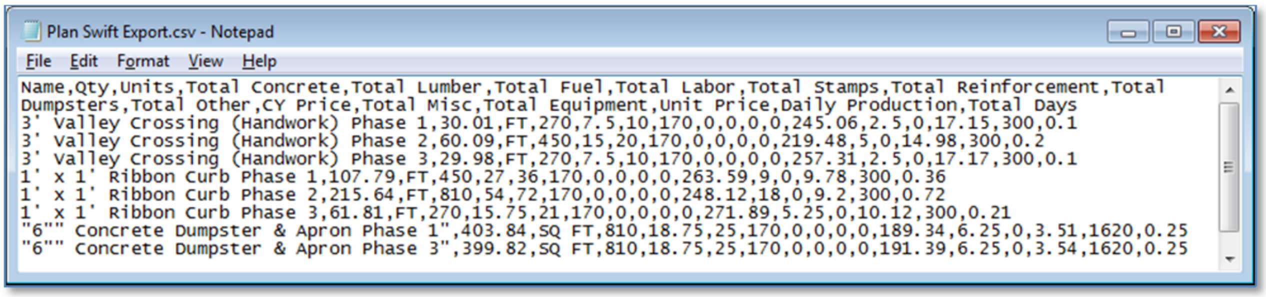
# PlanSwift Integration

## *Consuming exported proposal line items from PlanSwift*

PlanSwift is a well-recognized construction take-off application. Tincher Concrete utilizes it and we want to link with it. All of the mathematics associated with estimating the material, labor, rental, … is done in PlanSwift. It then has the ability to export the budget to a spreadsheet. From there we simply “save-as” a CSV file.

The current content of the CSV file looks like this:

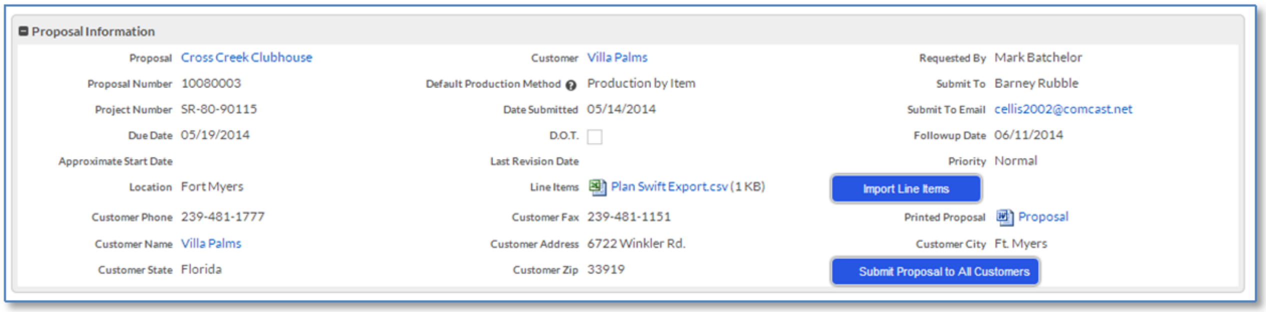


```
Name,Qty,Units,Total Concrete,Total Lumber,Total Fuel,Total Labor,Total Stamps,Total Reinforcement,Total
Dumpsters,Total Other,CY Price,Total Misc,Total Equipment,Unit Price,Daily Production,Total Days
3' Valley Crossing (Handwork) Phase 1,30.01,FT,270,7.5,10,170,0,0,0,0,245.06,2.5,0,17.15,300,0.1
3' Valley Crossing (Handwork) Phase 2,60.09,FT,450,15,20,170,0,0,0,0,219.48,5,0,14.98,300,0.2
3' Valley Crossing (Handwork) Phase 3,29.98,FT,270,7.5,10,170,0,0,0,0,257.31,2.5,0,17.17,300,0.1
1' x 1' Ribbon Curb Phase 1,107.79,FT,450,27,36,170,0,0,0,0,263.59,9,0,9.78,300,0.36
1' x 1' Ribbon Curb Phase 2,215.64,FT,810,54,72,170,0,0,0,0,248.12,18,0,9.2,300,0.72
1' x 1' Ribbon Curb Phase 3,61.81,FT,270,15.75,21,170,0,0,0,0,271.89,5.25,0,10.12,300,0.21
"6"" Concrete Dumpster & Apron Phase 1",403.84,SQ FT,810,18.75,25,170,0,0,0,0,189.34,6.25,0,3.51,1620,0.25
"6"" Concrete Dumpster & Apron Phase 3",399.82,SQ FT,810,18.75,25,170,0,0,0,0,191.39,6.25,0,3.54,1620,0.25
```

## Importing this data into Rollbase is a simple 3-step process

### *Step 1*

The first step is to make sure you have a File Upload field defined. I’ve created one called Line Items. Once you’ve created that field, added it to the UI, then you are ready to upload your exported file into the system. As you can see in the following screen snippet, I’ve uploaded the file “Plan Swift Export.csv” to the proposal.



Proposal Information

| | | | | | |
|---|---|---|---|---|---|
| Proposal | Cross Creek Clubhouse | Customer | Villa Palms | Requested By | Mark Batchelor |
| Proposal Number | 10080003 | Default Production Method | Production by Item | Submit To | Barney Rubble |
| Project Number | SR-80-90115 | Date Submitted | 05/14/2014 | Submit To Email | cellis2002@comcast.net |
| Due Date | 05/19/2014 | D.O.T. | ☐ | Followup Date | 06/11/2014 |
| Approximate Start Date | | Last Revision Date | | Priority | Normal |
| Location | Fort Myers | Line Items | Plan Swift Export.csv (1 KB) | | |
| Customer Phone | 239-481-1777 | Customer Fax | 239-481-1151 | Printed Proposal | Proposal |
| Customer Name | Villa Palms | Customer Address | 6722 Winkler Rd. | Customer City | Ft. Myers |
| Customer State | Florida | Customer Zip | 33919 | | |

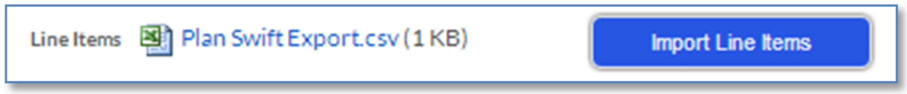
### *Step 2*

The second step is to create an Object Script Trigger to read and parse the contents of the file upload field. I’ve created a trigger called “Import PlanSwift”. The Javascript content of this trigger, along with commented explanation can be found on the next page.

```
//declare a variable to get and hold the text from the file upload field
var importData = rbv_api.getTextData("ccProposal", {!id}, "line_items");
//declare an array to hold each line of information
var lineData = new Array();
//declare an array to hold all fields for one line
var fieldData = new Array();
//populate the lineData array by using the string.split() function splitting
//on carriage returns “\r”
lineData = importData.split("\r");
//loop through the lines and extract the field content
for (var i = 1;i < lineData.length - 1; i++){
  //assign the current line to a string
  var s = lineData[i];
  //populate the fieldData array by spitting the string s by commas.
  fieldData = s.split(",");
  //create an array to hold the data we intend on sending to Rollbase.
  var aLine = new Array();
  //The following 4 lines is a creative way to eliminate double quote
  //issues. If you look at the CSV screen shot, you will see some records
  //that has one or more double quotes in the record name. This is common,
  //as a double quote is used to represent inches, which is significant.
  //Basically we replace all double occurrence of double quotes “” with a
  //tilde (~). Then we replace all remaining quotes with blanks.
  //Lastly, we replace the tilde with the desired double quote.
  var recordName = fieldData[0];
  recordName = recordName.replace('""',"~");
  recordName = recordName.replace('"',"");
  recordName = recordName.replace('~','"');
  //Populate all fields we need for the proposal line item.
  aLine["R97484106"] = {!id};
  aLine["ccProposalLineItem_quantity"] = parseFloat(fieldData[1]);
  aLine["ccProposalLineItem_rate"] = parseFloat(fieldData[14]);
  aLine["line_item_description"] = recordName;
  aLine["name"] = recordName;
  aLine["unit_of_measure"] = fieldData[2];
  aLine["concrete"] = parseFloat(fieldData[3]);
  aLine["lumber"] = parseFloat(fieldData[4]);
  aLine["fuel"] = parseFloat(fieldData[5]);
  aLine["labor"] = parseFloat(fieldData[6]);
  aLine["stamps"] = parseFloat(fieldData[7]);
  aLine["reinforcement"] = parseFloat(fieldData[8]);
  aLine["dumpsters"] = parseFloat(fieldData[9]);
  aLine["other"] = parseFloat(fieldData[10]);
  aLine["miscellaneous"] = parseFloat(fieldData[12]);
  aLine["ccProposalLineItemDaily_production"] = parseFloat(fieldData[15]);
  aLine["ccProposalLineItemsDays_allowed"] = parseFloat(fieldData[16]);
  aLine["ccProposalLineItem_display_order"] = parseInt(i);
  //Insert the record into Rollbase.
  rbv_api.createRecord("ccProposal_line_item", aLine);};
```

## *Step 3*

The final step is to add a script element to your user interface that has a button which will call the trigger we just built.  You can see the button in the original screen shot earlier, but here you can see what we are trying to accomplish.



Since we have already covered how to create nice looking buttons in Javascript, I'm only going to focus on the Javascript and HTML used for this button, ignoring all of the CSS style information which is required.

At the end of this script, the code looks like this:

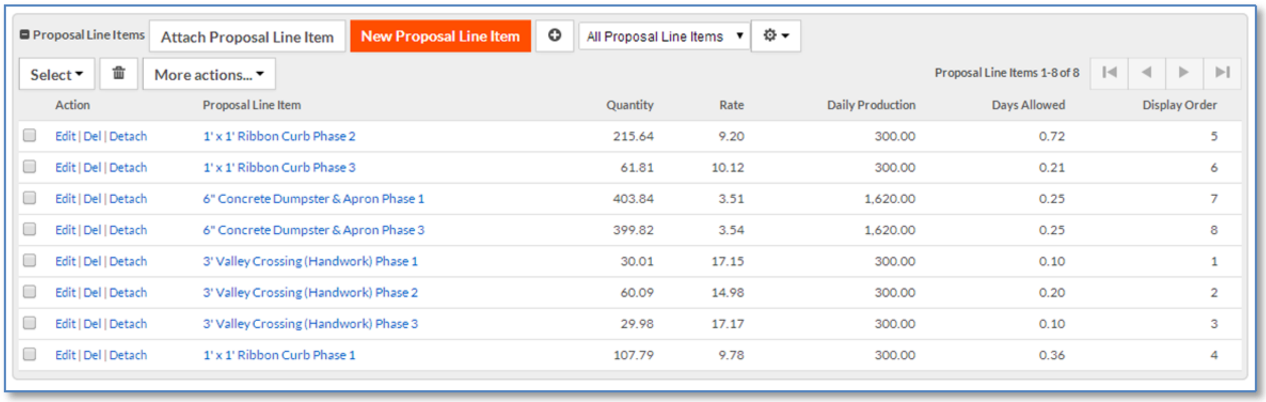
```
<script>
  function myCallback(){
  }

  function doImport(){
    rbf_runTrigger("ccProposal", {!id}, 97592987, false, myCallback);
    document.location.reload(true);
  }

</script>

</head>
<body>
<input id="gobutton" type="submit" onclick="doImport()" value="Import
Line Items" />
</body>
</html>
```

That's all it takes, just a simple AJAX call to run the trigger, wrapped in a function, and the assignment of that function to the onClick event of the button we create in HTML. Note: Once the user clicks the button, it will be necessary for them to refresh (F5) their browser to have the imported records show up. Clicking on the button, and hitting F5 confirm that we are getting our data.



| | Action | Proposal Line Item | Quantity | Rate | Daily Production | Days Allowed | Display Order |
|---|---|---|---|---|---|---|---|
| | Edit \| Del \| Detach | 1' x 1' Ribbon Curb Phase 2 | 215.64 | 9.20 | 300.00 | 0.72 | 5 |
| | Edit \| Del \| Detach | 1' x 1' Ribbon Curb Phase 3 | 61.81 | 10.12 | 300.00 | 0.21 | 6 |
| | Edit \| Del \| Detach | 6" Concrete Dumpster & Apron Phase 1 | 403.84 | 3.51 | 1,620.00 | 0.25 | 7 |
| | Edit \| Del \| Detach | 6" Concrete Dumpster & Apron Phase 3 | 399.82 | 3.54 | 1,620.00 | 0.25 | 8 |
| | Edit \| Del \| Detach | 3' Valley Crossing (Handwork) Phase 1 | 30.01 | 17.15 | 300.00 | 0.10 | 1 |
| | Edit \| Del \| Detach | 3' Valley Crossing (Handwork) Phase 2 | 60.09 | 14.98 | 300.00 | 0.20 | 2 |
| | Edit \| Del \| Detach | 3' Valley Crossing (Handwork) Phase 3 | 29.98 | 17.17 | 300.00 | 0.10 | 3 |
| | Edit \| Del \| Detach | 1' x 1' Ribbon Curb Phase 1 | 107.79 | 9.78 | 300.00 | 0.36 | 4 |