SIEMENS

The Basics of Writing Custom Checkers (For KF Novices)

Taylor Anderson
NX Product Manager
(DesignLogic, Knowledge Fusion and Validation)
Wednesday, June 4th, 2008
UGS Connection Conference

NX

© 2007. Siemens Product Lifecycle Management Software Inc. All rights reserved
Siemens PLM Software

CAUTION

SIEMENS

Check-Mate checker development as a road trip…

- **Using the KF "ICE" for Check-Mate** (Which car should we take?)
- **The %ug_base_checker class in depth** (How should we do this?)
- **The %ug_base_checker_profile class** (Shall we caravan together?)
- **Finding nasties to report in models** (Where are we headed?)
  - **The %do_check: procedural block** (How will we get there?)
  - **Recording the results to the log** (What did we actually see?)
  - **Looping constructs** (Did we pack everything we need?)
  - **Filtering functions** (Have we brought too much stuff?)

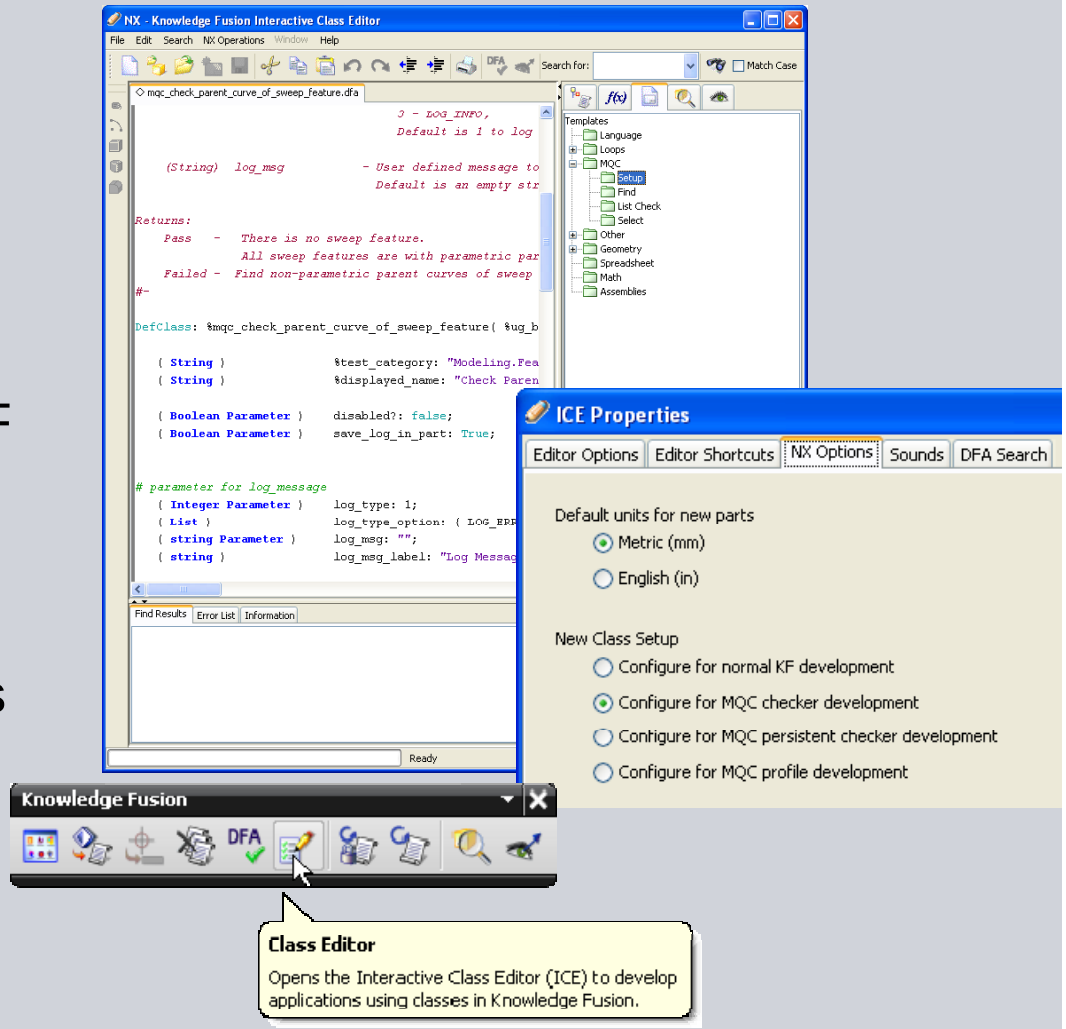
© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

Interactive Class Editor (ICE) for Knowledge Fusion

First Introduced in NX 5

SIEMENS

- Integrated Development Environment specifically for Knowledge Fusion
- Runs with full access to your live NX session
- Live, dynamic access to all KF content without manually updating external function libraries.
- Has class and code templates specifically for Check-Mate checkers and profiles.

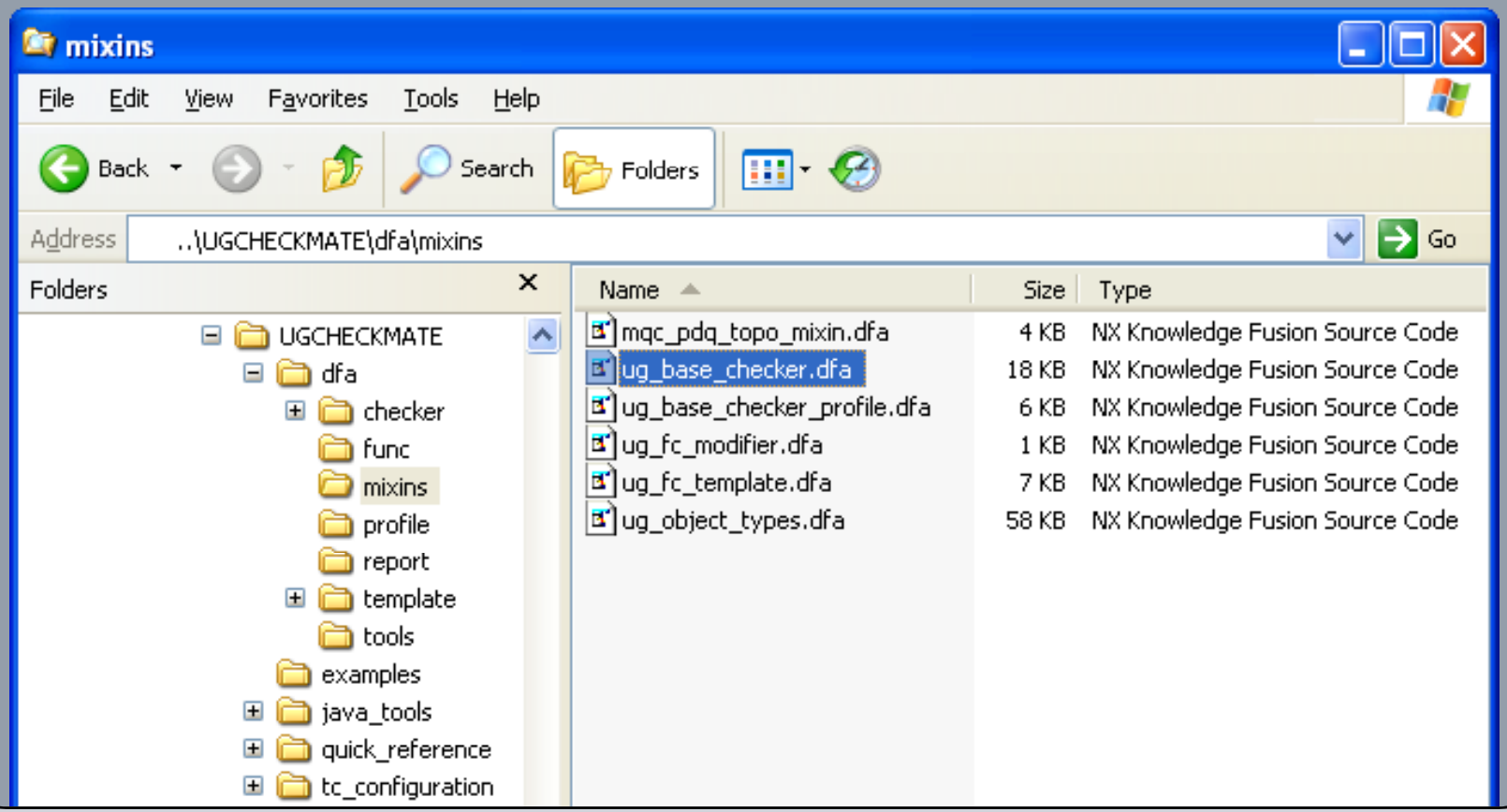


© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

“What do all of those attributes in the `%ug_base_checker` class actually do?”

SIEMENS

First step – Finding the `%ug_base_checker` mixin definition:



© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

The `%ug_base_checker` class in depth

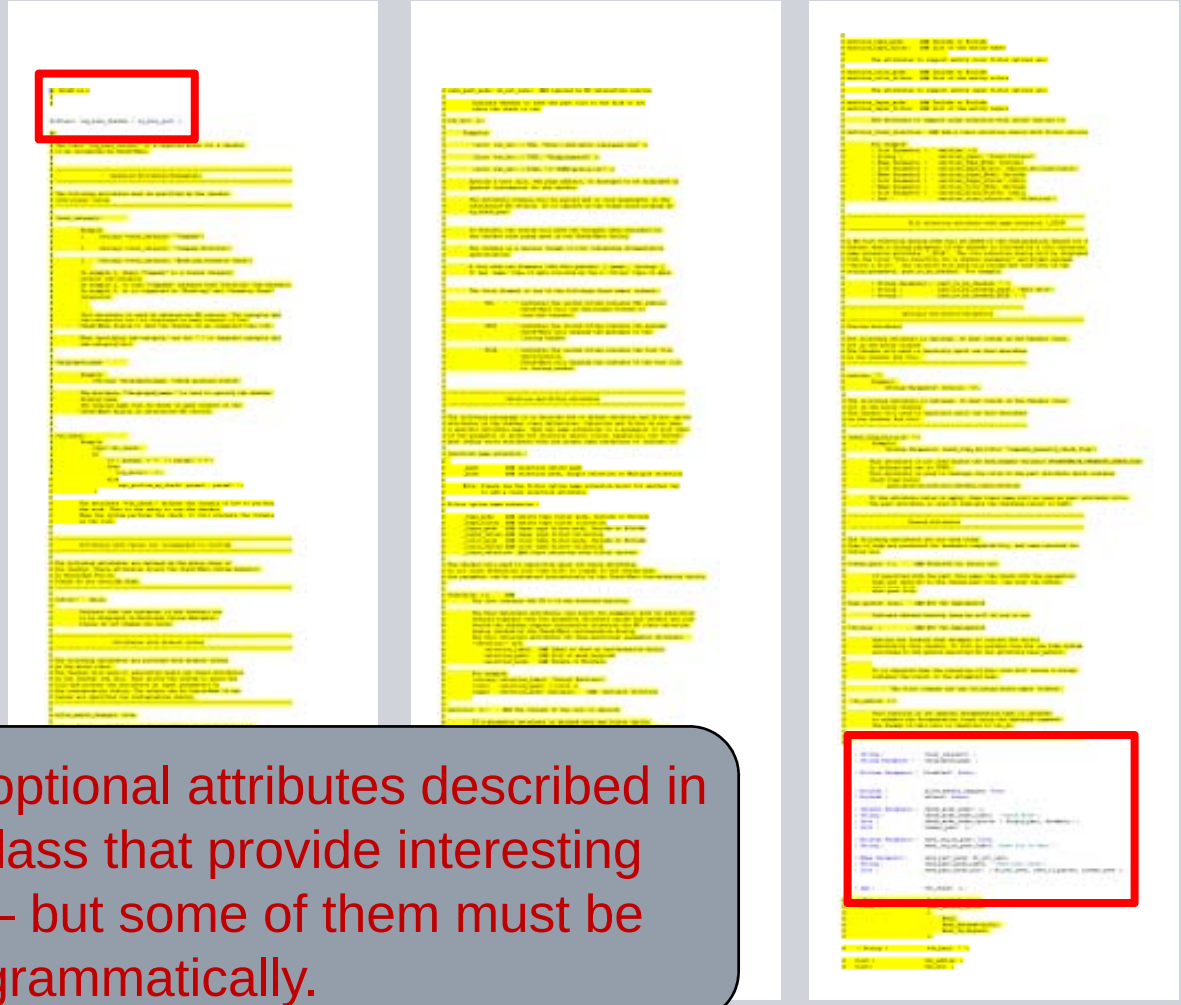
The Whole Enchilada

SIEMENS

The source code for the class is actually a tremendous source of information about the available options.

Yellow = Comments

Actual Code



There are optional attributes described in the mixin class that provide interesting behaviors – but some of them must be added programmatically.

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

The %ug_base_checker class in depth

The Default Set of Attributes

SIEMENS

NX - Knowledge Fusion Interactive Class Editor

File Edit Search NX Operations Window Help

Search for: Match Case

ug_base_checker.dfa

```
( String )              %test_category: ;
( String Parameter )    %displayed_name: ;

( Boolean Parameter )   Disabled?: false;


( Boolean )             allow_author_changes: true;
( Boolean )             inTree?: false;

( Integer Parameter )   check_mode_index: 1;
( String )              check_mode_index_label:  "Check Mode";
( List )                check_mode_index_option: { Single_part, Assembly };
( List )                %owner_part: {};

( Boolean Parameter )   save_log_in_part: true;
( String )              save_log_in_part_label: "Save Log in Part";

( Name Parameter )      save_part_mode: do_not_save;
( String )              save_part_mode_label:  "Save part mode";
( List )                save_part_mode_init: { do_not_save,
                                               save_if_passed,
                                               always_save };


( Any )                 %do_check: {};
```

Classes
- System
  - UG
  - NX
  - MQC
- User

The %ug_base_checker class in depth

The Really Useful Attributes

```
%test_category:  "Modeling.Features";

%displayed_name: "Early Chamfer or Edge Blend";

save_log_in_part: false;

save_part_mode: do_not_save;

%do_doc: { URL, "http://www.myurl.com/page1.htm" };

allow_author_changes: true;

version: “2";

check_flag_PA_title: "company_geometry_check_flag";
```

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

%test_category: and %displayed_name:

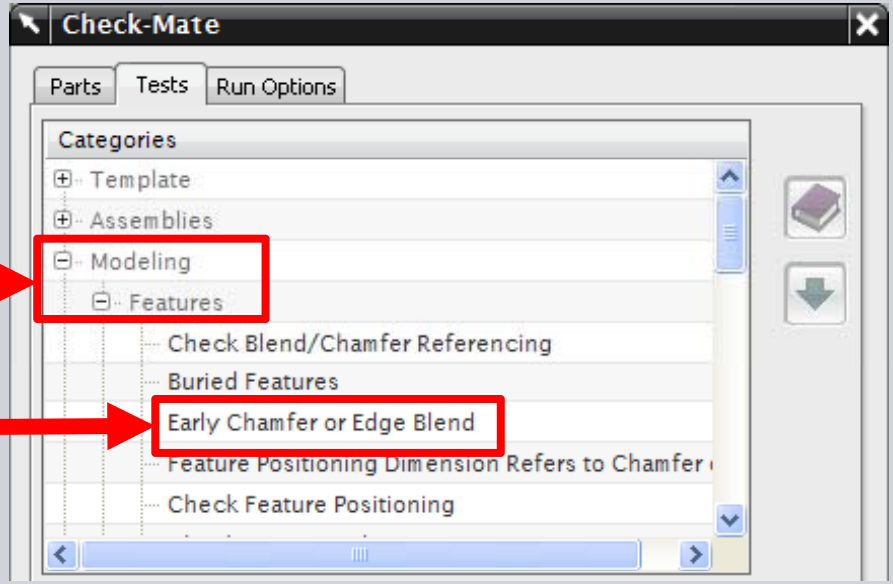
Attribute Details

SIEMENS

```
%test_category:  "Modeling.Features";
```

CAUTION

Typos here will create new branches in the tree.



CAUTION

Typos here will just look silly. ☺

```
%displayed_name: "Early Chamfer or Edge Blend";
```

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

`check_mode_index:`

Attribute Details

SIEMENS

1 and 2 options correspond to the following list:

`check_mode_index_option: { Single_part, Assembly };`

This one is ignored in interactive NX (as per the docs.)

EXPL

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

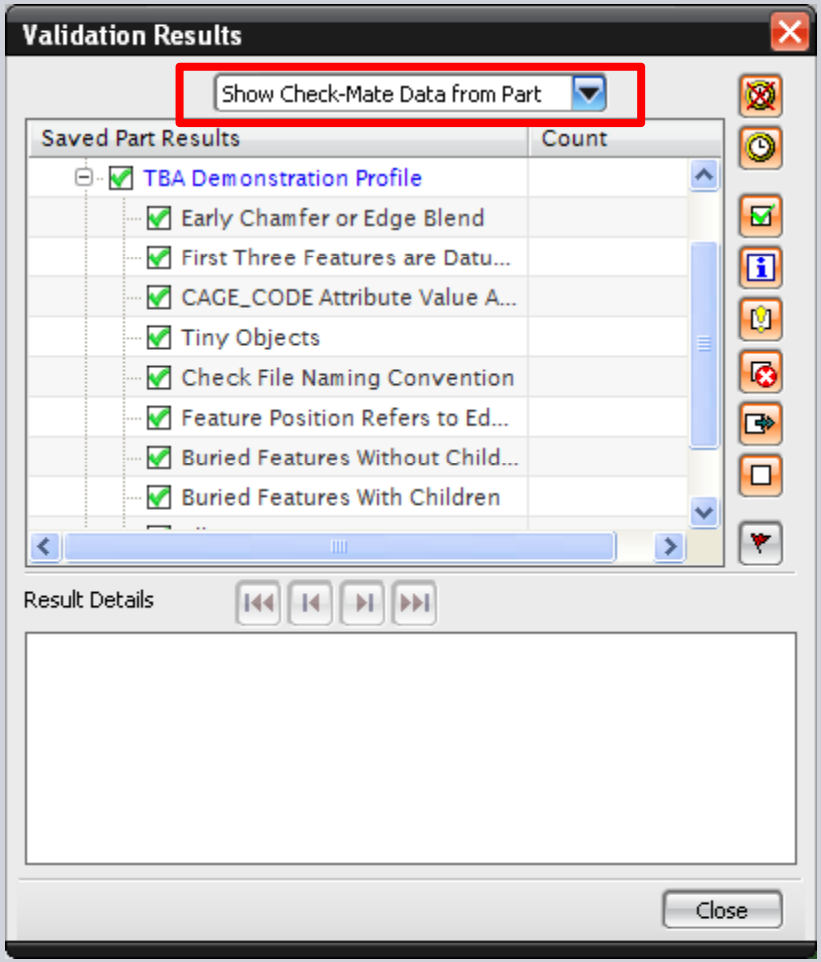
`save_log_in_part:`

Attribute Details

SIEMENS

When set to "**`true`**", the **`save_log_in_part:`** attribute will cause latest set of Check-Mate results to be stored persistently in the part file. These can then be examined later without re-running Check-Mate.

One set of results (the most recent) are stored in the part at a time. (It's not cumulative.)



© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

save_part_mode:

Attribute Details

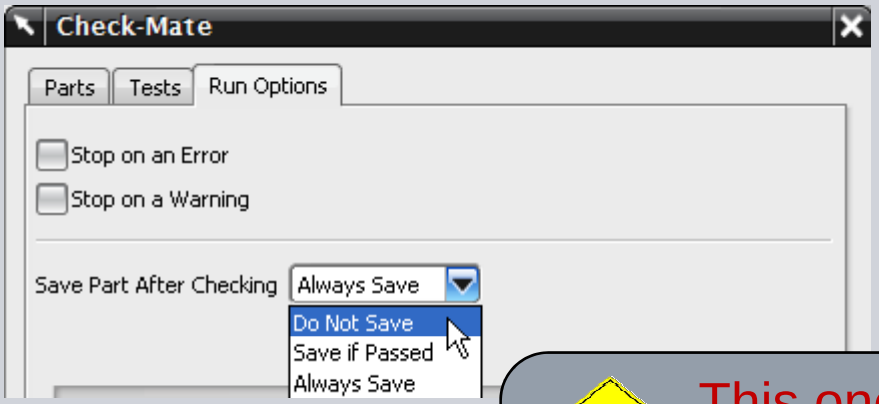
SIEMENS

The **save_part_mode:** attribute can be set to one of three modes:

1 = do_not_save

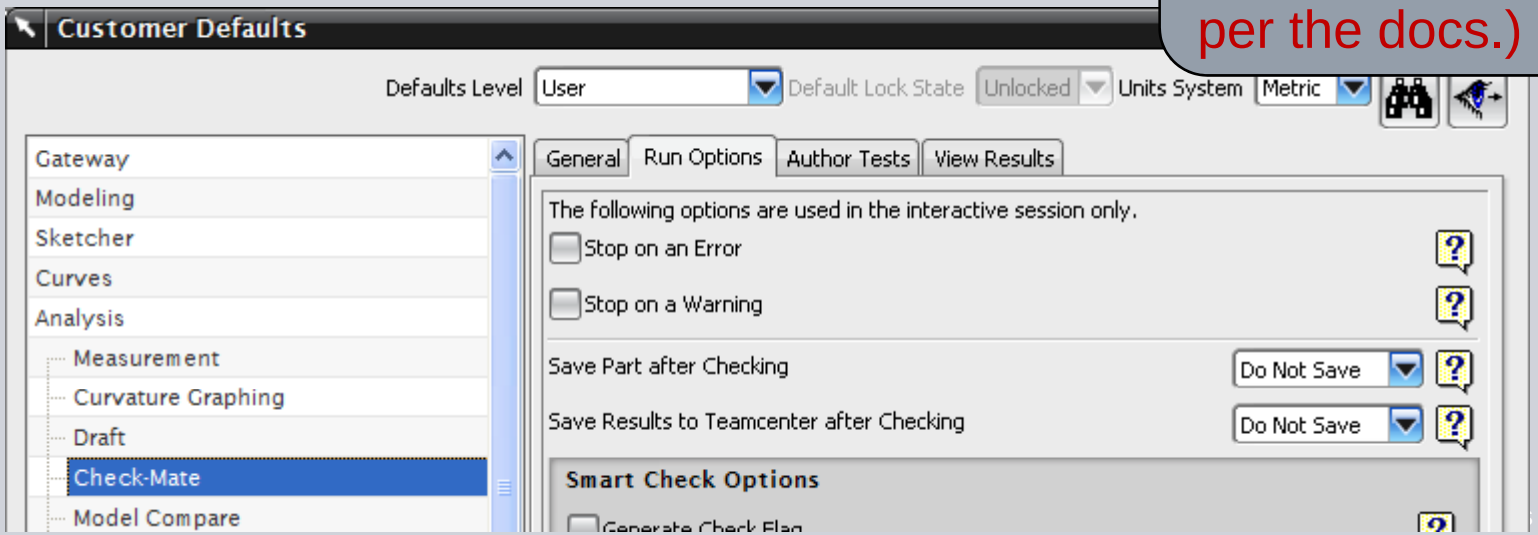
2 = save_if_passed

3 = always_save



CAUTION

This one is ignored in interactive NX (as per the docs.)



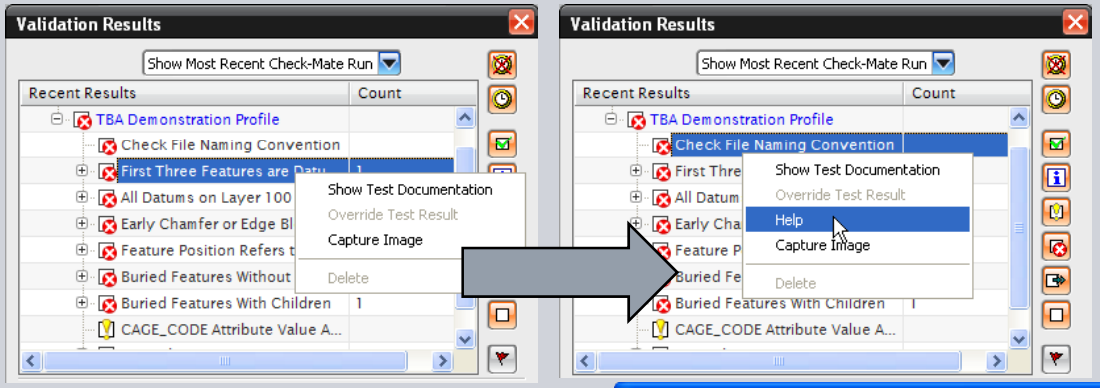
reserved

%do_doc:

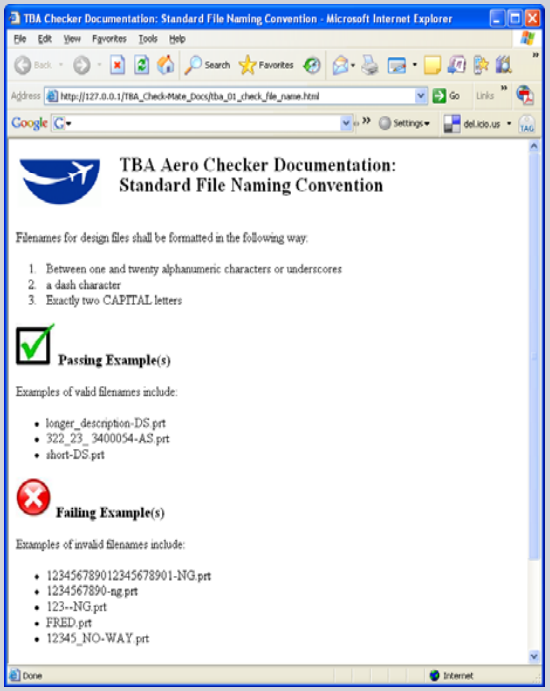
Attribute Details

SIEMENS

Adds a new MB3 action to the check in the Validation Results dialog.




```
(List) %do_doc: {TEXT,
                 "A text string"};

(List) %do_doc: {FILE,
                 "c:\TEMP\myfile.txt"};

(List) %do_doc: {URL,
                 "http://myurl.com/help.html"};
```



© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

`version:` and `check_flag_PA_title:`

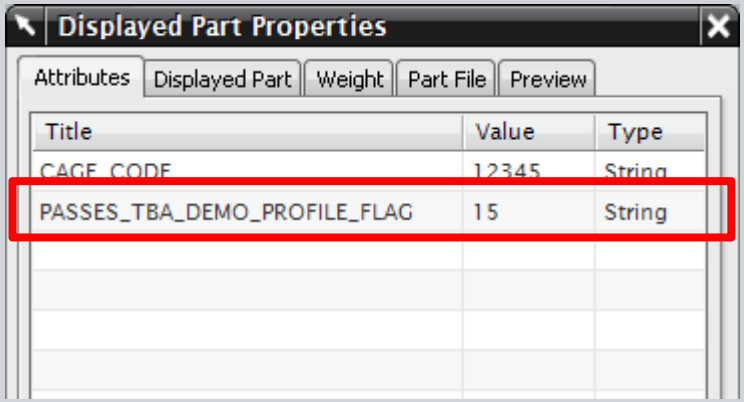
Attribute Details

SIEMENS

If used, **`version:`** will show up in two interesting places:

- Rule Set files for Teamcenter Workflow
- the “check flag” (part attribute) named in **`check_flag_PA_title:`**

(Both are optional.)

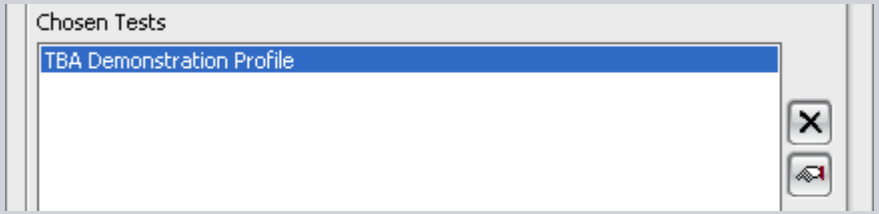


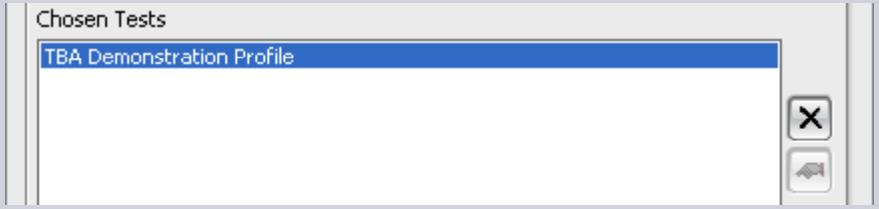
© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

`allow_author_changes:`

Attribute Details

Setting this one to FALSE disables the “customization” button for the check or profile, preventing end users from changing input values like tolerances, settings, etc. that should be kept standard.





© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

Making attributes user-modifiable (`Parameter`)

`Parameter` attributes are automatically presented to the user in the Check-Mate dialogs as follows:

Boolean type creates a checkbox

```
( Boolean Parameter )    Disabled?: FALSE;
( Boolean Parameter )    save_log_in_part: TRUE;
```

Number type creates a real input window

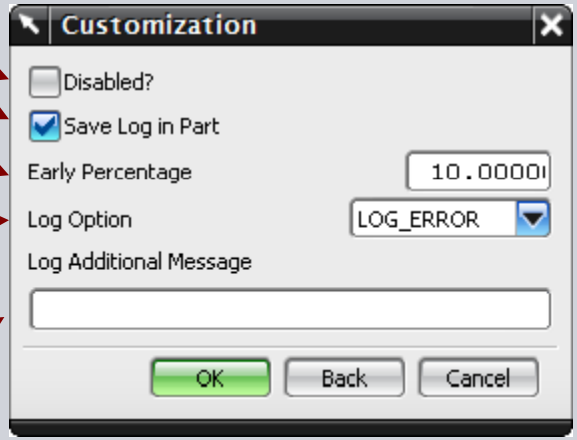
```
( Number Parameter )   early_percentage: 10;
( String )   early_percentage_label: %ug_MSGTXT("Early Percentage");
```

Integer (with extras) creates Option Menu

```
( Integer Parameter )   log_type: 1;
( List )   log_type_option: { LOG_ERROR, LOG_WARNING, LOG_INFO };
( String )   log_type_label: "Log Option";
```

String type creates a text entry

```
( String Parameter )   log_msg: "";
( String )   log_msg_label: %ug_MSGTXT( "Log Additional Message" );
```



© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

the `do_check:` attribute

All checks must have a `do_check:` attribute.

The `do_check:` attribute…

- Will be evalated every time the check is run.
- Uses a call to `ug_mqc_log()` to report results.
- Always **`uncached`** (so that old results are not stored.)

The rest of the checker class is all about organization, classification and behavior… This is where the rubber hits the road and the actual *checking* takes place.

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

`ug_mqc_log()` function (first argument)

```
-----------------------------------------------------------------------------
ug_mqc_log
Description: Generic MQC log function to log either error, warning or info.
Input:  (Name)     log_type        log type,can be type as follows:
                                        LOG_INFO
                                        LOG_WARNING
                                        LOG_ERROR

        (List)     object_tags     list of UG tags of objects
        (String)   message         detailed message
Return: (Integer) Status of log (0 - no error)
-----------------------------------------------------------------------------
```

(Pass)

You need at least one of these in your `do_check:` attribute .

First argument determines error severity

Second argument creates detailed lists for results

Third argument can be VERY descriptive, if done well. ☺

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

Tags vs. HostPointers

Check-Mate uses Tags instead of HostPointers

Tags are integers

Use mqc_XXX functions to cycle and filter objects using tags

Most importantly, `ug_mqc_log()` uses Tags.

```
%mqc_askHostpointerOfTag($tag)
%mqc_askTagOfHostpointer($hostpointer)
```

…can be used to move back and forth.
(…which can be handy sometimes.)

NOTE: Tags are not persistent across sessions, so they are not really very useful for KF apps where lists of entities are stored in saved part files. They are great for checking, however, where their use is transient.

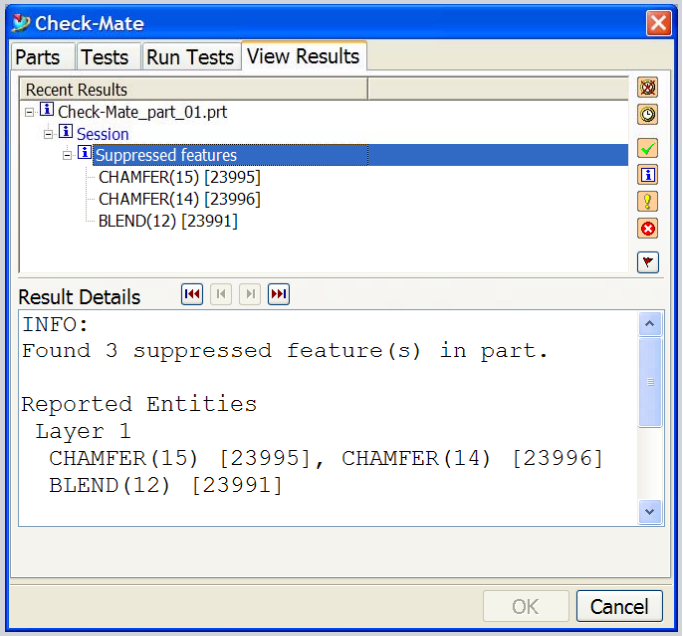
© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

ug_mqc_log() function (second argument)

List of tags:
{23991, 23996, 23995 }

(Not necessarily in timestamp order…)




```
( Any Uncached )   do_check:
      @{
          $suppressed_feats << mqc_askSuppressedFeatures();

          If  ( -Empty?( $suppressed_feats ) )
          Then
          @{
               $detail_msg << "Found " +
                  Stringvalue(Length($suppressed_feats ) ) +
                  " suppressed feature(s) in part.";
               ug_mqc_log( LOG_ERROR, $suppressed_feats, $detail_msg );
          }
          Else
              donothing;
      };
```

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

`ug_mqc_log()` function (third argument)

(concatenation)

```
$detail_msg << "Found " +
    Stringvalue(Length($non_linked_bodies)) +
    " non-linked bodies~nin this assembly part.";
```

(number to string)

(newline character)

(This also works…)

(concatenation)

```
$detail_msg << "Found " +
    Format( "%d", Length( $ non_linked_bodies) ) +
    " non-linked bodies~nin this assembly part.";
```

(number to string)

(newline character)

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

ug_mqc_log() function (third argument)

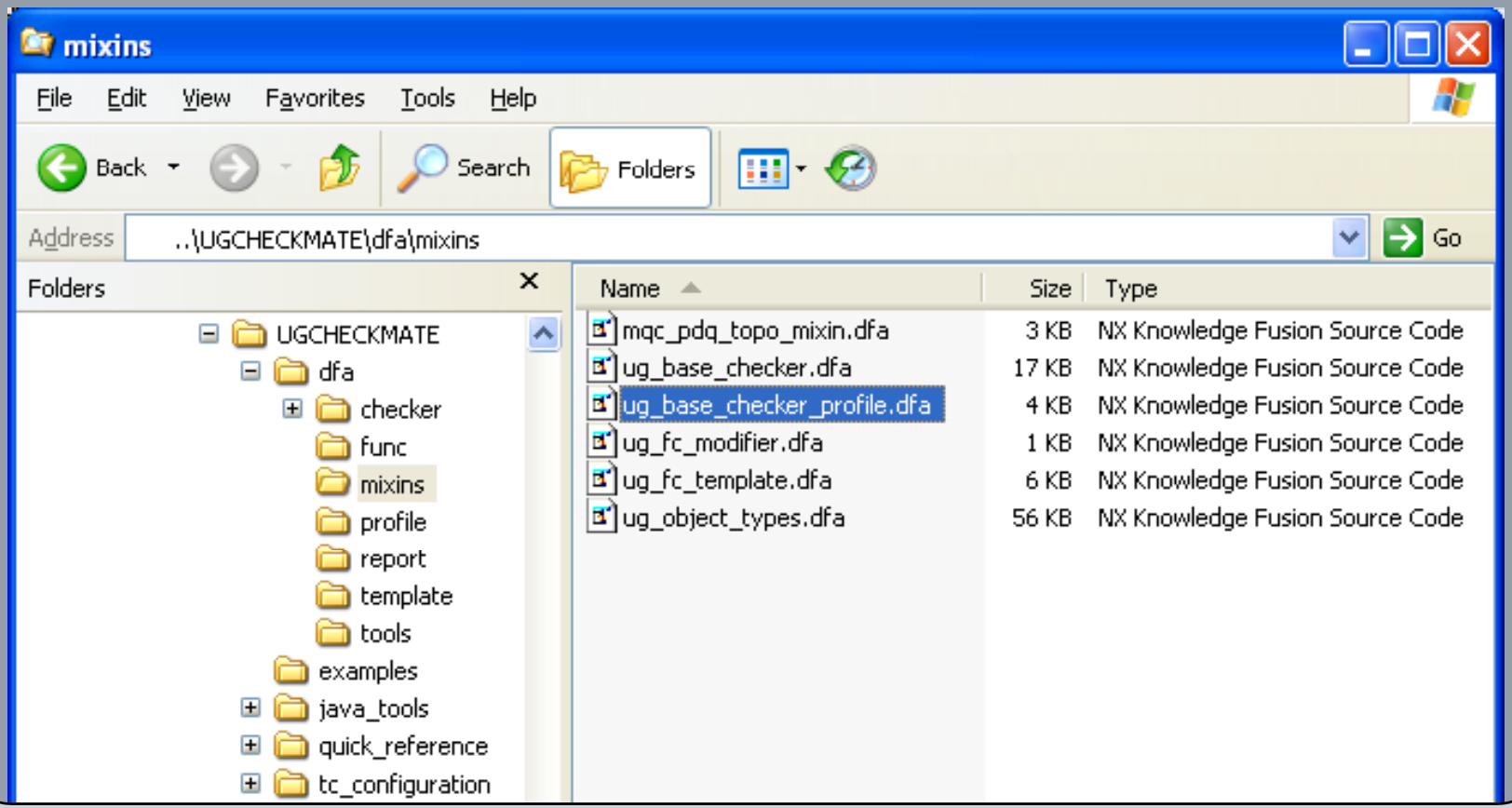
```
...
(string)  name_length_failure_str:
                if ( isLengthOK?: ) then ("") else ("The filename: " + filename: + " is too long. (max 30 characters)~n");
(string)  name_standard_failure_str:
                if ( isNameStandardOK?: ) then ("") else ("The filename: " + filename: + " does not meet filename standards.~n");
...
 $check << if (!isEverythingOK?:)
          then @{
            $looks_like_non_dwg_name_with_drawing << if (isDrawing?: & (passed_name_standard:="ppt or asm"))
                then ("This part meets the piece part or assembly~nname spec, but also contains a drawing.~n") else ("");
            $looks_like_drawing_name_with_no_drawing << if (!isDrawing?: & (passed_name_standard:="drawing"))
                then ("This part meets the drawing name spec, but~ndoes not contain a drawing.~n") else ("");
            $looks_like_drawing_name_with_no_components << if (isPiecePart?: & (passed_name_standard:="drawing"))
                then ("This part meets the drawing name spec, but~nhas no components.~n") else ("");

            $detail_msg << $looks_like_non_dwg_name_with_drawing +
                        $looks_like_drawing_name_with_no_drawing +
                        $looks_like_drawing_name_with_no_components +
                        name_length_failure_str: +
                        name_standard_failure_str:;

             ug_mqc_log( LOG_ERROR, {}, $usr_msg + $detail_msg + $debug_msg );
          }
```

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

“What do all of those attributes in the `%ug_base_checker_profile` class actually do?”

SIEMENS

First step – Finding the `%ug_base_checker` mixin definition:



© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

The %ug_base_checker_profile class in depth

The Default Set of Attributes

SIEMENS

NX - Knowledge Fusion Interactive Class Editor

File Edit Search NX Operations Window Help

Search for: Match Case

ug_base_checker_profile.dfa

```
(String)    %test_category: ;
(String)    %displayed_name: ;

(Boolean)   inTree?: false;
(Boolean)   Disabled?: false;

# check_mode_index: 1 = Single Part, 2 = Assembly
(Integer)   check_mode_index: 1;

# check_time_index: 1 = Manual, 2 = Update, 3 = Part Save
(Integer)   check_time_index: 1;

(List)      %owner_part: {} ;

(Boolean)   save_log_in_part: true;
(String)    save_log_in_part_label: "Save Log in Part";

(Name)      save_part_mode: do_not_save;
(List)      save_part_mode_init:
{
    do_not_save,
    save_if_passed,
    always_save
};

(Boolean)   allow_author_changes: true;
```

Classes

- System
  - UG
  - NX
  - MQC
- User

The %ug_base_checker_profile class in depth

The Really Useful Attributes (Do these look familiar?)

```
%test_category:  "Modeling.Features";

%displayed_name: "Early Chamfer or Edge Blend";

save_log_in_part: false;

save_part_mode: do_not_save;

%do_doc: { URL, "http://www.myurl.com/page1.htm" };

allow_author_changes: true;

version: “2";

check_flag_PA_title: "company_geometry_check_flag";

check_time_index: 1;
```

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

`check_time_index:`

Attribute Details

SIEMENS

(This attribute is also optional, of course.)

Must be used in conjunction with a couple of environment variables:

`UGII_CHECKMATE_ALLOW_AUTO_RUN = 1`

where... 1 = during save

2 = during update

3 = during both

`UGII_CHECKMATE_DEFAULT_CHECKER = profile_class_name`

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

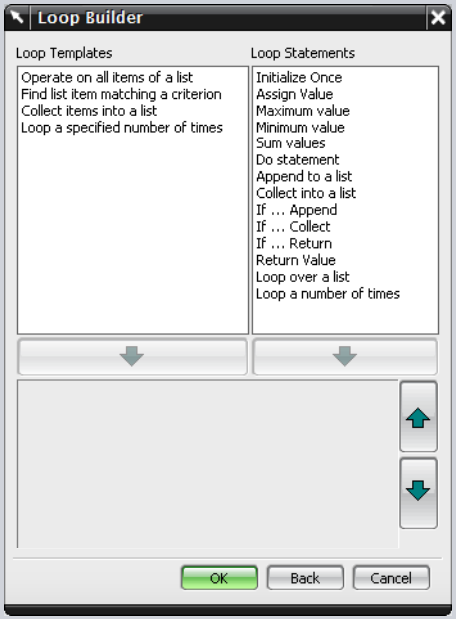
Looping Constructs

Operate on all items in a list

```
Loop {
For $a In my_list:;
...
}
```

Collect items into a list

```
Loop {
For $a In my_list:;
If ( $a > 5 ) Collect $a;
}
```



Loop a specified number of times

```
Loop {
For $a From 1 To 10 By 2;
...
}
```

Find list item matching a criterion

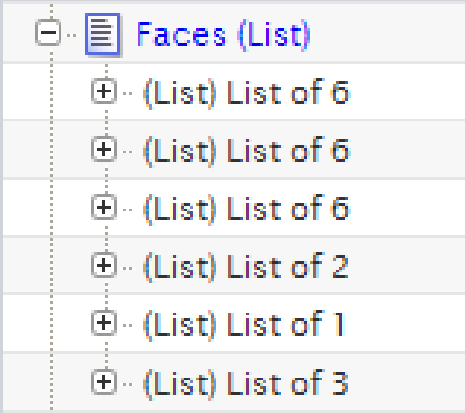
```
Loop {
For $a In my_list:;
If ( $a > 5 ) Return $a;
Return Is -1;
}
```

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

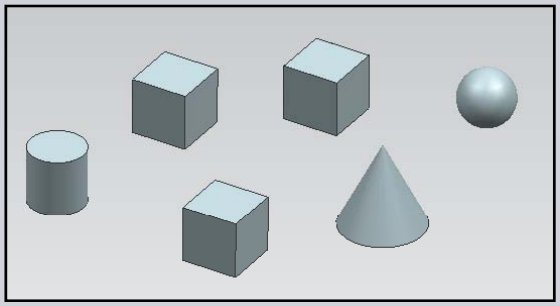
Collecting vs. Appending (in a loop)

Using "**collect**" will package each result into a list.




```
$bodies << mqc_askEntities(SOLID_BODY);

$faces << Loop {
  For $body in $bodies;
  collect ug_body_askFaces(%mqc_askHostpointerOfTag($body));
};
```

$faces is a list
containing six lists (one per body)
Containing between one and six faces each.



© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

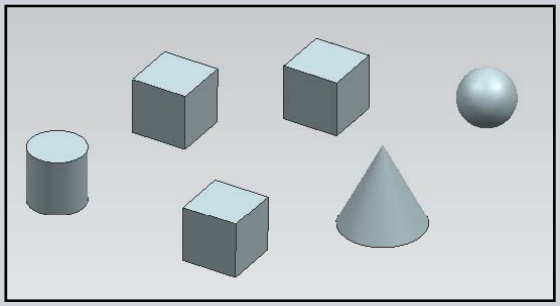
SIEMENS

Collecting vs. Appending (in a loop)

Using "**append**" will cumulatively add the result to the end of a flat list.

```
$bodies << mqc_askEntities(SOLID_BODY);

$faces << Loop {
  For $body in $bodies;
  append ug_body_askFaces(%mqc_askHostpointerOfTag($body));
};
```

`$faces` is a flat list containing twenty-four faces.



© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

Collecting and filtering a list of tags…

Objective: Check to see if there is more than one body on layer one.

Collecting

Filtering

```
$all_solid_bodies << mqc_askEntities( SOLID_BODY );

$layer_1_bodies << mqc_sort_entities_by_layer( $all_solid_bodies, 1, lib, "libufmqc",  );

$one_body_on_layer_one <<  if ( length( $layer_1_bodies ) <= 1 ) then (TRUE) else (FALSE);

$mutli_body_error_msg << if ($one_body_on_layer_one)
                            then "" else "Found multiple solid bodies on layer 1.~n";

$report << if ( !$one body on layer one  )
           then (ug_mqc_log( LOG_ERROR, $layer_1_bodies, $mutli_body_error_msg )
           else donothing;
```

Reporting

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

SIEMENS

Collecting and filtering some more tags…

Objective: Check that the first three features are datum planes.

```
$all_feats_tags << mqc_askFeatures();
$check_for_non_datums << Loop
{
  for $a from 1 to 3;
  for $plane_test1 is if ((second(mqc_askFeatureNameAndType(nth($a,$all_feats_tags)))) =
        ("ABSOLUTE_DATUM_PLANE")) then TRUE else FALSE;
  for $plane_test2 is if ((second(mqc_askFeatureNameAndType(nth($a,$all_feats_tags)))) =
        ("DATUM_PLANE")) then TRUE else FALSE;
  if ( !$plane_test1 & !$plane_test2 ) collect nth($a,$all_feats_tags);
};

if ( length($check_for_non_datums) > 0)
  then
  @{
    $detail_msg << "First three features are not all Datum Planes.";
    ug_mqc_log( Nth(log_type:, log_type_option:), $check_for_non_datums, $detail_msg);
  }
  else donothing;
```

Collecting

Filtering

Reporting

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved

QUESTIONS?

WARNING!
IF THE HELP DESK THINKS YOUR QUESTION
IS STUPID, WE WILL SET YOU ON FIRE

SIEMENS

Thank you!

www.siemens.com/plm

© 2008. Siemens Product Lifecycle Management Software Inc. All rights reserved