



ATOL PROBLEM RESOLUTION :
FORMAT MSG in Entity with comment (ATOLBAS)
And Commented+ (FOUNDATION+)
PLEX Version greater or equal than 5.0.
ATOL V5.0 level 286
ATOL V5.1 level 290
(27/03/03)



| | |
|---------------------------------------------------------|----------|
| I. PURPOSE : | 3 |
| II. RESOLUTION FOR ATOLBAS/ENTITY WITH COMMENT : | 3 |
| 1. STEP 1 : | 4 |
| 2. STEP 2 : | 4 |
| 3. STEP 3 : | 4 |
| 4. STEP 4 : | 5 |
| III. RESOLUTION FOR FOUNDAT+/COMMENTED+ : | 6 |
| 1. STEP 1 : | 7 |
| 2. STEP 2 : | 7 |
| 3. STEP 3 : | 8 |
| 4. STEP 4 : | 8 |
| 5. NOTES : | 9 |



I.PURPOSE :

This document describes how to manage the problem concerning the modification around the Plex format message function in ATOLBAS/Entity with comment (ATOLBAS) and Commented+(FOUNDATION+).

In fact the FORMAT MESSAGE function does not support a concatenation, of more than 2000 characters. To work around this, you should use a Source code API

So you should have problems in ATOLBAS/Entity With Comment.Edit function and FOUNDAT+/Commented+.Comment+.ClientFunctions.Edit+ functions. When you try to display a comment of more than 2000 characters it will generate a General Protection Fault (GPF).

To solve this problem, use the next steps, first for ATOLBAS/Entity With Comment and FOUNDAT+/Commented+.Comment+.ClientFunctions.Edit+ depending on the technology you use.

II.RESOLUTION FOR ATOLBAS/ENTITY WITH COMMENT :

The modification consists in adding a workaround which avoids the format message function in the ATOLBAS/Entity With Comment.Comment.Edit function.

```
... Set Details<_Edit multiline> = Details<_Edit multiline> CONCAT Base<_String>  
📁 Format Message Message: ATOLBAS/_Entity with comment.Comment.Edit.ConcatComment, Details<_Edit multiline>
```

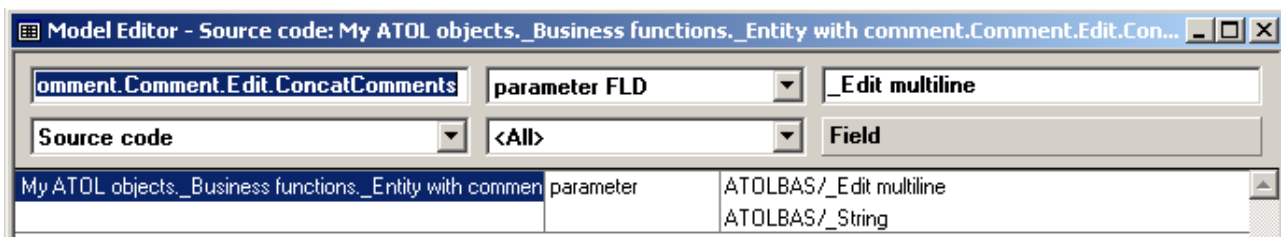
These modifications have to be done in your ATOLBAS customisation layout, then all your comment function will be changed. Consider that you Entity With comment entity is customised by <MyCustomizedCommentEntity>.

1. STEP 1 :

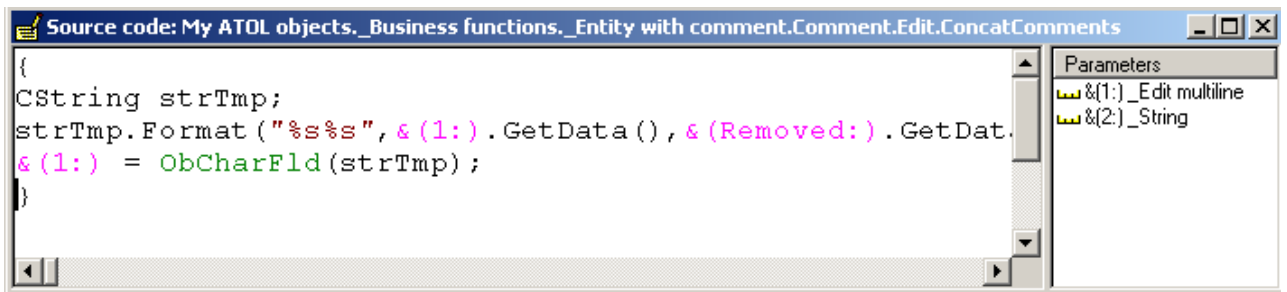
Add a API "ConcatComments" to the **<MyCustomizedCommentEntity>.Comment.Edit** function.

2. STEP 2 :

Add the following parameter to your API :



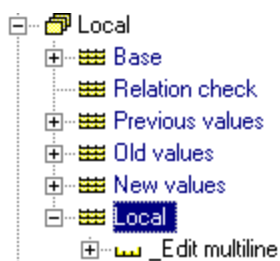
Fill it with the following statement :



3. STEP 3 :

Declare a new field ATOLBAS/_Edit multiline in variable **<Local>** in the **<MyCustomizedCommentEntity>.Comment.Edit** function.

field in that variable.





4. STEP 4 :

Add the following lines in the **<MyCustomizedCommentEntity>.Comment.Edit** function (subroutine set panel field values)

```
Pre Point 0 Set panel field values
Set Local<_Edit multiline> = <_Edit multiline.Nul>

Edit Point 1 Build edit multiline
API Call Source code: Company.Comment.Edit.Concat
Set Details<_Edit multiline> = <_Edit multiline.Nul>

Pre Point 1 Set panel field values
Set Details<_Edit multiline> = Local<_Edit multiline>
Set Old values<_Edit multiline> = Details<_Edit multiline>
```

Map parameter for ConcatComments API as followed :

| Type | To | From |
|------|-------------------|------------------------|
| I | <_Edit multiline> | Local<_Edit multiline> |
| I | <_String> | Base<_String> |

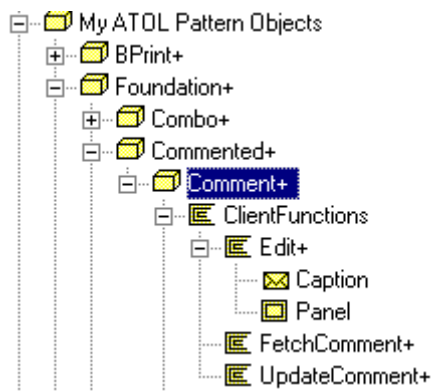
III.RESOLUTION FOR FOUNDAT+/COMMENTED+ :

The modification consists in adding a workaround which avoids the Format Message function in the FOUNDAT+/Commented+.Comment+.ClientFunctions.FetchComment+ function.

```

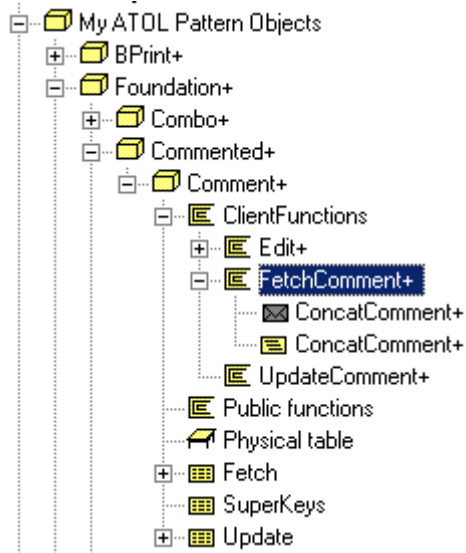
Edit Point  Process data fetched+
... Set  Output<CommentText+> = Output<CommentText+> CONCAT StringL+<String+>
Format Message  Message: UISTYLE+//FetchComment+.ConcatComment+, Output<CommentText+>
  
```

These modifications have to be done in your FOUNDAT+ customisation layout, then all your comment function will be changed. Consider that Commented+ entity is customised by **My ATOL Pattern Object.Foundation+.Commented+..**



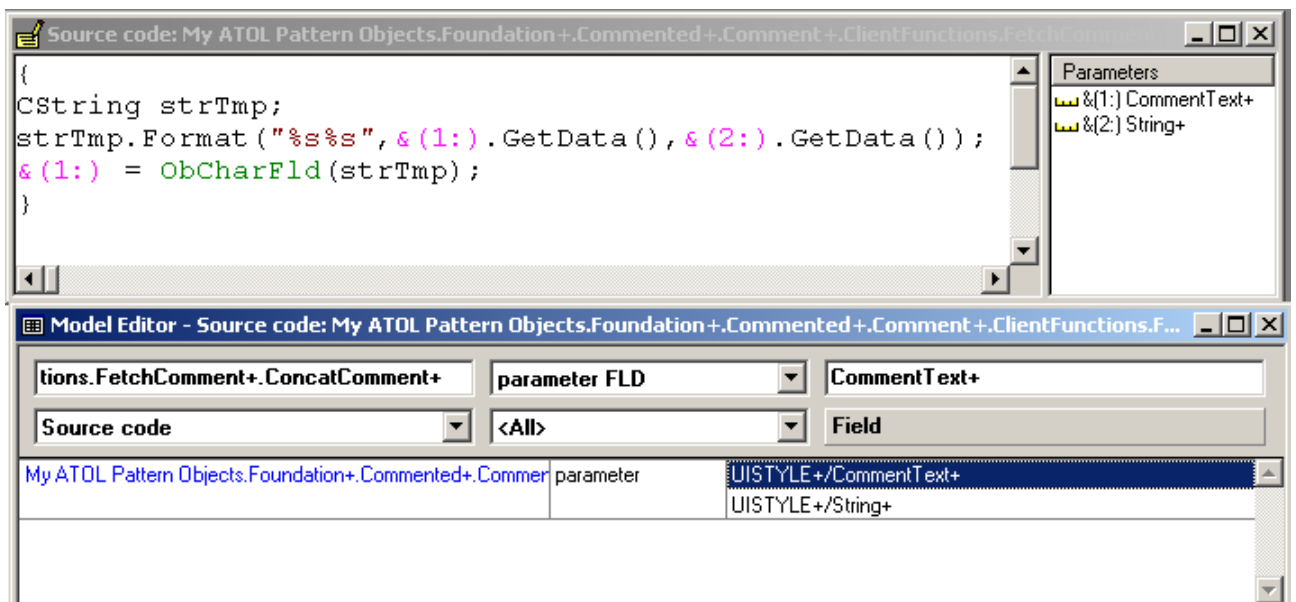
1. STEP 1 :

Add a API "ConcatComments" to the ...Comment+.ClientFunctions.FetchComment+ function.



2. STEP 2 :

Add the following parameter to your API, Fill it with the following statement : :



Source code: My ATOL Pattern Objects.Foundation+.Commented+.Comment+.ClientFunctions.FetchComment+

```

{
CString strTmp;
strTmp.Format ("%s%s", &(1:).GetData (), &(2:).GetData ());
&(1:) = ObCharFld (strTmp);
}
  
```

Parameters

- &(1:) CommentText+
- &(2:) String+

Model Editor - Source code: My ATOL Pattern Objects.Foundation+.Commented+.Comment+.ClientFunctions.F...

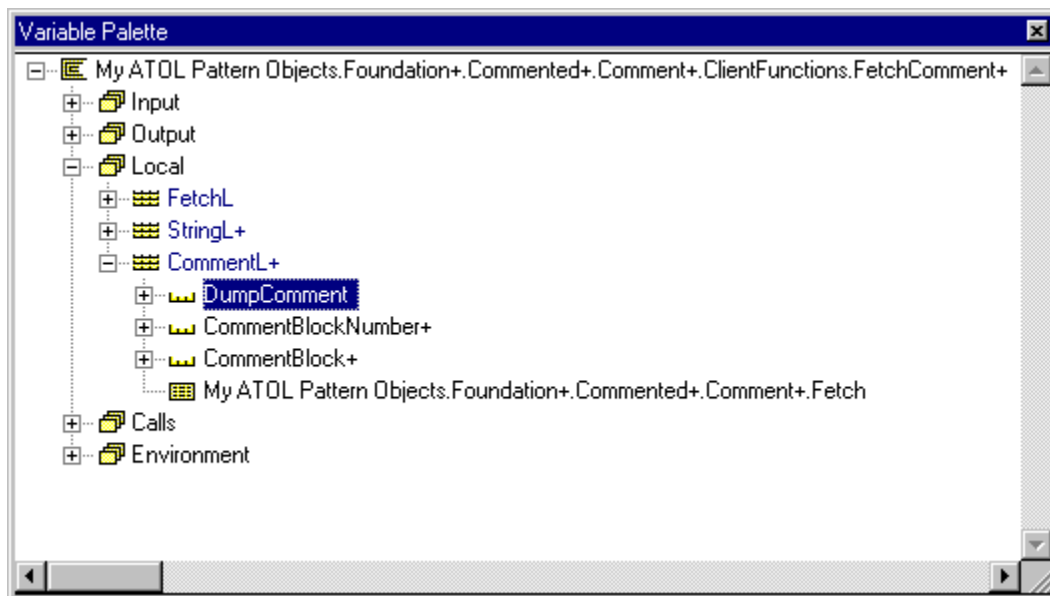
| | | |
|-------------------------------------------------------|---------------|-----------------------|
| itions.FetchComment+.ConcatComment+ | parameter FLD | CommentText+ |
| Source code | <All> | Field |
| My ATOL Pattern Objects.Foundation+.Commented+.Commer | parameter | UISTYLE+/CommentText+ |
| | | UISTYLE+/String+ |

3. STEP 3 :

Declare a new field Which inherits from CommentText+, for example DumpComment.

DumpComment is a UI STYLE+/CommentText+.

Add this field in the CommentL+ variable of your ...Comment+.ClientFunctions.FetchComment+ function.



4. STEP 4 :

Add the following lines in the your ...Comment+.ClientFunctions.FetchComment+ function (subroutine CallBlockFetch+)

```

Pre Point Process data fetched+
Set CommentL+<DumpComment> = Output<CommentText+>
API Call Source code: My ATOL Pattern Objects.Foundation+.Commented+.Comment+.ClientFunctions.FetchComment+.ConcatComment+
Set Output<CommentText+> = <CommentText+.*Null>

Edit Point Process data fetched+
... Set Output<CommentText+> = Output<CommentText+> CONCAT StringL+<String+>
Format Message Message: My ATOL Pattern Objects.Foundation+.Commented+.Comment+.ClientFunctions.FetchComment+.ConcatComme

Post Point Process data fetched+
Set Output<CommentText+> = CommentL+<DumpComment>

```



Map parameter for API call as followed :

| To | From |
|----------------|------------------------|
| <CommentText+> | CommentL+<DumpComment> |
| <String+> | StringL+<String+> |

5. NOTES :

As the ...Comment+.ClientFunctions.FetchComment+ inherits from UISTYLE+/FetchComment+. If you have a customisation layout for UISTYLE+ functios you should do these modification in that customised function.