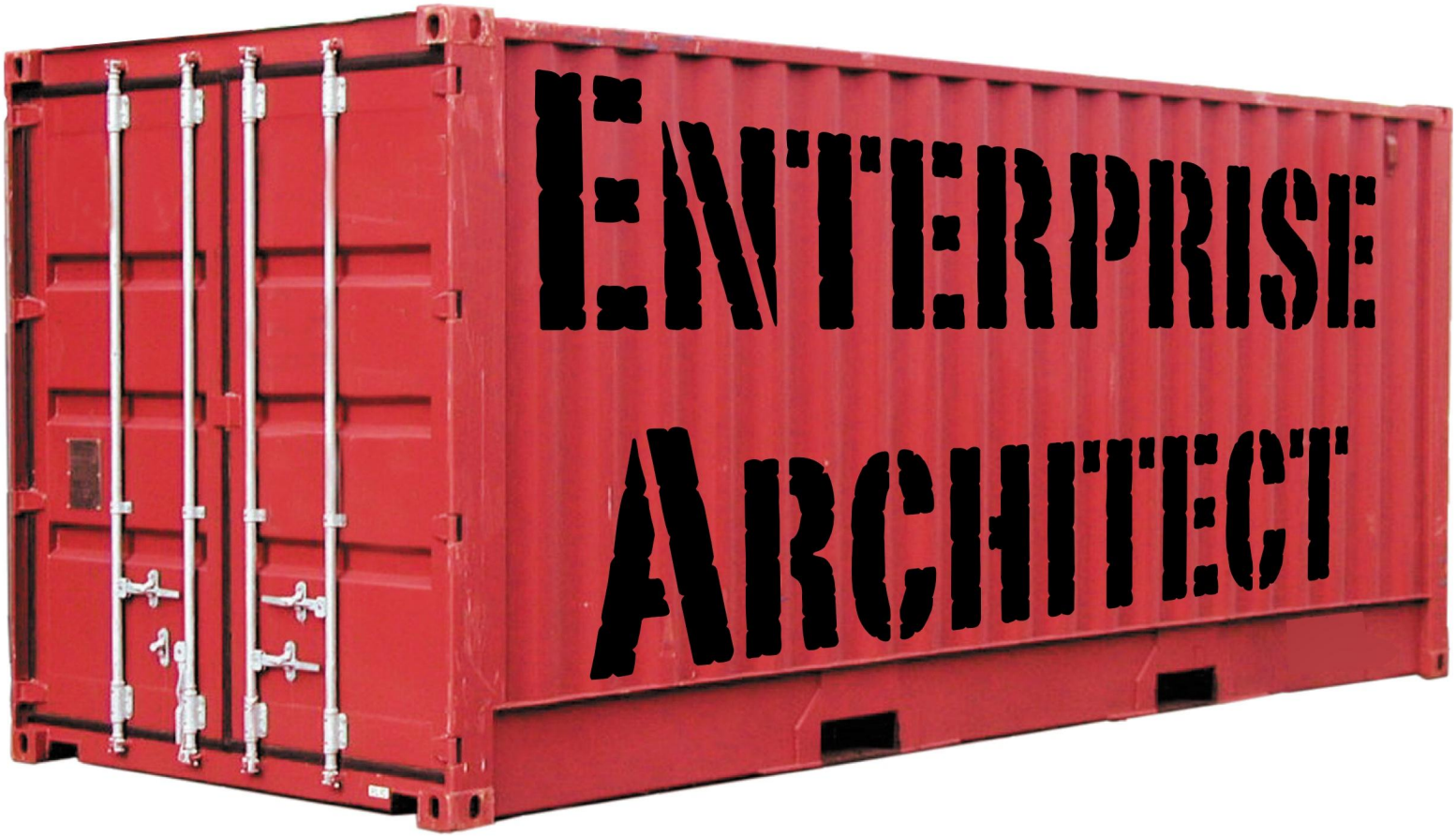
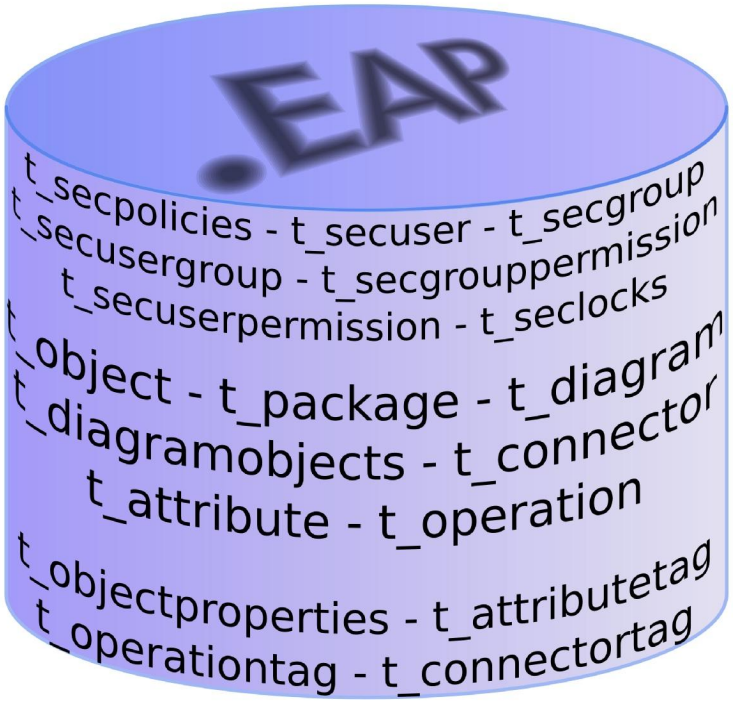
Inside



Querying EA's Database



By Thomas Kilian

Inside Enterprise Architect

Querying EA's Database

Thomas Kilian

This book is for sale at http://leanpub.com/InsideEA

This version was published on 2023-09-19

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Contents

1. Preface

Enterprise Architect[1] (EA) offers a wealth of API functions to support automated manipulation of UML models. However, quite a number of tasks require actions not directly supported by the API. Here the fact comes handy that EA is based on a database model which has proven to be very stable with respect to its structure. The last major change was introduced with audit functionality which added a couple of new tables but left the structure of the existing tables untouched. So you can assume that your add-ins will run in future versions of EA if you follow a few rules.

The contents of this book is the essence of a continuous work with EA since end 2003. It surely lacks prose but likely you won't need that anyway. I'd call it a hacker's guide into EA[2].

Special thanks to Peter Doomen who inspired me to write this book. You likely might be interested in his book Fifty Enterprise Architect Tricks[3]. Also I like to thank Helmut Ortmann for supplying me with most of the query examples and a couple of hints which had passed my attention[4]. Probably I should mention a couple of other guys[5] but I'm not going to bother you with my family history.

This book starts with a short introduction on how to query EA's database. This is followed by a concise list of all available tables and details for the most important ones. The details contain cross references into more details as well as screen shots of the GUI where the appropriate elements appear. Vice versa the screen shots point to the according table columns. The final sections conclude with a practical approach to using SQL in Enterprise Architect.

[1] The EA version used to create this book was actually 9.3 (build 930). However, most of the references are also valid for earlier versions of EA.

[2] Not all tables/columns are clear in their meaning (to me). A ?! mark is placed where this is the case. Comments about clarification of their meaning are welcome! Just send me a mail to thomas.kilian@me.com.

[3] http://leanpub.com/entarch

[4] The latest finding by Colin Wood has been fixed December 2017. Thanks for notifying me.

[5] Cheers to Paolo and all the supporters at Sparx.

2. Copyright and Disclaimer

Also all of the information in this book has been tested by me in many circumstances I can not hold any liability for use of the here presented information[1]. However, I'd be glad to receive any kind of feedback to correct future updates of this book which you will receive for free in turn. Having said this, all information presented here is subject to change without notice.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Important note: this book is about **querying** EA's database. You might think that updating the database is easy with an UPDATE statement. Sure. But that's playing with a loaded and unsecured weapon! You might shoot yourself in your knee or even in the middle of your heart. If you are going to change your repository: use the API.

[1] I really loathe writing such legal blurb since it should be obvious. By the way: German Law applies! (Does that change anything?)

3. Accessing the Database

The lowest layer in EA is that of its database. When you first start with EA you will most likely deal with EAP files. A simple though not official fact is: EAP is MS Access. So if you want to play around just open one of those EAP files and see what MS Access is telling you. If you are using a Corporate license you will most likely use a more advanced SQL server. Be it MS SQL Server, Oracle, MySQL or whatever. In that case you need some client software to perform manipulations.

Before doing so you should get familiar with the database in a more simple way.

3.1 Inspecting EA's Tables

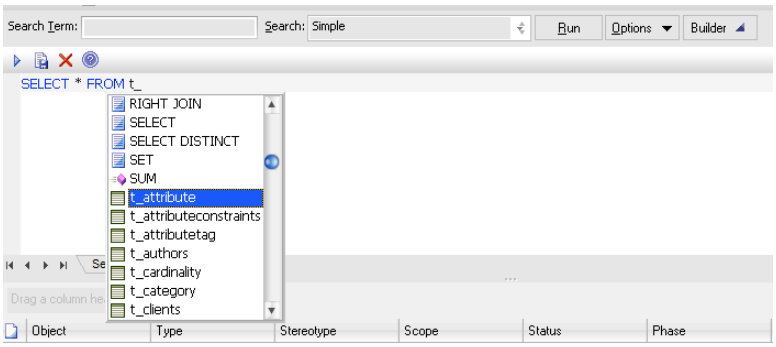
The most simple way is to open the respective EA repository with EA itself.

Access seems to be picky with some columns and suppresses them. So preferably you use EA like explained here to actually inspect the database.

Use the EAExample.EAP which comes with your EA installation. Now press `Ctrl-F` to open the search window. Click the Builder button and select the SQL tab. Type the text

```
1          SELECT * FROM t_
```

and press Ctrl-Blank. You should now be presented with the following:



Search Window

As you can see this is a list of tables which reside in EA's database. For a start let's choose one of the important tables: `t_object`. After pressing the Run button or the little triangle top left you will get a list of all elements present in the repository. Obviously the column `Object_Type` is the type of an element and Name its name. Simple! You might go on with `t_package` to see details of all packages in the repository.

3.2 Ways to Query Tables

The clean way to query the tables is the API. This is recommended for most cases. However, there's also a demand to be able to access these data where the API is simply too slow. EA is kind of object oriented in how it queries its database. That means for a global change (like changing the status) it will issue single UPDATEs instead of a compound. Of course you can query the database much faster with an intelligent query than any iterative API calls.

Anticipating a simple structure element of the table containing the elements (`t_object`) the best way to retrieve an element is by

```
elem = Repository.GetElementByID(4711);
```

which yields the 'same' element as

```
SELECT * FROM t_object WHERE Object_ID = 4711
```

where just the API object data are cooked while those of the SQL are raw. So that's no wizardry. But imagine you need all elements with a certain stereotype. While in the API this would need quite some programming effort, the SQL is simple:

```
SELECT * FROM t_object WHERE stereotype = "stereo"
```

Now it's upon your imagination what you can do by joining different tables in SQL. A few sample can be found in this chapter at the end of this book.

Note:
This window accepts only *SELECT* statements. Any other SQL (like e.g. *UPDATE*) is silently ignored!

3.3 A List of All Tables

Name	Description
t_attribute	Attributes defined for elements
t_attributeconstraints	Constraints for attributes
t_attributetag	Tagged values for attributes
t_authors	List of authors defined with `Settings/Project Types/People/Project Author(s)`
t_cardinality	List of cardinalities defined with `Settings/UML Types/Cardinality Value`
t_category	Legacy ?!
t_clients	List of authors defined with `Settings/Project Types/People/Project Clients`
t_complexitytypes	Legacy ?!
t_connector	Connectors between elements

Name	Description
t_connectorconstraint	Constraints for connectors
t_connectortag	Tagged values for connectors
t_connectortypes	This table is used for the connector metatypes shown in the profile dialog. Also t_connector.Connector_Type can contain just elements from this table.
t_constants	Various key/value pairs in misc. dialogues
t_constrainttypes	List of constraint types defined with `Settings/Project Types/General/Constraint`
t_datatypes	Definitions from `Settings/Code Datatypes`
t_diagram	Diagram properties
t_diagramlinks	Non-standrad links appearing in diagrams
t_diagramobjects	Diagram elements
t_diagramtypes	Legacy ?!
t_document	Contents of linked documents, baselines and more
t_ecf	List of complexity factors defined with `Settings/Project Types/Estimation Factors/Environment...`
t_efforttypes	List of effort factors defined with `Settings/Project Types/Project Indicators/Effort`
t_files	?!
t_genopt	Various options
t_glossary	The system glossary
t_html	Some HTLM strings for the doc generation
t_image	The alternate pictures defined with `Settings/Images...`
t_implement	Legacy ?!
t_issues	The system issues defined with `View/More Project Tools/Project Information/Issues`
t_lists	Status Types defined with various `Settings/Project Types/General Types/...` tabs
t_mainttypes	?!
t_method	Legacy ?!
t_metrictypes	List of metric factors defined with `Settings/Project Types/Project Indicators/Metric`
t_object	Basic UML elements
t_objectconstraint	Constraints for elements
t_objecteffort	Something related to project estimation
t_objectfiles	The files linked in the properties/files for elements
t_objectmetrics	Something related to project estimation
t_objectproblems	?!
t_objectproperties	Tagged values
t_objectrequires	The internal requirements defined for elements
t_objectresource	Something related to project estimation
t_objectrisks	Something related to project estimation
t_objectscenarios	The use case scenarios
t_objecttests	Tests defined for elements
t_objecttrx	?!
t_objecttypes	EA internal rendering support
t_ocf	Something related to project estimation
t_operation	Operations for elements
t_operationparams	Parameters for operations

Name	Description
t_operationposts	Postconditions for operations
t_operationpres	Preconditions for operations
t_operationtag	Tagged values for operations
t_package	Package container
t_palette	Legacy ?!
t_paletteitem	Legacy ?!
t_phase	Legacy ?!
t_primitives	Code primitive types
t_problemtypes	Problem types defined with `Settings/Project Types/Maintenance/Problem Types`
t_projectroles	Values from `Settings/People/Project Roles`
t_propertytypes	Predefined tagged values
t_requiretypes	List of requirement types defined with `Settings/Project Types/General/Requirements`
t_resources	List of resource defined with `Settings/Project Types/People/Resources`
t_risktypes	List of risk factors defined with `Settings/Project Types/Project Indicators/Risk`
t_roleconstraint	Legacy ?!
t_rtf	EA internal doc generation settings
t_rtfreport	Some doc generation settings
t_rules	Related to model validation ?!
t_scenariotypes	List of scenario types defined with `Settings/Project Types/General/Scenario`
t_script	Local scripts
t_secgroup	Security groups
t_secgrouppermission	Security group permissions
t_seclocks	Security locks
t_secpermission	?!
t_secpolicies	Security settings
t_secuser	Security users
t_secusergroup	Security user/group assignments
t_secuserpermission	Security user permissions
t_snapshot	Audit log
t_statustypes	List of status types defined with `Settings/Project Types/General/Status`
t_stereotypes	Stereotypes
t_taggedvalue	Smorgasbord for WSDL model elements. These are NOT the tagged values. Instead use t_objectproperties
t_tasks	The system tasks
t_tcf	List of complexity factors defined with `Settings/Project Types/Estimation Factors/Technical...`
t_template	RTF templates ?!
t_testclass	?!
t_testplans	This table is not currently used by EA[1]
t_testtypes	List of test factors defined with `Settings/Project Types/Maintenance/Test Types`
t_trxtypes	Matrix Profile; Painter Settings; Auto counters defined with `Setting/Auto Names and Counters...` more ?!

[1] As of EA version 9.3

Name	Description
t_umlpattern	UML patterns imported via `View/More Project Tools/Project Resources/UML Patterns`
t_version	?!
t_xref	This and that
t_xrefsystem	Various profiles
t_xrefuser	?!
usys_system	Key-value pairs for repository wide settings
usystables	A list of all the tables above along with the version where they were introduced. This table is needed (only?) during a project transfer.
usysoldtables	I have not the faintest idea ?!
usysqueries	I have not the faintest idea ?!

4. Most Important Tables

In this chapter we are going into quite some details of the most important tables. Namely these are that for elements, packages, diagrams, diagram objects, connectors and tagged values. Those are the ones you most likely need to retrieve often.

The single columns have a short description of what I think is their meaning. Some are obvious, some are just smoke signals. A reference to the GUI screen shots is placed where this is possible.

To improve readability and reference the properties were either split into several sub-tables or the table has an *indicator* on top of a logical section. Each sub-table is sorted alphabetically according to the name in `Column`.

4.1 More things than you find in the Project Browser: t_object

As you already know, this table holds all elements stored in the repository. That is any element you can see in the project browser plus those not shown like notes, boundaries and a couple of other elements. Please note that the Package element is a pendant[1] to the Package itself. Part of the information in both is redundant and both link to each other.

The following table lists the most important properties also to be found in the General properties window. The remaining properties are listed subsequently in logical groups.

Column	Description
Alias	Alias property
Author	Author property
Complexity	Complexity property Valid values are: *1* for Easy, *2* for Medium, *3* for Difficult
GenType	Language property String value. Note that this value does not appear unless it is defined as Product in the Language Datatypes
Name	Name property
Notes	Notes property
PDATA5	Keywords property
Phase	Phase property String value
Scope	Scope property String value
Status	Status property corresponds to values in `t_statustypes`
Stereotype	Stereotype property
Version	Version property

[1] I have detailed this in my book Scripting EA.

In order to retrieve the `Notes` column with EA's SQL Search you need to name the column like this:

```
1  SELECT t_object.Note AS [Notes], * FROM t_object
```

The `Note` column will not appear in the `*` nor when selected unnamed. EA will filter all columns named `Note`. The square brackets around `Notes` are meant for use in EAP files and might have to be removed for other DB providers.

This warning applies not only for the table `t_object` but all tables which have a `Note` column.

4.1.1 Key links

Column	Description
Diagram_ID	Only for Text elements; reference to primary key of the diagram
ea_guid	A global UID shown here Use `Repository.GetElementByGUID (ea_guid)` to retrieve this element
Object_ID	Primary, unique key for the element Use `Repository.GetElementByID (Object_ID)` to retrieve this element
Package_ID	Primary key of the package where the element is located
ParentID	Only for nested elements: primary key of the object
Classifier_guid	See below

4.1.1.1 Classifier_guid

For `ActivityParameter` elements the `Classifier_guid` encodes the type of the parameter. This can eventually be one of

{EABOOL00-B653-4f3c-A010-30205D67F5F5}
{EAINT000-B653-4f3c-A010-30205D67F5F5}
{EAREAL00-4339-434b-BC17-A5E1FDC63F6C}
{EASTRING-B653-4f3c-A010-30205D67F5F5}
{EAUNAT00-B653-4f3c-A010-30205D67F5F5}

And now guess…

Another use came with the introduction of connectors from connectors. These create elements with object type `ProxyConnector`. Here the `Classifier_guid` holds the GUID of the corresponding connector.

4.1.2 Details

Column	Description
Abstract	Abstract property
Cardinality	Cardinality property String value
Concurrency	String equivalent of the Concurrency property for class elements
IsActive	Boolean values for the Is* properties
IsLeaf	ditto
IsSpecification	ditto
IsRoot	ditto
Persistence	Persistence property String value

4.1.3 Dock

Column	Description
CreatedDate	Created property
GenFile	Filename property String value
ModifiedDate	Modified property
Multiplicity	Multiplicity property String value

4.1.4 Appearance

Column	Description
Backcolor	Background Color property RGB values in decimal
Bordercolor	Border Color property RGB values in decimal
BorderStyle	For frame-like elements (boundaries etc.). Corresponds to the style (0..3) where *3* = solid line
BoderWidth	Border Width property
Fontcolor	Font Color property RGB values in decimal
StyleEx	Individual font settings

4.1.5 `Object_Type` and `NType`

Another classical database design anti-pattern can be found in the two columns `Object_Type` and `NType`. While `Object_Type` is a string value, `NType` adds forgotten salt to any of the expressions depending on some additional context informtion.

For `Object_Type` EA seems to mangle the display value in a query. In particualar this happens with a BPMN2.0 `StartEvent`. Other cases may exist. If you have a model with such a `StartEvent` the `Object_Type` contains `Event` but the SQL result is displayed as `StartEvent`. You can verify this:

```
1  SELECT count(*),object_type FROM t_object GROUP BY object_type
```

will show that actually there are `Object_Type = 'Event'` rows. Now

```
1  SELECT * FROM t_object where object_type = 'Event'
```

will find that row(s) but display `StartEvent` instead of `Event`!

The meaning of `NType` per `Object_Type` is hard to structure due to its context sensitivity. So here are a couple of statements:

- If `Object_Type` is one of `Activity`, `Artifact`, `Class`, `Interaction`, `Requirement`, `State`, `StateMachine`, `UseCase` (and probably some more) and `NType` is 8 and `PDATA1` has a number greater then zero then `PDATA1` corresponds to t_diagram.Diagram_ID of the composite diagram.
- `Artifact` and 32 means that it renders an image. Unfortunately I have no idea where that image comes from.
- `NType` has 0 or 1 for lots of plain elements. It's not obvious what might be the difference.
- `Constraint` and `MessageEndpoint` seem to have 2 for `NType`.
- The following table lists a couple of further combinations[2]:

Object_Type	NType	Meaning
Text	0	plain text
	18	Diagram Note
	19	Hyperlink; `Name` has `$help://`, `$inet://`, etc.
	76	Legend
	82	Diagram Hyperlink; `PDATA1 == t_diagram.Diagram_ID`
Event	0	Send
	1	Receive
	2	Accept timer
UMLDiagram	0	Frame; `PDATA1 == t_diagram.Diagram_ID`
	1	Diagram reference; `PDATA1 == t_diagram.Diagram_ID`
StateNode	3	Initial
	4	Final
	5	History
	10	Junction
	11	Choice
	13	Entry point
	14	Exit point
	15	Deep history

[2] Including all would make the table an unreadable mess.

Object_Type	NType	Meaning
	100	Activity initial
	101	Activity final
	102	Flow final
InteractionFragment	0	alt
	1	opt
	2	break
	3	par
	4	loop
	5	critical
	6	neg
	7	assert
	8	strict
	9	seq
	10	ignore
	11	consider
ConditionalNode	18	makes no sense at all

4.1.6 Misc

Column	Description
Classifier	NULL or *0* where not defined. If > *0* then it is the primary key of the element which classifies the element
EventFlags	semi-colon separated list of attributes with links into the Risk/Metrics/etc. tables
GenOptions	Some very nasty semi-colon separated list of attributes (used for code generation?!)
GenLinks	String value of the class which this one is specialized from Usually only set where classes are reverse engineered and the general class is missing
Header1/2	Used for code generation
PDATA1	For Package elements: primary key of the package For Elements: Same as the Status column For Parts/Instances: GUID of the classifier For UseCase: *#EXP#=<ep>;* semi-colon separated list of Extension Points *<ep>* For Notes: linked element feature name For Text displaying as hyperlink: t_diagram.Diagram_ID For Requirements: Status property For UMLDiagram: Diagram_ID of the underlying diagram;
PDATA2	For Elements: Same as the Priority column For Notes: Object_ID of the linked feature element For Requirements: Priority property
PDATA3	For Elements: Same as the Difficulty column For Notes: Reference name into the linked feature element For Ports in classified Parts: the GUID of the corresponding Port in the Classifier. For State: t_diagram.Diagram_ID of the composite diagram For Requirements: Difficulty property

Column	Description
PDATA4	For Note elements: *Yes* if the note is linked to an element feature and *idref=<val>;* list where *<val>* is the primary key of the connector(s) to which the note is linked For elements: If *> 0* this is the primary key of the connector for which this element is defines as association class
RunState	For objects a list of run state variables
Style	see below
Tagged	Flag to show that an element it bookmarked (little red triangle in diagram)
TPos	Tree order of the element in the project browser

I have no idea why the column Style is called by that name since only StyleEx really has some character style information. Alas, this is a semi-colon separated list of attribute assignments. It's a Sparxian zoo where all sorts of homeless animals can gather. For example the `Locked=true;` appears whenever an element has been marked on a diagram with the `Is Locked` context. The element is now protected from manual changes in the GUI (and even the API, though it does not tell the reason for an update failure in `GetLastError`) and the element shows a red excalamtion mark in the browser. Another on indicates a linked document if *= MDoc=1;* There are some more options used like `ShowBeh`, `EScrpt` and `LinkOpen` which seem to be too exotic to be explained here (currently).

4.1.7 Unknown or heritage

Column	Description
Effort	Always *0* ?!
Visibility	Always NULL ?!
StateFlags	Always NULL ?!
PackageFlags	Always NULL ?!
ActionFlags	Always NULL ?!

... omitted ...

5. Element Feature Tables

Mainly the element features comprise attributes and methods. Both are stored in the tables detailed below.

… omitted …

6. Tagged Value Tables

Tagged values are not stored in `t_taggedvalue` but a couple of different tables.

… omitted …

7. Security Related Tables

The following tables are only relevant if user security has been turned on. In order to check whether security is turned on in the repository the `t_secpolicies` table must be queried.

… omitted …

8. Rarely Used Tables

Here you will find some details about tables which are not of major importance. However, from time to time you will also need to deal with them. Note that this section is going to be populated with more information during the next near future.

… omitted …

9. Marvelous References

This chapter might be the start of a new book. Or maybe it will stay thin because the treasures inside the `t_xref` table are too secret to be uncovered. We will see.

However, here are some bits from this marvelous table. Currently they are about stereotypes and MDG profiles but there's a lot more hidden.

… omitted …

10. API Cross References

This chapter contains a cross reference from table columns to object properties in the API[1].

The references are presented for both directions. As you will notice not all columns map to an API property and vice versa. The table also omits EaCollections as those are a result of a query itself and not a simple column.

I have currently only included the two most important table `t_package` and `t_object`.

The API help tells that `Created` and `Modified` are writeable. However, neither will be saved on an `Update()`. `Created` remains unchanged and `Modified` will be set to the current time stamp. In order to change them you must bypass the API.

10.1 t_package — EaPackage

t_package	EaPackage	EaPackage	t_package
BatchLoad	BatchLoad	Alias	t_object.Alias
BatchSave	BatchSave	BatchLoad	BatchLoad
CodePath	-	BatchSave	BatchSave
CreatedDate	Created	Connectors	-
ea_guid	PackageGUID	Created	CreatedDate
Gen_Notes	-	Diagrams	-
IsControlled	IsControlled	Element	-
LastLoadDate	LastLoadDate	Elements	-
LastSaveDate	LastSaveDate	Flags	PackageFlags
LogXML	LogXML	IsControlled	IsControlled
ModifiedDate	Modified	IsModel	-
Name	Name	IsNamepace	Namespace
Namespace	IsNamepace	IsProtected	Protected
Notes	Notes	IsVersionControlled	-
Package_ID	PackageID	LastLoadDate	LastLoadDate
PackageFlags	Flags	LastSaveDate	LastSaveDate
ParentID	ParentID	LogXML	LogXML
PkgOwner	Owner	Modified	ModifiedDate
Protected	IsProtected	Name	Name
TPos	TreePos	Notes	Notes
UMLVersion	UMLVersion	ObjectType	fixed 5
UseDTD	UseDTD	Owner	PkgOwner
Version	Version	PackageGUID	ea_guid
XMLPath	XMLPath	PackageID	Package_ID
		Packages	-

[1] To find out more about the API have a look in my book Scripting EA.

t_package	EaPackage		EaPackage	t_package
			ParentID	ParentID
			TreePos	TPos
			UMLVersion	UMLVersion
			UseDTD	UseDTD
			Version	Version
			XMLPath	XMLPath

... omitted ...

11. Bits and Pieces

This section contains a couple of details for selected columns from where they are referenced. A back reference is included at the end of each chapter. Note that most of the following descriptions do not detail the contents but give a sample of the contents only. For those having dug that far it will be obvious how to decode the contents.

11.1 CSV Lists

A number of columns contain semi-colon separated lists in the format `<key>=<value>;` where these pairs can appear more than once thus forming a list of key-value pairs. Usually `<key>` is alphanumeric including '_' (underscore). Value itself can contain any chars except '=' and ';'.

I found that EA often does not check this constraint and if people enter e.g. a semi-colon in a name it will simply confuse EA in it's later behavior but will not croak[1] that an illegal char is used. Well, it's EA.

... omitted ...

[1]Just try this with a stereotype. Enter *abc;def* as stereotype. Save, close and re-open the element. Now it shows just *abc*. However, using the ellipsis will show *abc;def* as possible (but unchecked) stereotype. I already reported that as bug years ago. It's still not fixed in V12...

12. User Settings

There are a couple of locations where EA stores information locally per user. These can be found in

- Registry
- %APPDATA%Sparx SystemsEA
- %PROGRAMFILES%Sparx SystemsEA

These locations are mainly used to hold user options (those from `Tools/Options`), Layout information, MDG data and more.

12.1 Registry

In former EA versions the registry keys were just a handfull, but now there are tons. The EA relevant keys are found at

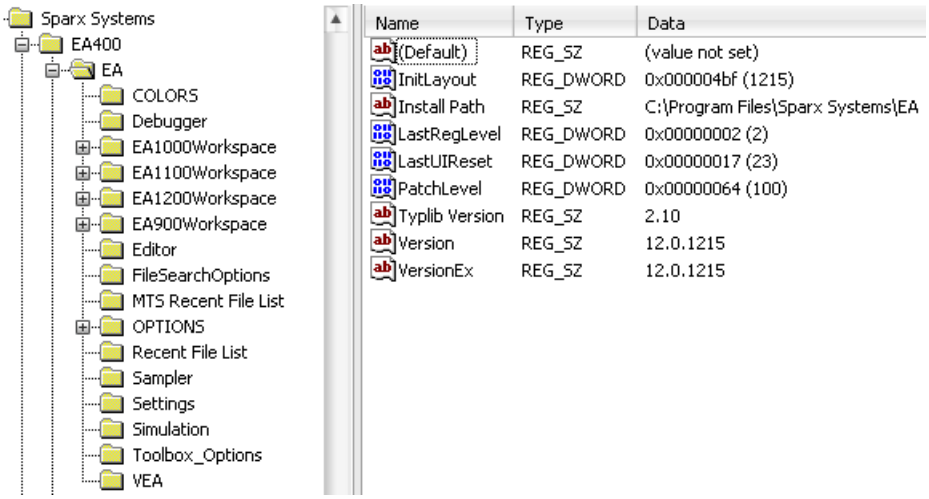
- `HKEY_CURRENT_USER\Software\Sparx Systems\EA400`
- `HKEY_CURRENT_USER\Software\Sparx Systems\EAAddins`
- `HKEY_LOCAL_MACHINE\SOFTWARE\Sparx Systems\EAAddins`

Obviously the `EAAddins` keys are used for addins and the usualy suspects will know how to deal with them.

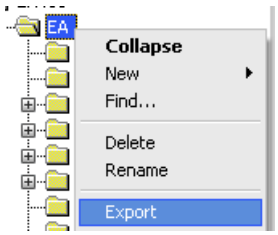
The first key contains all the marvelous settings which EA needs to show up. Quite some of the `Tools/Options` are hidden here besides all the layout stuff. So when you need to find certain settings and don't find any of the files being changed you should export the EA registry part, change the settings and export the registry again. Now you can compare the exports (e.g. with `WinMerge`).

If you need to change any of `Tools/Options` you likely have to go through the registry. And in order to take the changes effect you have to restart EA. I can't go into details for each single option but instead give you a simple receipt how to find the right one in case you need it.

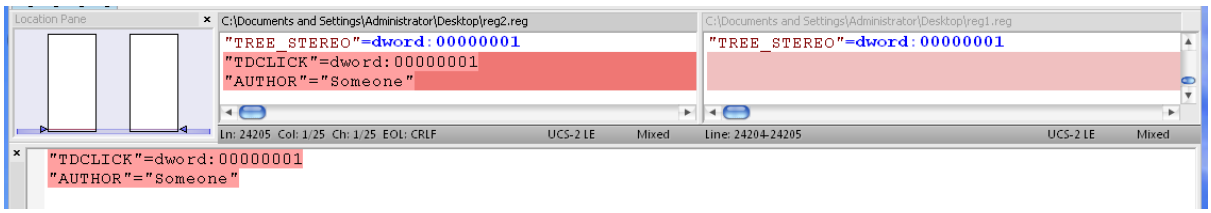
As an example we want to change the `General/General/Author` and `General/General/Double Click`. Per default they are set to *Administrator* and *Shows Properties* on my machine. So the first thing to do is to locate them. You can simply open the registry by entering `regedit` in the command line of the Windows `Start` menu. Next navigate to `HKEY_CURRENT_USER\Software\Sparx Systems\EA400\EA` which should look like this:



Now you need to export the current contents of the `EA` branch:



In order to spot the difference the options must be changed. Setting the author to *Someone* and checking *Opens Branch* will modify the registry. The `EA` branch must be saved once more to another file. When comparing the two exports with WinMerge[1] you can easily spot the differences.



Obviously the key `Author` holds the `Author` option as plain string and `TDCLICK` is an enumeration[2]. Since both values were added as new they will have defaults when not in the registry. So to get the default back you would need to remove the keys. In case of the `TDCLICK` you could also set it to *0*.

Using the above schema you should be able to figure out any option/key pair.

12.2 APPDATA

... omitted ...

[1] http://winmerge.org

[2] You would find that out when comparing after setting `Double Click` to *Open Branch & Diagram* which would result in a value of *2* for `TDCLICK`.

13. GUI References

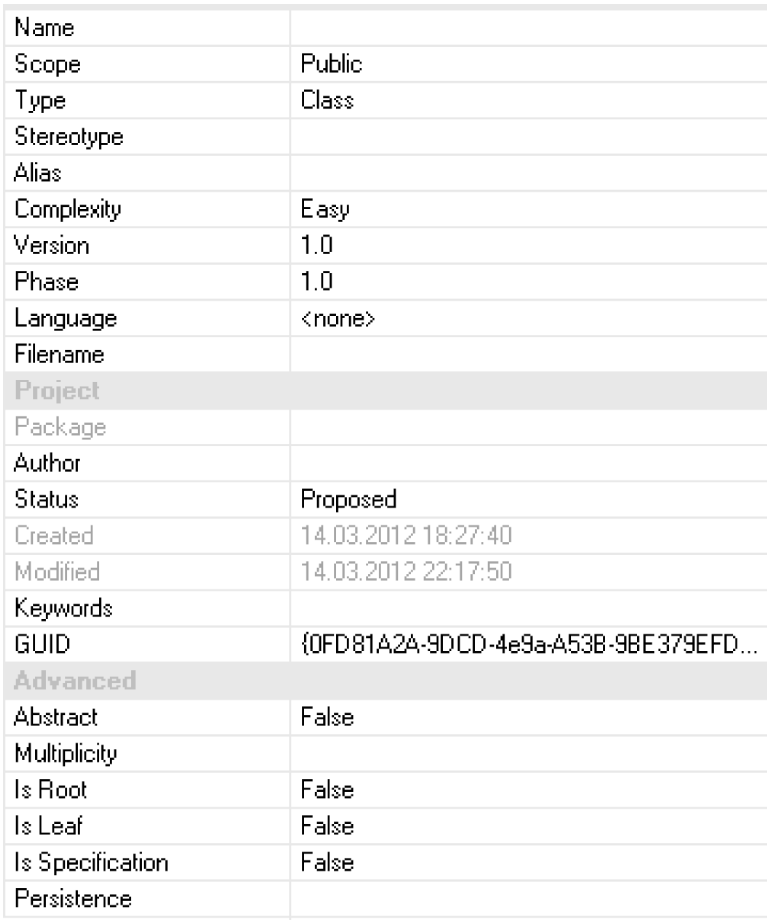
This section contains snapshots of various property windows. Many properties are highlighted with a red rectangle. For those a reference into the according table if given below the snapshot. In most cases the name of the property and the column name are identical. Some have just an additional blank for user readability. However, some properties differ from the column name. In that case the according user readable label is specified in parentheses right after the column name.

Recent versions of EA received a redesign of quite some property windows. I have updated the screenshots in this section to reflect the layout in EA V12. However, there are a few windows that have not changed substantially and are still shown as those from former EA versions. That is where e.g. only the window tabs were moved from top to left.

13.1 Element

Element related property windows also apply to packages due to the dualism of packages being also elements.

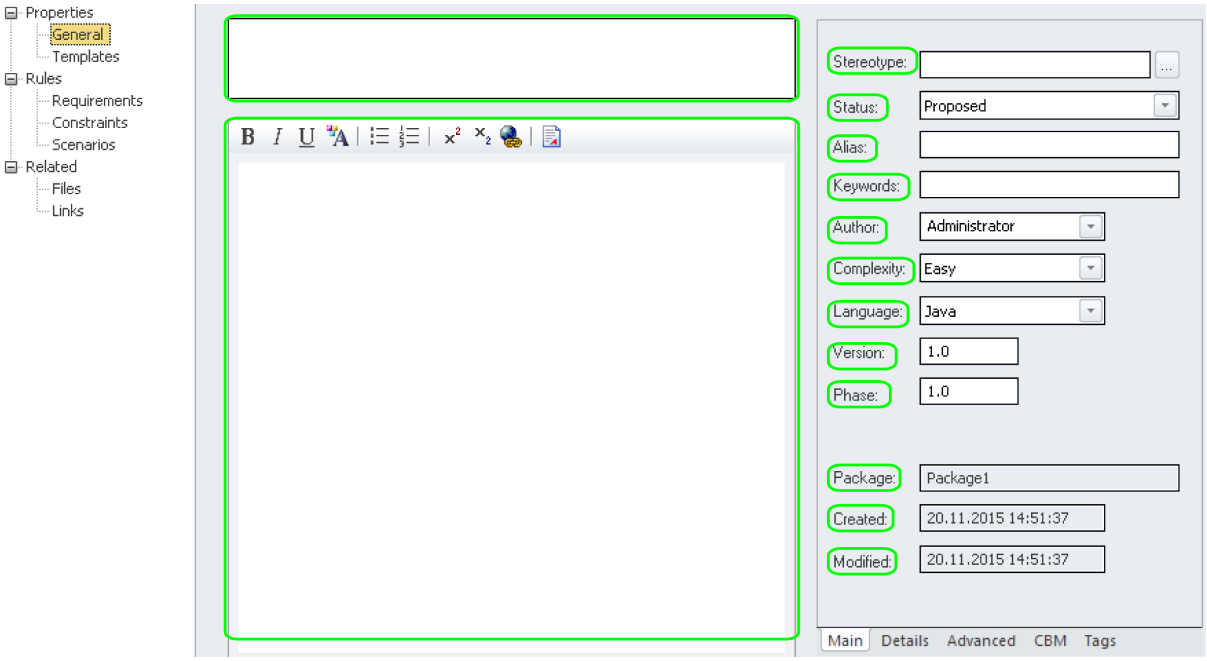
13.1.1 Dockable Properties Window



References:

Label	Column	Label	Column
`Name`	t_object.Name	`Status`	t_object.Status
	t_package.Name	`Created`	t_object.CreatedDate
`Scope`	t_object.Scope		t_package.CreatedDate
`Type`	t_object.Object_Type	`Modified`	t_object.ModifiedDate
`Stereotype`	t_object.Stereotype		t_package.ModifiedDate
`Alias`	t_object.Alias	`Keywords`	t_object.PDATA5
`Complexity`	t_object.Complexity	`GUID`	t_object.ea_guid
`Version`	t_object.Version		t_package.ea_guid
`Phase`	t_object.Phase	`Abstract`	t_object.Abstract
`Language`	t_object.GenType	`Multiplicity`	t_object.Multiplicity
`Filename`	t_object.GenFile	`Is*`	t_object.Is*
`Author`	t_object.Author	`Persistence`	t_object.Persistence
`Package`	t_object.Name (Name of parent package)		

13.1.2 General/Main Properties Window



References:

Label	Column
`Name`	t_object.Name / t_package.Name
`Notes`	t_object.Notes / t_package.Notes
`Version`	t_object.Version / t_package.Version
`Alias`	t_object.Alias
`Author`	t_object.Author
`Version`	t_object.Version
`Stereotype`	t_object.Stereotype
`Keywords`	t_object.PDATA5
`Complexity`	t_object.Complexity
`Status`	t_object.Status
`GenType`	t_object.GenType
`Phase`	t_object.Phase
`Package`	t_object.Package_ID -> t_package.Name
`Created`	t_object.CreatedDate
`Modified`	t_object.ModifiedDate

Note that some properties are ambiguous as they appear in both t_object and t_package for package elements.

... omitted ...

14. Query Caveats

Once you start using direct SQL you need to know:

> **All SQL are equal**
>
> But some SQL are more equal than others

And that's unfortunately true. A SQL for MS Access (EAP) is different to that of its 'big brother' MS SQL Server.

… omitted …

15. SQL Search Builder

Once you worked out the previous chapters you are ready to provide your EA users with some fancy searches (available as download[1]). You find the search builder when navigating from `Ctrl-F/Builder` and pressing the left `New Search` icon.

... omitted ...

[1] https://liquit.biz/query.zip

16. Further Reading

Finally I would like to add a few links where you will get further help.

16.1 Feedback

Any questions you have are important. So you should ask them. Feedback is important for you, me and of course all the other readers. So you are encouraged to send these to one of the below links:

http://leanpub.com/InsideEA
: The page where you bought the book has a small discussion forum enabled. Here you can also post.

thomas.kilian@me.com
: You can mail me directly if you have specific questions. Or maybe you have information regarding the ?! markers.

16.2 Scripting Enterprise Architect

I have not touched scripting much in this book. Basically because it would simply break the scope of this book. However, you might be interested in EA's automation interface. My book **Scripting Enterprise Architect** which is available at http://leanpub.com/ScriptingEA will introduce you in that matter. Even if you are already firm with the API its references and a couple of not well known hints will make this a valuable guide for you.

16.3 Sparx Forum

Probably not necessary to name this source, but anyway:

> https://www.sparxsystems.com/forums/smf/index.php

When you post your questions here you should select the right forum and only post once. Getting help here is most likely the fastest lane you can find. I also post regularly.

16.4 Sparx Community

A not so well known source as Sparx itself does not link this on its forum pages:

> http://community.sparxsystems.com

Here you will find a variety of articles and resources around EA.

16.5 SQL in General

Of course: Google is your friend. But sometimes it's nice to have a direct link:

> http://www.1keydata.com/sql/sql-commands.html

This is nice place to get a quick reference to most of SQL. This info is available in different languages if you enter via the main site.

> http://www.w3schools.com/sql/sql_syntax.asp

is a bit more responsive and condensed.

I once downloaded a compact HTML page with all information in it, but that site has ceased. So to get something similar see my remark above…

16.6 Geert Bellekens

Geert has become a kind of institution at Sparx' forum. He is a regular poster. But beyond that he also has an excellent blog dealing with UML in general and Enterprise Architect in particular. Check it out:

> https://bellekens.com

You'll also find entries dedicated to SQL usage in EA

> https://bellekens.com/#SQL

with some more advanced SQL stuff.