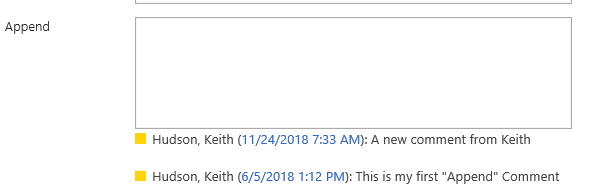
# Sidestepping the "Append Changes to Existing Text" booby trap in SharePoint

SharePoint 2013 contains some booby traps that can catch users unawares. This lesson will provide some ways to deal with one of those booby traps.

In a SharePoint list, when adding a multi-lines of text column, the user can choose to "Append Changes to Existing Text"



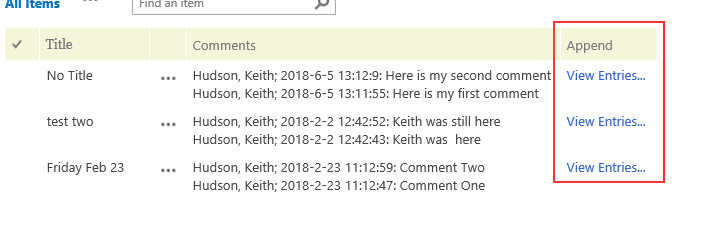
This sounds very inviting, as it evokes a picture of being able to create a running record of comments added to the field. Unfortunately, it is a booby trap, as nothing gets appended. The way this setting works is that every time someone edits the item and adds text to this field, all previous entries in the field are removed, and the new text is saved in the field. On the display and edit forms, previous entries are shown below the field, with a time and author stamp, as you can see below on the Edit form:



On the Display form, it looks like this:



Although this works fine when one is using just the Edit or Display form, when looking at a view of the list, the text entries don't show at all, just a link to go see them:



When you migrate to a new version of SharePoint on premises, many companies will dictate that you can only migrate one version of your list items. In that event, you lose all but the latest entry in your text field. (SharePoint Online users, be glad you don't have that problem!) Even if you are allowed to migrate 5 or 10 versions of list items, an active list could easily have 60 or 100 versions, and only keeping 10 versions causes all previous entries in the text field to be lost.

Instead of relying on the mis-named "Append Changes to Existing Text" setting, I use DFFS to accomplish something very similar, but without the drawbacks. There are some variations on this approach, but for this article I will show just one.

**Enhanced rich text field using DFFS**

1. Create a multi-lines of text field named "Notes" and choose "Enhanced Rich Text". After creating the field, you can rename the field display name as desired.
2. Add the following code to the CustomJS section of your DFFS config in the new form and the edit form:

function addNote(){

//get field value

var oldComment = getFieldValue('Notes');

//get date and time

var dateTime = spjs.dffs.buildValStr("{timestamp}");

//get current logged in user

var userInfo = spjs.utility.userInfo(spjs.dffs.data.currUserID);

var user = userInfo.Title;

//set field values

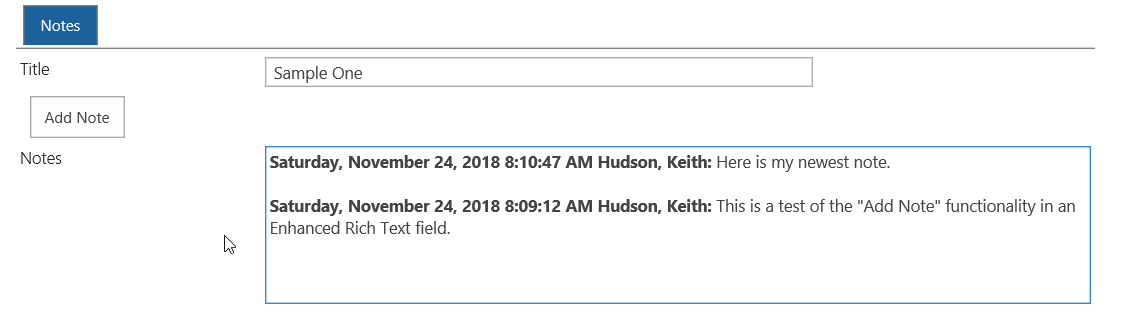
setFieldValue('Notes','<b>' + dateTime + ' ' + user + ':</b> <br>' + oldComment );

}

1. Put an HTML section on your form with a button to run the function. I usually put the button just above my Notes field. Here is sample code for the button: <button type='button' onclick='addNote()' >Add Note</a>

If you already have an existing multi-line field, you can modify the above code to use the internal name of your existing field. You can also change the "Add Note" text showing on the button to suit your needs.

Here's a screenshot of what the result looks like:



If you don't want the blank line separating the entries, simply remove the <br> from last line of js code.

Keith Hudson

November, 2018