activ	Sort	Environment	Command	Database	Layout	Name		Contents
	P2E	✓	20		# comment	**Table MEMORY for testing purposes**				
▶	P2E	✓ 1	40	Development	Get all records	GDBS_Mod_Raumplanu	REST__ARB_Arbeitsbereich		**; put to Variable**	$$arb
▶	P2E	✓ 1	60		Store Variable to memory			arbeitsbereich	**<= memory | Variable =>**	$$arb
▶	P2E	✓ 1	80		Drop Variable					$$arb
	P2E	1	100		# comment	**... now do the testing, which edits all records ... and then go back**				
▶	P2E	1	120		Get Variable from memory			arbeitsbereich	<= memory | Variable =>	$$arb
▶	P2E	1	140	Development	Send records	GDBS_Mod_Raumplanu	REST__ARB_Arbeitsbereich		; get from Variable	$$arb
▶	P2E	1	160		Drop Variable					

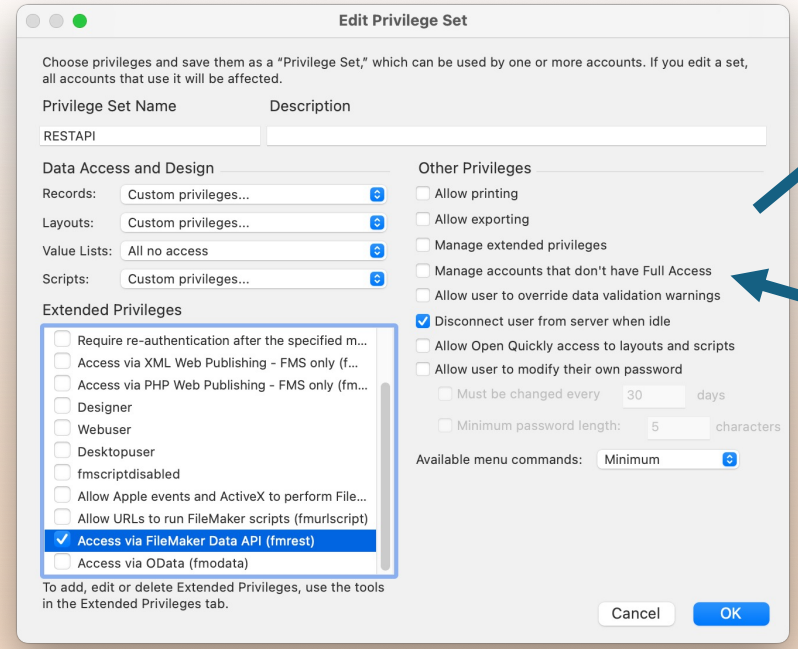
After the test

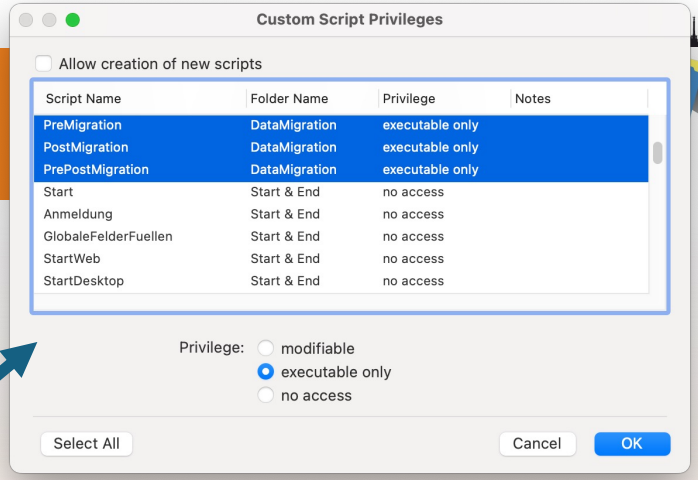
	PrePost	Activ	Sort	Environment	Command	Database	Layout	Name		Contents
	P2E	1	20		# comment	**Table MEMORY for testing purposes**				
▶	P2E	1	40	Development	Get all records	GDBS_Mod_Raumplanu	REST__ARB_Arbeitsbereich		; put to Variable	$$arb
▶	P2E	1	60		Store Variable to memory			arbeitsbereich	<= memory | Variable =>	$$arb
▶	P2E	1	80		Drop Variable					$$arb
	P2E	✓	100		# comment	**... now do the testing, which edits all records ... and then go back**				
▶	P2E	✓ 1	120		Get Variable from memory			arbeitsbereich	**<= memory | Variable =>**	$$arb
▶	P2E	✓ 1	140	Development	Send records	GDBS_Mod_Raumplanu	REST__ARB_Arbeitsbereich		**; get from Variable**	$$arb
▶	P2E	✓ 1	160		Drop Variable					

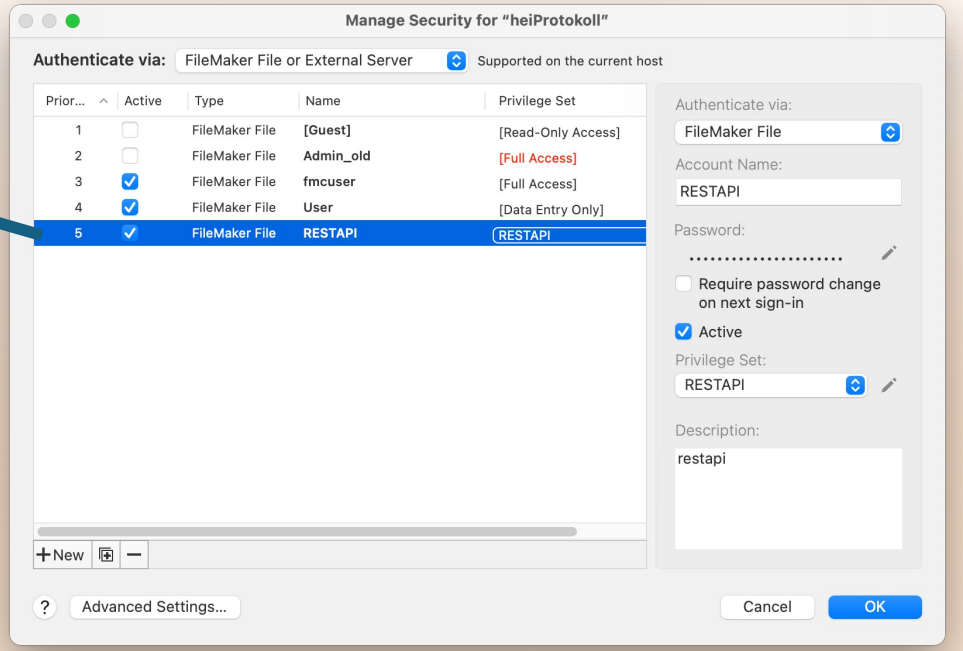
Prerequisites

Michael Heider GmbH

Privilege Set & Account







Problems / Considerations

Michael Heider GmbH

API access to each file requires its own token ➔

Each API call checks whether the corresponding token exists and fetches the new token if necessary.

(I do not establish a connection at the start of the process, but check for each API-call whether a token exists and is valid – see next point!)

Problems / Considerations

Heider

Michael Heider GmbH

```
21  If [ $api = "DATA" ]
22      Set Variable [ $token ; Value: JSONGetElement ( $$prePostMigrationTokensDATA_mdhaf ; $environment & "_" & $database ) ]
23  Else
24      Set Variable [ $token ; Value: JSONGetElement ( $$prePostMigrationTokensADMIN_mdhaf ; $environment ) ]
25  End If
26
27  If [ not _isValidJsonContent ( $token ) or IsEmpty ( $token ) ]
28      Set Variable [ $par ;
        Value: JSONSetElement (   "";   ["environment"; PrePostMigration::Environment; JSONString];   ["database"; PrePostMigration::Database; JSONString] ) ]
29      If [ $api = "DATA" ]
30          Perform Script [ Specified: From list ; "DATA_API_login" ; Parameter: $par ]
31      Else
32          Perform Script [ Specified: From list ; "ADMIN_API_login" ; Parameter: $par ]
33      End If
34      Set Variable [ $error ; Value: _sre ]
35      Set Variable [ $errortext ; Value: _sret ]
36      Set Variable [ $status ; Value: JSONGetElement ( _sr; "status" ) ]
37      If [ $error = 0 and $status = 200 and $api = "DATA" ]
38          Set Variable [ $token ; Value: JSONGetElement ( $$prePostMigrationTokensDATA_mdhaf ; $environment & "_" & $database ) ]
39      Else If [ $error = 0 and $status = 200 ]
40          Set Variable [ $token ; Value: JSONGetElement ( $$prePostMigrationTokensADMIN_mdhaf ; $environment ) ]
41      End If
42      If [ not IsEmpty ( $errortext ) ]
43          Set Variable [ $errortext ; Value: "Status: " & $status & "¶¶" & $errortext ]
44      End If
45  End If
```

Problems / Considerations

Token-Timeout === error 1627

- If the process takes a long time, the access tokens become invalid what results in error 1627. The old token is then deleted and the command is repeated (whereby a new token is automatically retrieved).

Problems / Considerations

ProcessOneRecord

```
92  # errorhandling
93  If [ $error = 1627 ]
94      Set Variable [ $par ;
        Value: JSONSetElement (   "";   ["database"; PrePostMigration::Database; JSONString];   ["environment"; PrePostMigration::Environment; JSONString] ) ]
95      Perform Script [ Specified: From list ; "ReleaseToken" ; Parameter: $par ]
96      Set Variable [ $error ; Value: _sre ]
97      If [ $error = 0 and $dontRepeat ≠ 1 ]
98          Set Variable [ $par ; Value: JSONSetElement (   "";   ["dontRepeat"; 1; JSONNumber];   ["always"; $always; JSONNumber] ) ]
99          Perform Script [ Specified: From list ; "ProcessOneRecord" ; Parameter: $par ]
100         Set Variable [ $error ; Value: _sre ]
101     End If
102 Else If [ $error = 0 ]
103     Set Field [ PrePostMigration::TimeStampLastRun ; Get ( CurrentHostTimestamp ) ]
104 Else If [ $error = -1 ]
105     Set Field [ PrePostMigration::TimeStampLastRun ; Get ( CurrentHostTimestamp ) ]
106     Halt Script
107 Else If [ $error = -11 ]
108     Show Custom Dialog [ "Something went wrong!" ; "This command can't be performed!" ]
109     If [ Get ( LastMessageChoice ) = 2 ]
110         Halt Script
111     End If
112 End If
```

Problems / Considerations

Michael Heider GmbH

Scriptcall via fmp-protocoll

- The called script is written to the script stack and executed **after** all running scripts have been processed ➔ The running script loop must be canceled and a new call of the script loop must also be placed on the script stack to continue afterwards.

Problems / Considerations


```
15      Loop [ Flush: Always ]
16          Perform Script [ Specified: From list ; “ProcessOneRecord” ; Parameter: _sp ]
17          Set Variable [ $error ; Value: _sre ]
18          If [ $error ≠ 0 ]
19              Perform Script [ Specified: From list ; “Errorhandling” ; Parameter: JSONSetElement ( “”; "error"; $error; JSONNumber ) ]
20              Set Variable [ $error ; Value: _sre ]
21          End If
22          If [ Position ( PrePostMigration::Command; "Client"; 1; 1 ) > 0 ]
23              Set Variable [ $url ; Value: "fmp://" & Let ( ip = Get ( HostIPAddress ); If ( IsEmpty ( ip ); "~/"; ip )) & "/" & Get ( FileName ) & "?" & "script=" & "… ]
24              Open URL [ With dialog: Off ; $url ]
25              Set Variable [ $error ; Value: Get ( LastError ) ]
26              Exit Loop If [ True ]
27          End If
28          If [ #Global::IsStepwise and PrePostMigration::IsActive = 1 and PrePostMigration::fln__IsComment ≠ 1 and not ( $count = Get ( RecordNumber)) ]
29              Set Field [#Global::gt__Text ; "Process is paused after performing action of step " & PrePostMigration::SortOrder     //Get ( RecordNumber ) ]
30              Set Field [#Global::gn__Number ; 1 ]
31              Commit Records/Requests [ With dialog: Off ]
32              Pause/Resume Script [ Indefinitely ]
33              Set Field [#Global::gt__Text ; “” ]
34              Set Field [#Global::gn__Number ; “” ]
35              Commit Records/Requests [ With dialog: Off ]
36          End If
37          Exit Loop If [ Get ( RecordNumber ) >= $count ]
38          Go to Record/Request/Page [ Next ; Exit after last: Off ]
39          Refresh Window [ ]
40      End Loop
```

Problems / Considerations

Michael Heider GmbH

Scriptcall via fmp-protocoll

- Calling a script via the fmp protocol cannot return a script result.
  This means that an error must be reported in the called script.

Problems / Considerations

Re-Login via REST

- When calling a script via REST, a re-login is not valid! No error message is thrown, but the authorizations are not changed.

Problems / Considerations

Michael Heider GmbH

Scriptcalls via REST

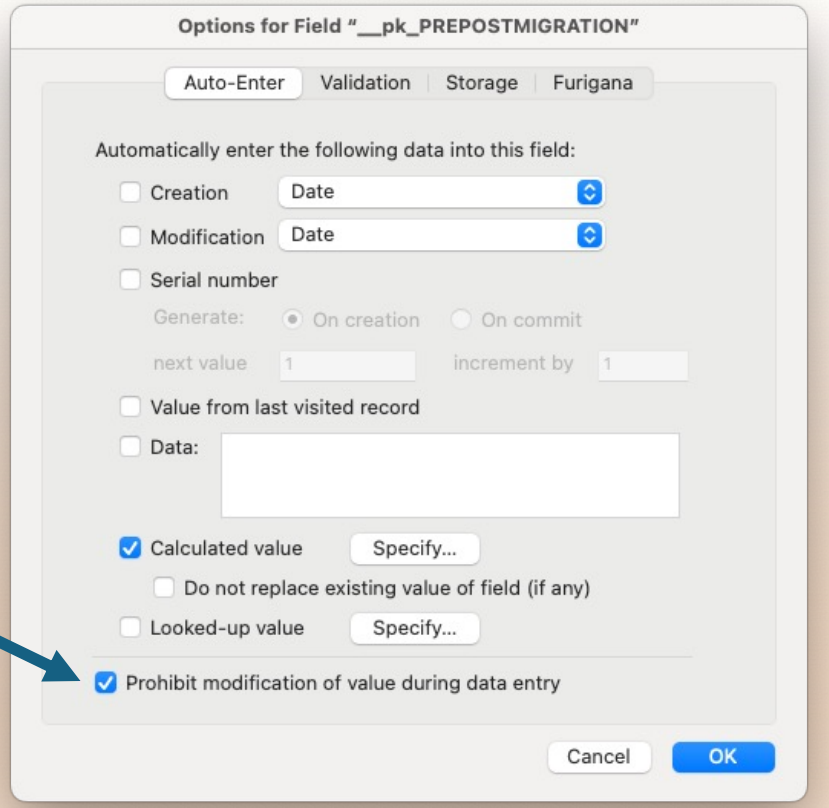
- Scripts that are not part of the set of authorized scripts for REST calls can be executed if they have the "Execute with full permissions" option.

Problems / Considerations

Michael Heider GmbH

Fieldoptions

- The option “Prohibit modification of value during data entry" is an exclusion criterion for data change / data entry via REST.



Last but not least: Security

Heider

Michael Heider GmbH

REST is a convenient way of transferring data from the development environment to the production environment (and vice versa) and for automate this process.

But is it also secure?

- We are in a "shielded" environment.
- REST API is only opened for the migration process and then closed again
- REST accounts are only activated for the migration process and otherwise deactivated.

Discussion

Heider
Michael Heider GmbH

PrePost	Activ	Sort	Environment	Command	Database	Layout	Name		Contents	
Pre	✓	10		# comment	**Start PreMigration: REST-API on**					
Pre	✓	20	Development	Start DATA API						
Pre	✓	30		# comment	**Enable all accounts & Test all databases.**					
Pre	✓	40	Development	Execute Script Client	GDBS_MILSEELS		REST_MigrationMaster	**with Parameter**	{"sp":	JSON
Pre	✓	50	Development	Execute Script REST	GDBS_MILSEELS		REST_MigrationMaster	**with Parameter**	{"sp":{"task":"Test"}}	JSON
Pre	✓	60		Display Message with abort					Wenn eine Fehlermeldung	
Pre	✓	70		# comment	**Call PreMigration script**					
Pre	✓	80	Development	Execute Script REST	GDBS_MILSEELS		PreMigrationMaster	**with Parameter**	{"sp":{"silent":1}}	JSON
Pre	✓	90		Display Message with abort					Wenn eine Fehlermeldung	
Pre	✓	100		# comment	**Set everything OFF**					
Pre	✓	110	Development	Execute Script Client	GDBS_MILSEELS		REST_MigrationMaster	**with Parameter**	{"sp":	JSON
Pre	✓	120	Development	Stop DATA API						
Pre	✓	130		# comment	**Test should get "Status 502 Bad gateway" => OK, API is off**					
Pre	✓	140	Development	Execute Script REST	GDBS_MILSEELS		REST_MigrationMaster	**with Parameter**	{"sp":{"task":"Test"}}	JSON