How can I eliminate unwanted white space with "Run Script at First Pass"?

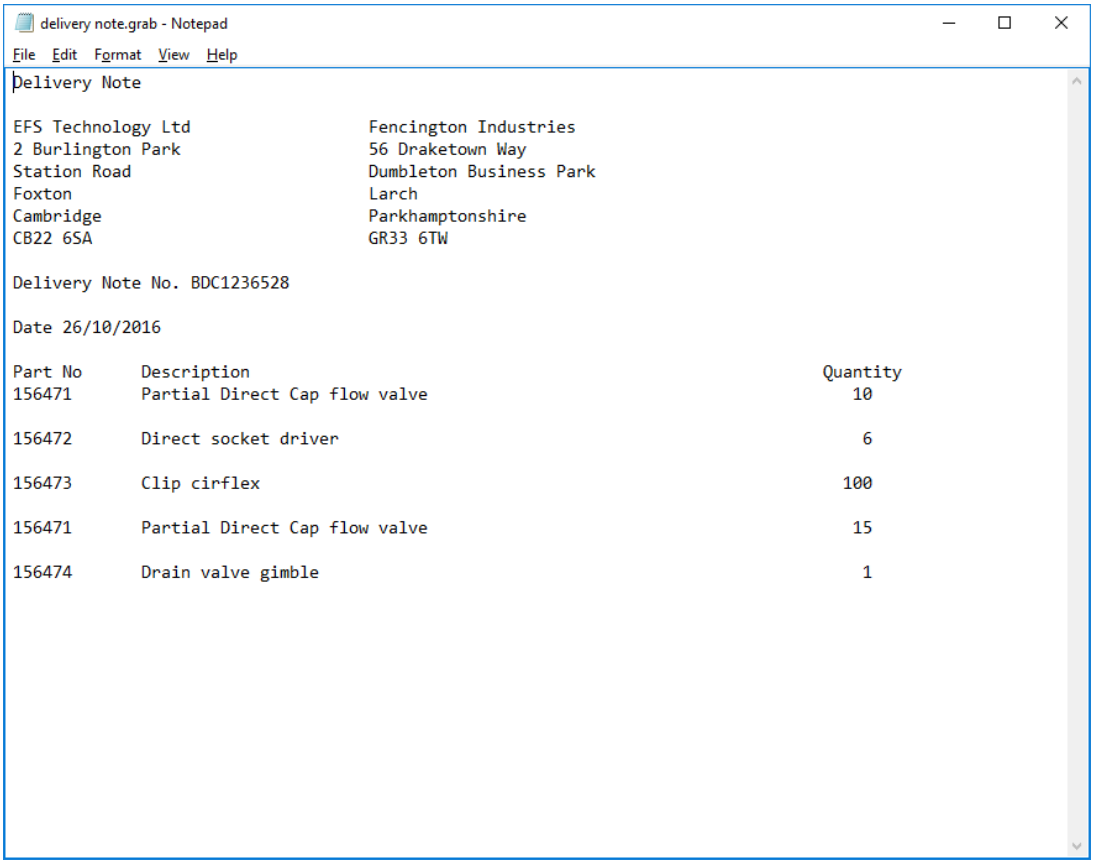
Alex Clemons - 2023-11-22 - Comments (0) - Lasernet FAQs

Lasernet

You can eliminate unwanted white space from your forms with “Run Script at First Pass”. The following article will show you how. This article assumes you are familiar with conditional rearranges, database commands and scripting with Lasernet. It will not go into detail about setting those up.

We will be augmenting an example form with some additional data from another database table, so our customers have more information about each example part we stock.

Currently, our report (shown below), only has a single line of information against each part number:



Whereas our database table (below), has a number of additional lines of useful information for each part number that we would like to include:

	type	Text
▶	156471	Partially flattened screw casing to 0.2mm
	156471	Flap grommet is extended by 1mm
	156471	Partial flange retention to 0.3mm
	156471	ISO 6659-1 retract required
	156471	Quench at 15 degrees
	156472	Multi faceted angle grinder positioned to 0.3mm
	156472	Parts non returnable
	156472	WARNING : worm gear is non replaceable
	156473	Chamfer cam lever to 45 degrees
	156474	Heat treat casing for 12 hours at 200 degrees
*	*NULL*	*NULL*

To solve this problem we could have the standard report altered in the ERP system to include all the information and then map it to the form. However, these alterations (regardless of size) will have to be done every time the ERP system is upgraded and will likely incur additional consultancy costs from your ERP provider. Alternatively, this information can be quickly and easily extracted by Lasernet and placed on the form by using a script to call the database.

Create your own script

The following information can be used as a guide in creating your own script

In this example, the script is called on a rearrange that maps the part number to the page. The part numbers in our data match the “type” field in our database and that is the value that will be passed into the database call

We don’t want the part number to show in the form, so the function is only being used to run the database command i.e. the part number is NOT returned to the page out of the function

The JobInfo array that will be created from the database call will be called “Text”, as that is the name of the field in the database we are extracting:

CODE

```
function testFirstPass() {
job.deleteJobInfo("Text");    //Delete previous jobinfo array called
Text so it is clean

job.setJobInfo("passval", Trim(CurrentText)); //Set Jobinfo passval
to the value that we are currently mapping (current part number) -
```

```
See Notes for more info.

databasecommands["Test sql"].run(job);  //Run the database command
called "Test sql"

return ""; //return blank to the page

}
```

Where the Database Command (“Test sql”) being called is:

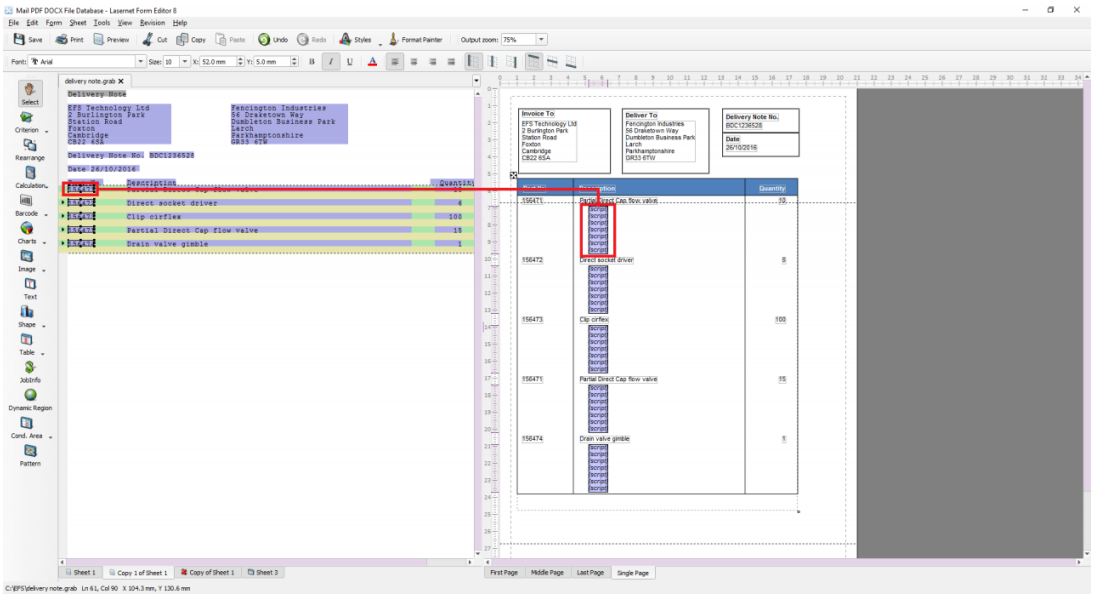
CODE

```
SELECT Text FROM Test WHERE type = '#passval#'
```

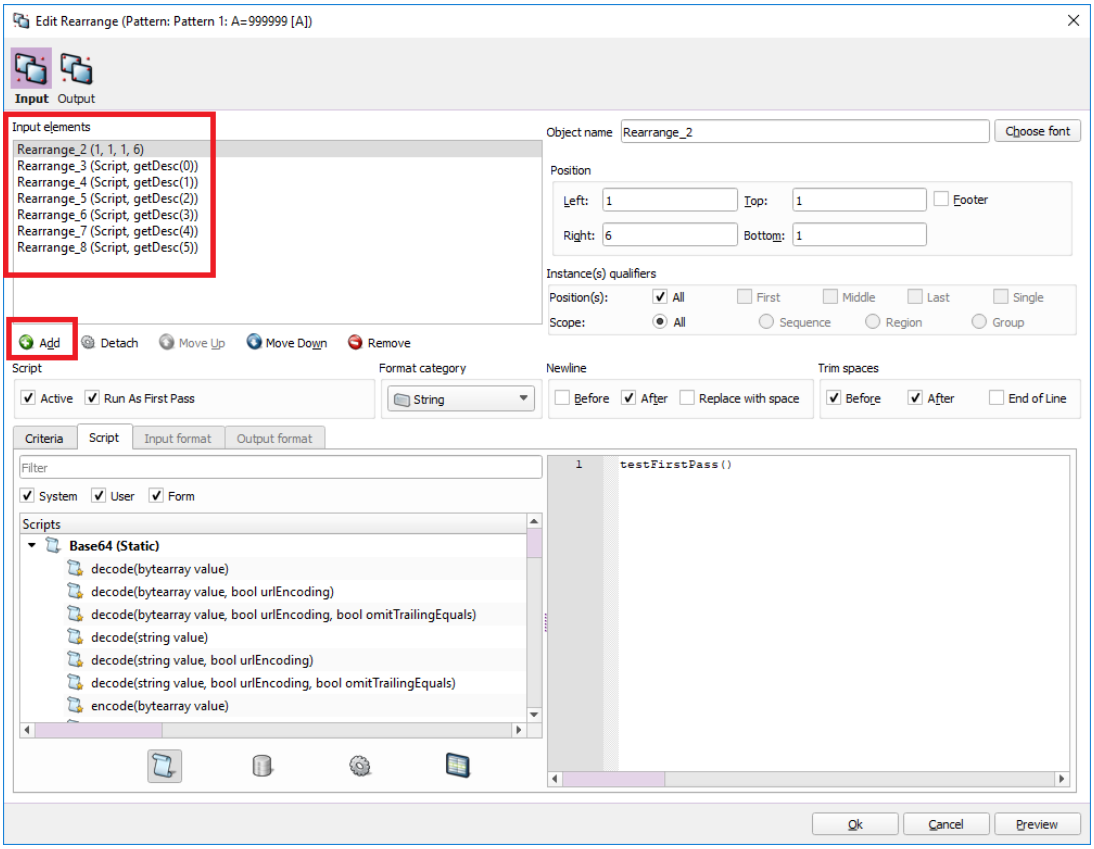
If there is more than one entry in the database for a particular part number you will get back a JobInfo array of values Text[0],Text[1],Text[2}...Text[n]

Follow these steps:

1. Use the image below as a guide to adding the database call script to your form:



2. Once you have applied the javascript call to the part number, use the **Add** button on the same rearrange to add the function getDesc() which will fetch the values from the jobinfo array (“Text”) and add them to the final string.



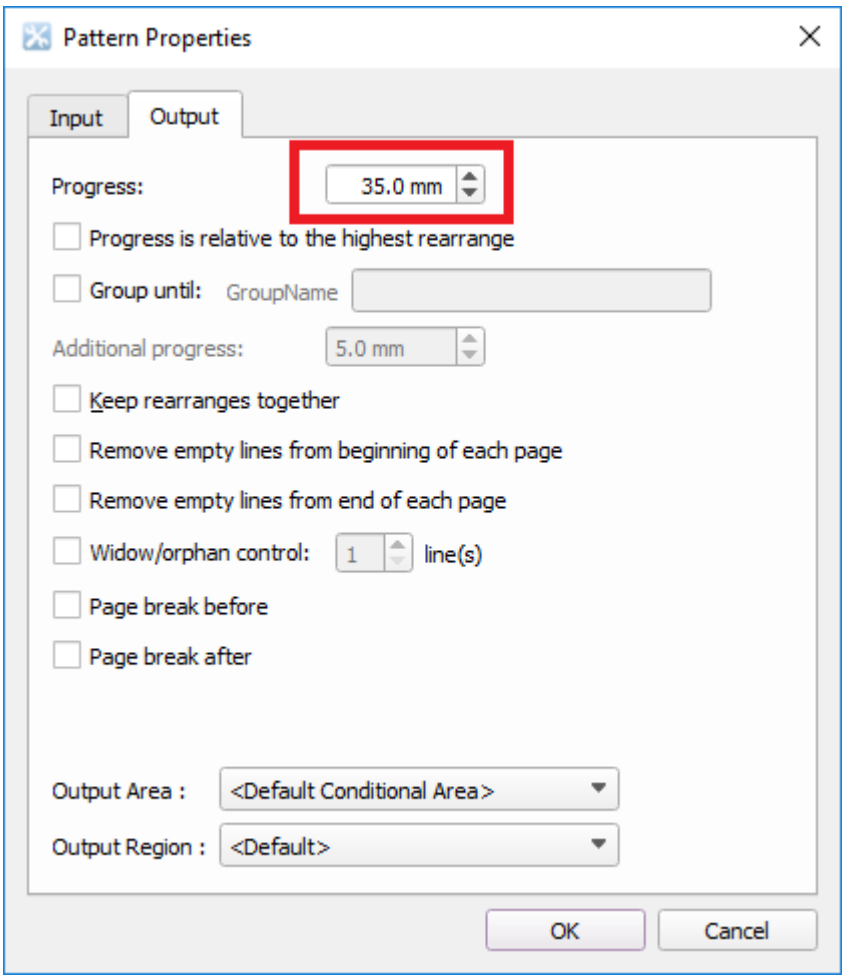
CODE

```
function getDesc(x) {
//Index x is passed into this function and it returns the value at
index x in the job info array return job.getJobInfo("Text",x); }
```

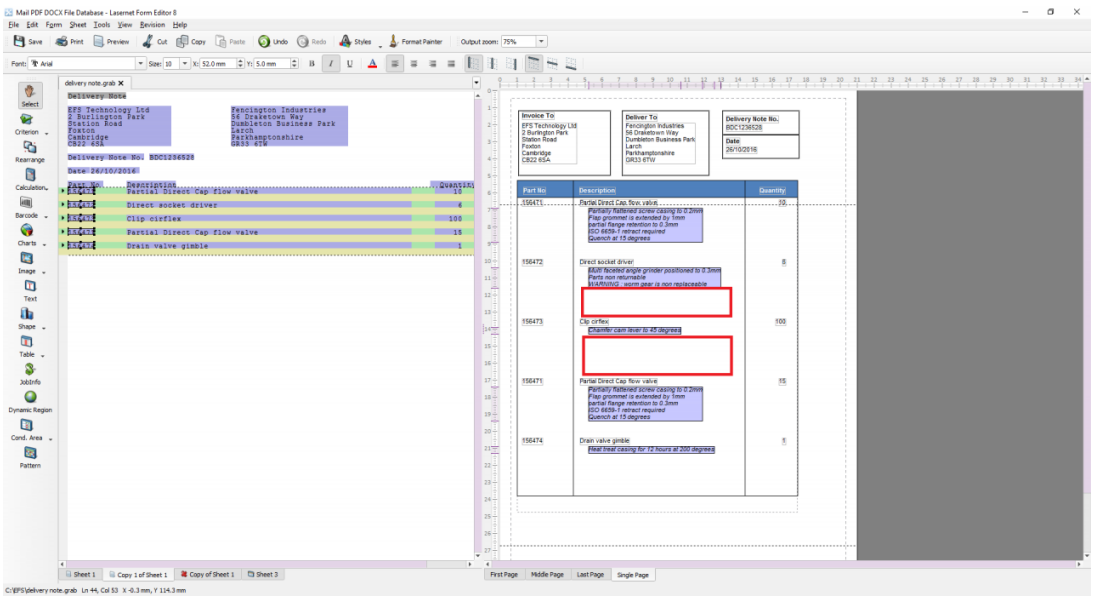
In our example table (FIG 2), you can see that there is a maximum of five possible extra lines of description that can come back from the database call. As such there would need to be five calls to the getDesc() function, each one passing in its own index (0-4).

For each extra description line to be added to a new line, ensure the **new line** after the checkbox is selected on each function call.

3. Ensure that the progress on the conditional pattern that maps the part number is large enough to cater for the maximum amount of extra description lines that can exist. In this case, 35mm would be sufficient enough spacing to ensure the next item does not overwrite the current item. In general terms, the progress value will usually depend on the maximum amount of extra lines that can be returned from your database. Setting the value to 35mm at this stage is done purely to illustrate how extra whitespace can appear in a form. Normally, you would reduce this to a sensible value when the gapping becomes dynamic between each item.



When you run the script in the form (shift+F5), the following would be displayed:

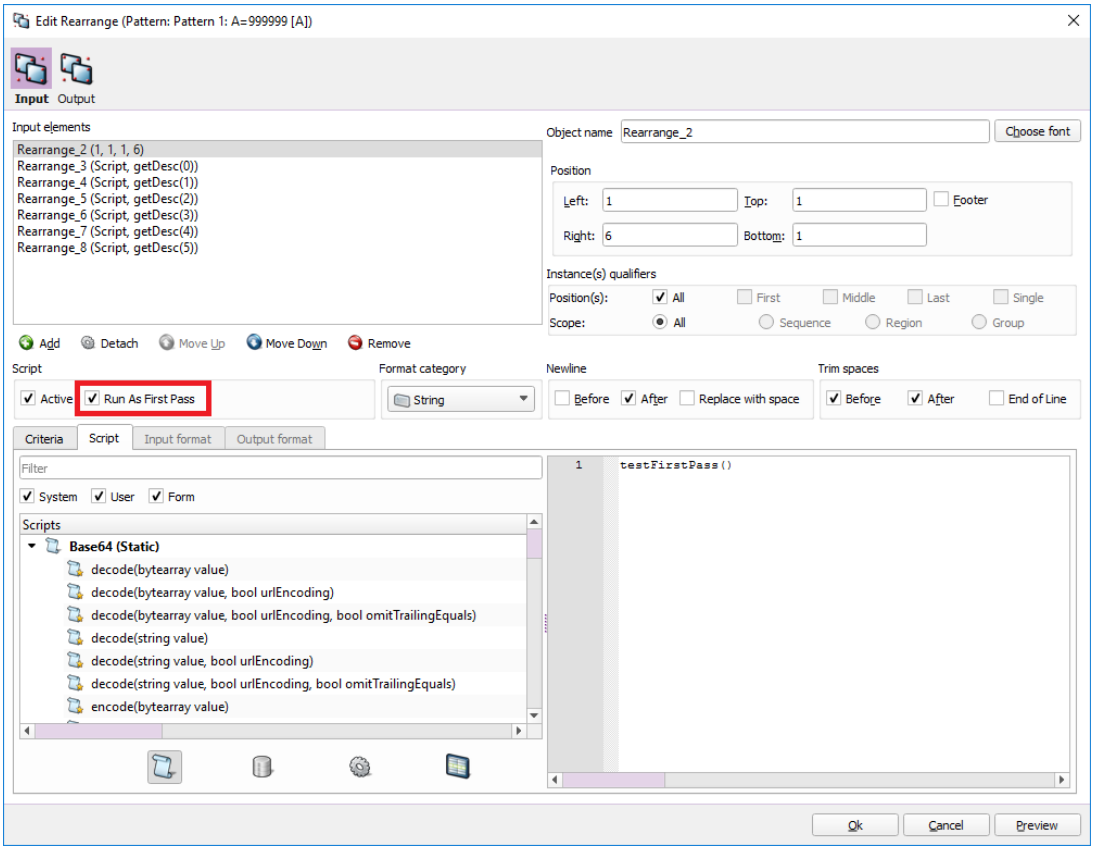


4. As expected, large gaps of white space are now present between items that only have one or two extra lines of description associated with them.

5. Change the progress value on the pattern from 35mm to 10mm and tick the ‘progress is relative to highest re-arrange’ box. If you run the script now (shift+F5), you can see that we still do not get the result that we want. We get the same result with 10mm extra space added than before. Leave the value set to 10mm for now.

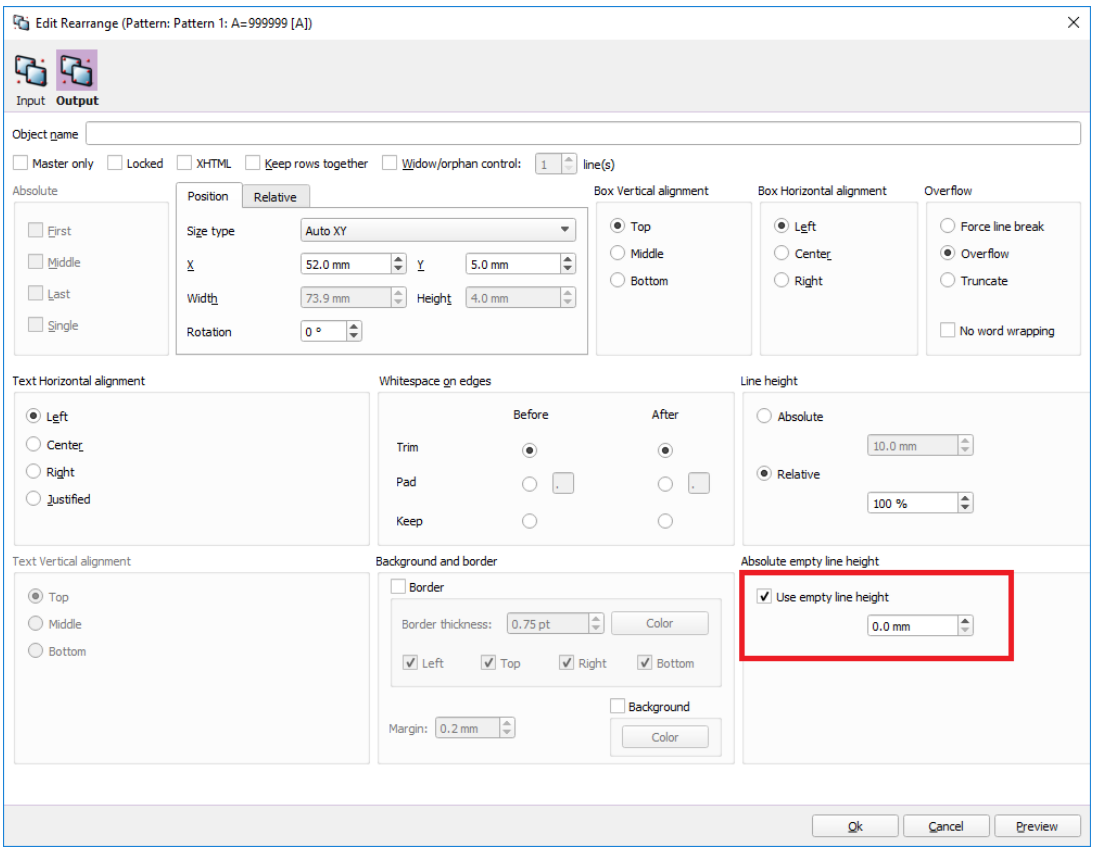
6. Select the **Run Script at First Pass** checkbox for each function call on both TestFirstPass() and each getDesc() in the rearrange.

At this stage, whilst Lasernet is calculating the spacing, it doesn’t yet know how many lines of data are going to be returned by the database call, and for this reason, the “Run Script at First Pass” checkbox was implemented. Its purpose is to run the database call and find out exactly how many lines will be returned to the re-arrange from the database before any other space processing is calculated.

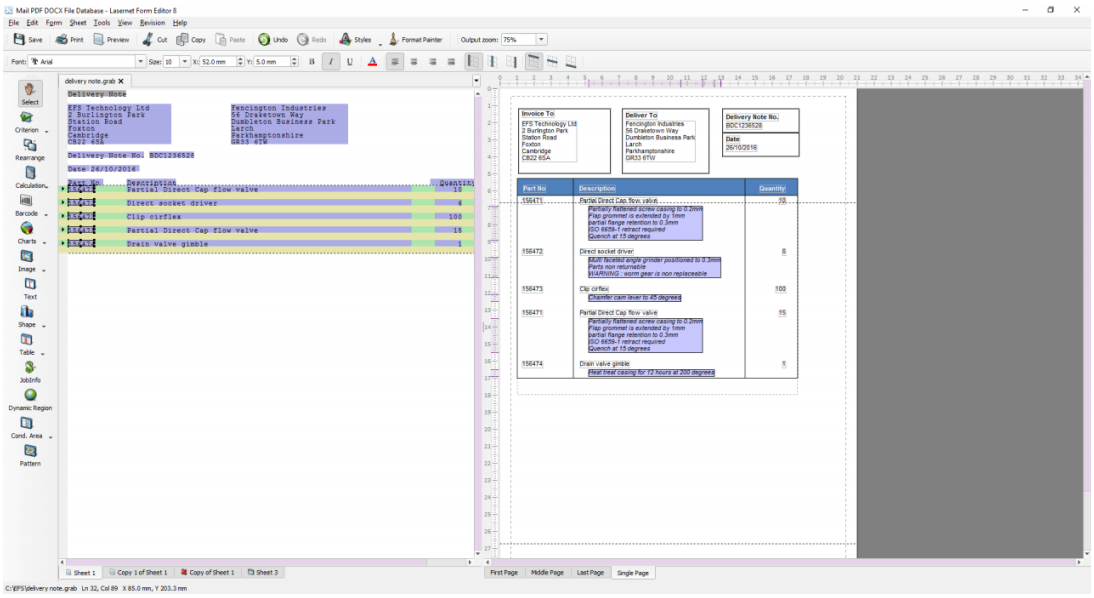


7. The “Progress is relative to the highest Rearrange” will now work correctly because it knows the number of lines that are going to be returned into that space.

8. Set the “Use Empty Line Height” on the rearrange (output tab of the re-arrange). You should select the box and set it to **0mm**, which means if the re-arrange line is blank then no vertical progress is added and any empty lines of description are therefore removed.



After completing these steps, the form is now free of unnecessary white space between items, as shown below:



Because the script is running at first pass, we can only use the **direct object** method of collecting the part number (FIG 9) Trim(CurrentText), because at First Pass Lasernet has no knowledge of the re-arrange name. Likewise, this would not work if we tried collecting the part number via a JobInfo, as again, at first pass phase, Lasernet does not know anything about the values of JobInfos yet.

CODE

```
job.setJobInfo("passval", Trim(CurrentText));
```