

Tech tools

Links to other tools that may be useful when dealing with Salesforce

- [Excel that convert SF datetimes to Excel format and vice versa](#)
- [PSCSV](#)
- [@tsalb's "getting started with VSCODE" list](#)
- [Getting the Case Reference Id](#)
- [Useful Regex](#)
- [Reset Passwords Using Dev Console](#)

Excel that convert SF datetimes to Excel format and vice versa

```
Public Function ConvertSFDateTime(ctl As Range) As Variant

Dim strSFDateTime As String, strSFDatePortion As String, strSFTimePortion As String, strCel As String

Dim sglCharT As Single, sglCharZ As Single, sglTimeLen As Single

Application.Calculation = xlCalculationManual
Application.ScreenUpdating = False
Application.EnableEvents = False
Application.DisplayStatusBar = False

strCel = CStr(ctl.Value2)

If strCel <> vbNullString Then

    ' +1 added, so that it returns the rightmost part of the string starting at the first
    character after "T"
    sglCharT = InStr(1, strCel, "T") + 1
    sglCharZ = InStr(1, strCel, "Z")
    sglTimeLen = sglCharZ - sglCharT - 1

    strSFDatePortion = Left(strCel, 10)

    strSFTimePortion = Mid(strCel, sglCharT, sglTimeLen)

    ConvertSFDateTime = Format(DateValue(strSFDatePortion) + TimeValue(Replace(strSFTimePortion,
    ".00", "")), _
    "yyyy-mm-dd hh:mm:ss")
```

```

Application.Calculation = xlCalculationAutomatic
Application.ScreenUpdating = True
Application.EnableEvents = True
Application.DisplayStatusBar = True

End If

End Function

Public Function ConvertToSFDateTime(cel As Range) As String

Dim strCel As String, strDatePart As String, strTimePart As String

Dim dateCel As Date

Dim sglFractionalLocation As Single

Application.Calculation = xlCalculationManual
Application.ScreenUpdating = False
Application.EnableEvents = False
Application.DisplayStatusBar = False

' VBA won't handle fractional seconds, finds the character number in the string where a period
occurs
sglFractionalLocation = InStr(1, cel, ".")

' Strips out the fractional second
If sglFractionalLocation > 0 Then
    strCel = Left(cel, sglFractionalLocation - 1)
Else
    strCel = cel
End If

If IsDate(strCel) Then
    strDatePart = DateValue(strCel)
    ' Determines if time component exists in the value of cel
    If DateValue(strCel) = CDate(strCel) Then
        ConvertToSFDateTime = Format(strCel, "yyyy-mm-dd")
    Else

```

```

        ConvertToSFDateTime = Format(strCel, "yyyy-mm-ddThh:mm:ssZ")
    End If
End If

Application.Calculation = xlCalculationAutomatic
Application.ScreenUpdating = True
Application.EnableEvents = True
Application.DisplayStatusBar = True

End Function

Public Function ConcatenateMult(rngConcatenateCells As Range, bAddSingleSpace As Boolean, _
Optional strDelimiter As String, Optional strWrapCellValue As String)

Dim rngCel As Range, rngLastCel As Range

Dim bCompareCells As Boolean, bWrapCellValue As Boolean

Dim lngLastRow As Long, lngLastCol As Long

Dim strWrapTrue As String, strWrapFalse As String

Application.Calculation = xlCalculationManual
Application.ScreenUpdating = False
Application.EnableEvents = False
Application.DisplayStatusBar = False

' Sets the bounds for the last row and column from the rngConcatenateCells argument
lngLastRow = rngConcatenateCells.Row + rngConcatenateCells.Rows.Count - 1
lngLastCol = rngConcatenateCells.Column + rngConcatenateCells.Columns.Count - 1

Set rngLastCel = Cells(lngLastRow, lngLastCol)

' Loops through each cell in rngConcatenateCells
' If bAddSingleSpace is True, adds a single space in-between concatenated cells
' If optional strDelimiter argument is provided, adds delimiter between each concatenated
value
' If optional strWrapCellValue argument is provided, wraps each concatenated cell with that

```

delimiter

For Each rngCell In rngConcatenateCells

bCompareCells = (rngCell.Address = rngLastCell.Address)

' Sets bWrapCellValue to true and applies strWrapCellValue,

' only if the length of the argument is greater than 0 characters

'

' NOTE: [Uses CStr(Replace(rngCell, "'", "\'")) to add the escape "\"" character

' before all apostrophes, so errors don't occur with SQL queries

'

' Change this back to CStr(rngCell) if you need to concatenate values without the escape]

If Len(strWrapCellValue) > 0 Then

bWrapCellValue = True

strWrapTrue = strWrapCellValue & CStr(Replace(rngCell, "'", "\'")) & strWrapCellValue

Else

bWrapCellValue = False

strWrapFalse = CStr(Replace(rngCell, "'", "\'"))

End If

' bCompareCells determines if the cell's address is equal to the last cell's address in the range

Select Case bCompareCells

' If true, the strDelimiter and strWrapCellValue arguments are not applied; function ends

Case Is = True

If bWrapCellValue Then ConcatenateMult = ConcatenateMult & strWrapTrue & " "

If Not bWrapCellValue Then ConcatenateMult = ConcatenateMult & strWrapFalse & " "

' If false, continues to concatenate cells

Case Is = False

Select Case Len(strDelimiter)

Case Is = 0

' Adds a single space in between values

If bAddSingleSpace Then

If bWrapCellValue Then ConcatenateMult = ConcatenateMult & strWrapTrue & " "

If Not bWrapCellValue Then ConcatenateMult = ConcatenateMult & strWrapFalse & " "

Else

```

                If bWrapCellValue Then ConcatenateMult = ConcatenateMult & strWrapTrue
                If Not bWrapCellValue Then ConcatenateMult = ConcatenateMult &
strWrapFalse
            End If
        Case Else
            If bAddSingleSpace Then
                If bWrapCellValue Then ConcatenateMult = ConcatenateMult & strWrapTrue
& strDelimiter & " "
                If Not bWrapCellValue Then ConcatenateMult = ConcatenateMult &
strWrapFalse & strDelimiter & " "
            Else
                If bWrapCellValue Then ConcatenateMult = ConcatenateMult & strWrapTrue
& strDelimiter
                If Not bWrapCellValue Then ConcatenateMult = ConcatenateMult &
strWrapFalse & strDelimiter
            End If
        End Select
    End Select
Next rngCel

```

```

' Hardcoded parentheses added for SFDC conversions -
' Apex Data Loader, Force.com Explorer, and other SOQL query tools require format of values
' in an IN clause as:

```

```

'
' ('[value1]', '[value2]', '[value3]', ..., '[value(n)]')
'
' (e. g. Select Id, Name FROM Contact WHERE Account.Name IN ('Kimberly-Clark', 'IBM'))
ConcatenateMult = "(" & Trim(ConcatenateMult) & ")"

```

```

Application.Calculation = xlCalculationAutomatic
Application.ScreenUpdating = True
Application.EnableEvents = True
Application.DisplayStatusBar = True

```

```

End Function

```

PSCSV

<https://github.com/SFXD/PSCSV>

A powershell script to help Salesforce admins and consultant to save time and do data load operations without having to rely on Excel.

Current list of stuff the script can do:

- transcode a file from one encoding or separator to another
- remap values in a column from one to another - reformat Dates from whatever format to the salesforce format
- reformat Date Times from whatever format to the salesforce format
- replace whatever you want in a column with ""
- great if someone is sending you files with `null`.

@tsalb's "getting started with VSCODE" list

1) To get started on VSCode + SFDX CLI:

- <https://forcedotcom.github.io/salesforcedx-vscode/articles/getting-started/install>.
- Focus on the "Org Development" model, which you can target production/sandboxes to pull metadata.
- You would need to create a "SFDX Project with Manifest".

2) Look at the VSCode / Environment / SFDX CLI modules (Ignore anything about LWC and JS):

- <https://trailhead.salesforce.com/users/tsalb/trailmixes/lwc-and-js-101>.

3) VSCode + ForceCode is also a possibility (you need to set up VSCode first, but you can ignore SFDX CLI / Salesforce Extensions)

- <https://marketplace.visualstudio.com/items?itemName=JohnAaronNelson.ForceCode>.
- ForceCode can help you create a package.xml easier, which is a manifest of metadata you wish to pull.

4) This is a tool to help you construct a package.xml if you decide to go with SFDX CLI + Salesforce Extensions

- I do not use this, I cannot vouch for the security.
- <https://packagebuilder.herokuapp.com/>
- COPY PASTE whatever the output of this is into your package.xml in your SFDX manifest project.

5) I found this vscode extension which can generate you a raw package.xml. I use this with my SFDX org development.

- This one i do use, it seems to work nicely (installed on 2019-06-20)
- <https://github.com/vignaesh01/sfdx-package-generator>

Getting the Case Reference Id

in email `{! Case. Thread_Id}`

in apex/other

After Winter'16: Your org still has the broken ref ID, but we no longer parse them, because there is no expectation that we would, and the org ID is incorrect. And we no longer parse it due to a change on our end where we tightened the org ID requirement.

For your reference you can look at below example on how reference it is generated - WE DON'T SUPPORT THE FORMULA BELOW AND FORMAT CAN CHANGE AT ANY TIME:

Organization ID Format: 00DXXYYYYYYYYYYY

Case ID Format: 500AABBBBBBBBBBB

Acceptable Thread ID Formats: (1) ref:_00DXXyyyyyyyyyyy._500AAbbbbbbbbbbb:ref (2)
ref:00DXyyyyyyyyyyy.500Abbbbbbbbbbb:ref

For example (example org id): Organization ID = 00D5000000IQwR Case ID = 5005000000PQo5L

Thread ID = ref:_00D50IQwR._50050PQo5L:ref

Custom Formula:

```
"ref:_"&LEFT(
$Organization. ID, 5) &SUBSTITUTE( RIGHT( $Organization. ID, 10), "0", "" ) &"._"&LEFT( Id, 5) &SUBSTITUTE( L
EFT( RIGHT( Id, 10), 5), "0", "" ) &RIGHT( Id, 5) &":ref"
```

NOTE: An important disclaimer here is that any formula you use for this purpose could one day begin failing if the format of the Case ID or Thread/Reference ID changes.

Useful Regex

Profile Permissions

Use this regex to massively replace profile permissions in your editor.

Note that for the `.profile` to contain the field permissions, you will have to query both at the same time. This is because the metadata API only gives permissions tied to what you are currently requesting - which also explains some changeset behaviour when deploying profiles.

Replace `YOUROBJECTNAMEHERE` byt the name of the object you want to set the permissions for. This regex finds permissions for all Custom fields on that object and sets them to TRUE/TRUE (editable, writable).

If you wish to replace permissions accross all custom objects for example, simply replace

`YOUROBJECTNAMEHERE` by `.*__c`

Find:

```
( <fieldPermissions>\s*<editable>).*(</editable>\s*<field>YOUROBJECTNAMEHERE.*__c</field>\s*<readable>).*(</readable>\s*</fieldPermissions>)
```

Replace:

```
$1TRUE$2TRUE$3
```

Reset Passwords Using Dev Console

1. Open Developer Console
2. Click the Debug dropdown menu and select the Open Execute Anonymous Window
3. Enter the following to manually set a new password for a user:

```
User usr = [select Id from User where username=' ENTER_USERNAME_HERE' ];  
System.setPassword(usr.Id, ' ENTER_NEW_PASSWORD_HERE' );
```