COOL REPORTING TECHNIQUE

Non-BIRT Reports Integrated into Maximo

Presented by Stephen Hume, MaxTECH Chair

Agenda

1. **Background**
2. **Method**
3. **Steps to Implement**
4. **The Results**

Stephen Hume

Stephen has been working with IBM Maximo for over ten years in a variety of industries (Oil and Gas, Utilities). He has taught Maximo courses to end users for both Technical and Functional audiences and chairs the MaxTECH User Group.

Background

As part of Health and Safety Review, a group of managers reviewed all completed Incident Records to see if the information was up to date, and records be closed or not.

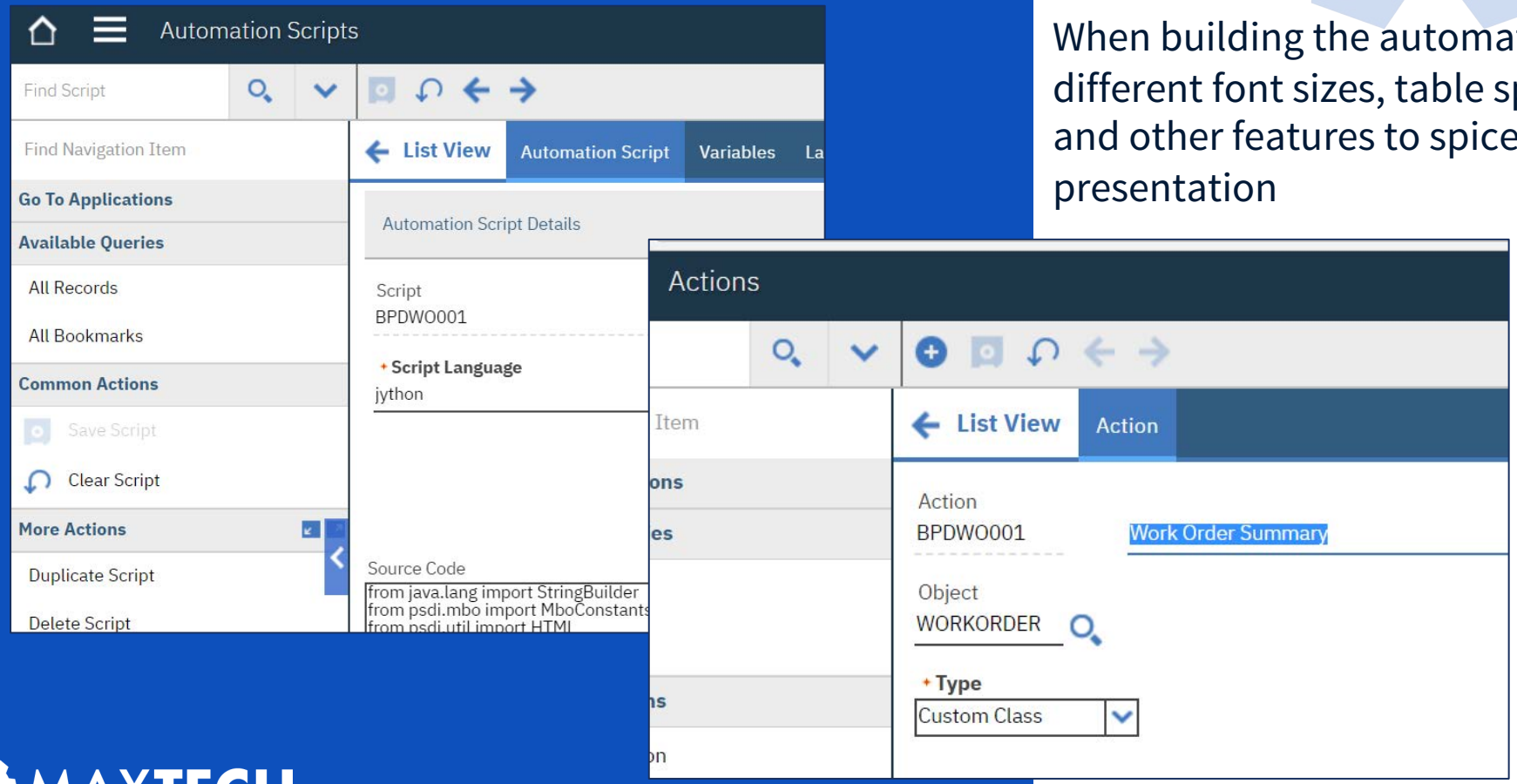
This was a tedious process, clicking on the various tabs in the Incident application to check out the information.

A detailed BIRT report was created which pulled together all of the Incident details, but it was taking too long to run the report for each incident, scroll through the report, close the report, and then workflow the Incident record to CLOSED or Needing further Action status.

Enter a blog posting by Bruno Portaluri where he talked about using an automation script to generate a summary.

Method

This reporting technique uses Automation Scripts, Actions, Application Designer and HTML to make the Summary look great.



When building the automation script, try different font sizes, table spacing, underlines and other features to spice up the summary presentation

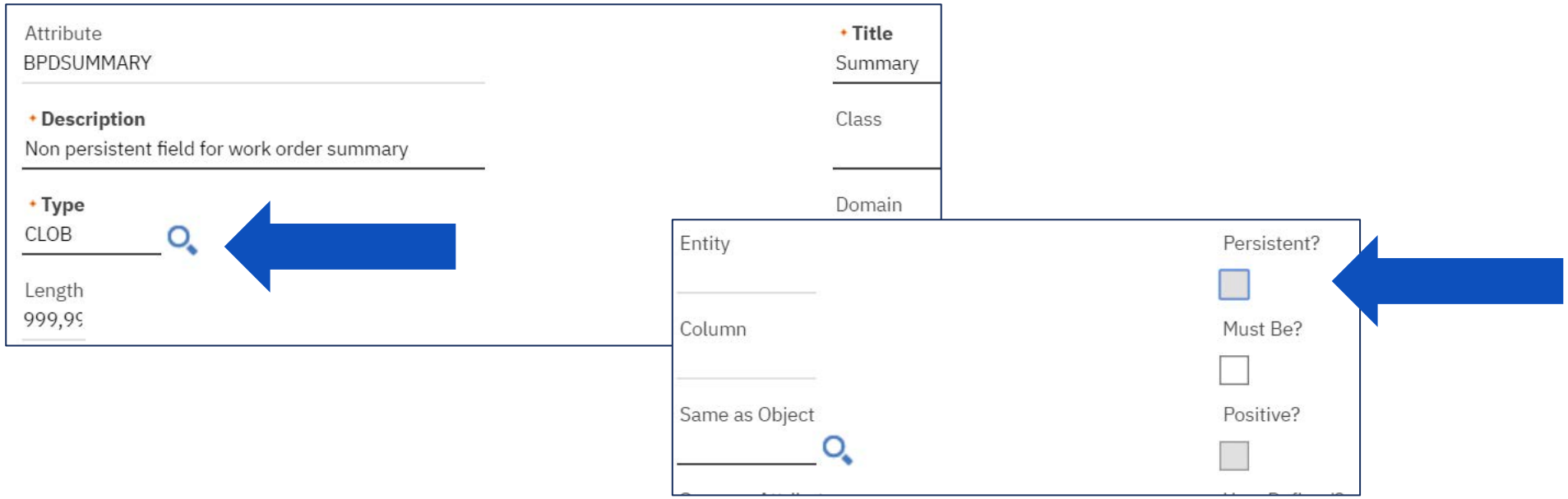
MAXTECH

The Steps to Implement

1. Use Database Configuration to add a NON-PERSISTENT attribute to the main table of the application
2. Write the automation script to populate the new attribute when an action is triggered
3. Add the new tab to the application where the summary will be displayed
4. Give security access to the action so that users can generate the summary

Database Configuration

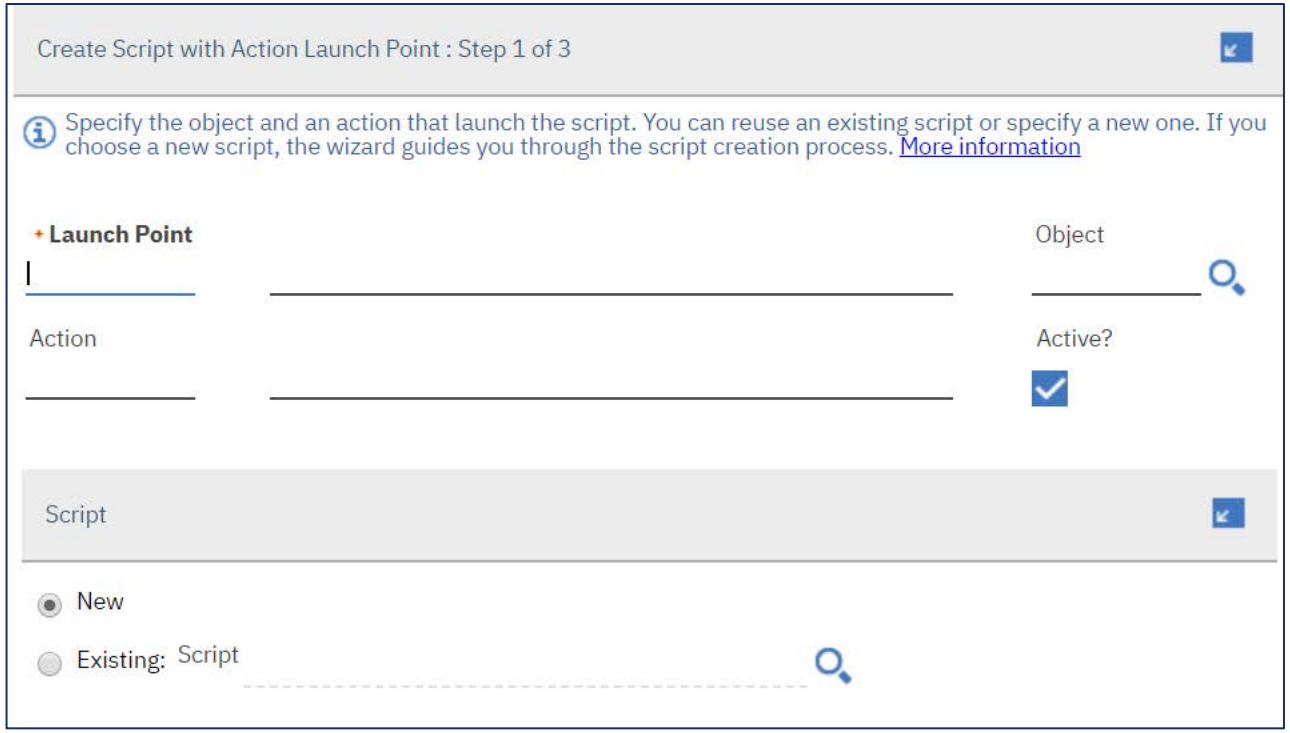
In the example being shown the non-persistent attribute has been added to the work order table.



Automation Script

An automation script needs to be created to populate the summary attribute with data from which ever table is linked to the application where the summary is being implemented.

- Give the Launch Point, Action and Script the same name
- Identify the main Object as the same Object where you added the summary attribute



Automation Script continued…

The SCRIPT breakdown:

Source Code

```
from java.lang import StringBuilder
from psdi.mbo import MboConstants
from psdi.util import HTML
```

At the very top of the script is the list of “functions” that need to be imported into the script for the script to work properly.

They are “StringBuilder”, “MboConstants” and “HTML”

```
val = StringBuilder()

# retrieve attributes and writes them in HTML format in the buffer

#Work Order header row
val.append("<table border=0 width=500>" + "<col width=10%>" + "<col width=10%>" + "<col width=10%>"
                    + "<thead>"
                    + "<tr>"
                    + "<td  width=10%>" + "<font color=#000000 size=-2>" + "<b>" + "WO Number" + "</b>" + "</font>" + "</td>"
                    + "<td  width=10%>" + "<font color=#000000 size=-2>" + "<b>" + "SITE"  + "</b>" + "</font>" + "</td>"
                    + "<td   width=10%>" + "<font color=#000000 size=-2>" + "<b>" + "STATUS"  + "</b>" + "</font>" + "</td>"
                    + "</thead>"
                    + "</table>")
```

The next part of the script builds a string of characters to dynamically construct an HTML document.

Automation Script continued…

The SCRIPT breakdown:

```
#Work Order header row
val.append("<table border=0 width=500>" + "<col width=10%>" + "<col width=10%>" + "<col width=10%>"
                + "<thead>"
                + "<tr>"
                + "<td  width=10%>" + "<font color=#000000 size=-2>" + "<b>" + "WO Number" + "</b>" + "</font>" + "</td>"
                + "<td  width=10%>" + "<font color=#000000 size=-2>" + "<b>" + "SITE"  + "</b>" + "</font>" + "</td>"
                + "<td   width=10%>" + "<font color=#000000 size=-2>" + "<b>" + "STATUS"  + "</b>" + "</font>" + "</td>"
                + "</thead>"
                + "</table>")
```

This part of the script builds the title area for the summary report. It establishes the width of the table, and the widths of each column in the table.

The TR section if the building of a ROW in the table and places data (TD) into three columns in that row.

The effect of this HTML is to generate a row of the Summary that looks like the following:

WO Number	SITE	STATUS

Automation Script continued…

The SCRIPT breakdown:

```
val.append("<table border=0 width=500>" + "<col width=10%>" + "<col width=10%>" + "<col width=10%>"
        + "<tr>"
        + "<td width=10%><font size=-2>" + mbo.getString("WONUM") + "</font></td>"
        + "<td width=10%><font size=-2>" + mbo.getString("SITEID") + "</font></td>"
        + "<td width=10%><font size=-2>" + mbo.getString("STATUS") + "</font></td>"
        + "</font></td>"
        + "</tr>"
        + "</table>")
```

Then a second table is added to the HTML which has the exact same dimensions as the first and this table gets the actual data from the workorder database table to retrieve the WONUM, SITEID and STATUS

Automation Script continued…

The SCRIPT breakdown:

```
val.append("<hr>")
##-------------------------------------------------------------------------------
val.append("<font size=-2><b>RELATED RECORDS</b></font><br>")

# retrieve Related Records records and writes them in the buffer
relatedrecSet = mbo.getMboSet("RELATEDWO")

#related records header row
val.append("<table border=0 width=1000>" + "<col width=10%>" + "<col width=10%>" + "<col width=10%>"  + "<col width=70%>"
                + "<thead>"
                + "<tr>"
                + "<td  width=10%>" + "<font color=#000000 size=-2>" + "<b>" + "Class" + "</b>" + "</font>" + "</td>"
                + "<td  width=10%>" + "<font color=#000000 size=-2>" + "<b>" + "Recordkey"  + "</b>" + "</font>" + "</td>"
                + "<td   width=10%>" + "<font color=#000000 size=-2>" + "<b>" + "Status"  + "</b>" + "</font>" + "</td>"
                + "<td  width=70%>" + "<font color=#000000 size=-2>" + "<b>" + "Summary"  + "</b>" + "</font>" + "</td>"
                + "</thead>"
                + "</table>")
```

Continuing down the script the next section adds a line to the report by appending an <HR> tag and then prepares a section to get the related records information for the workorder.

It created a title (RELATED RECORDS), connects to the RELATEDWO data (This is a relationship on the work order object), then lays out a table to display the RELATEDWO data.

Automation Script continued…

The SCRIPT breakdown:

```
currMbo=relatedrecSet.moveFirst()

while currMbo is not None:
        val.append("<table border=0 width=1000>" + "<col width=10%>" + "<col width=10%>" + "<col width=10%>" + "<col width=70%>"
        + "<tr>"
        + "<td width=10%><font size=-2>" + relatedrecSet.getString("RELATEDRECWOCLASS") + "</font></td>"
        + "<td width=10%><font size=-2>" + relatedrecSet.getString("RELATEDRECWONUM") + "</font></td>"
        + "<td width=10%><font size=-2>" + relatedrecSet.getString("RELATEDRECWO.STATUS") + "</font></td>"
        + "<td width=70%><font size=-2>" + relatedrecSet.getString("RELATEDRECWO.DESCRIPTION") + "</font></td>"
        + "</font></td>"
        + "</tr>"
        + "</table>")
        currMbo=relatedrecSet.moveNext()
```

This next bit of code moves through all of the related records and for each one it outputs into an HTML table for display in the Summary Report.

Automation Script continued…

The SCRIPT breakdown:

```
#retrieve work log records and writes them in the buffer
worklogSet = mbo.getMboSet("MODIFYWORKLOG")

worklogSet.setOrderBy("CREATEDATE")
```

In the next section of the script, where the worklog entries are pulled into the summary, this demonstrated that you can use an orderby statement to sort that section in whichever order you require.

Automation Script continued…

The SCRIPT breakdown:

```
#  sets the formatted string in the attributes value
mbo.getMboValue("BPDSUMMARY").setValue(val.toString(), MboConstants.NOACCESSCHECK | MboConstants.NOVALIDATION)
```

Once all of the sections and data from related records that you require have been brought into the summary.

The final line of the script populates the non persistent summary field with the contents of the string field which has been built by the script.

Learn Basic HTML

When we first started building these summary objects and automation scripts, we just built the string. The result was a very flat, badly laid out summary.

Key areas to improve the look of the summary:

- Learn about HTML Tables
- Table Rows, Table Data
- Learn the various attributes of the font command
- Learn how to make a field bold

If you are going to display a Long Description in the summary report, our recommendation is that you convert it to plain text using the following command in your script.

```
HTML.toPlainText(currMbo.getString("DESCRIPTION_LONGDESCRIPTION")
```

The reason for this is that certain characters in a Long Description can cause the script to fail.

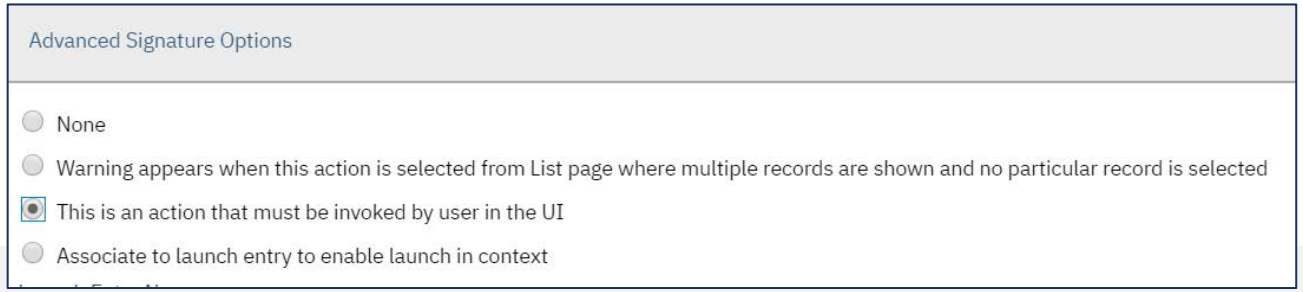
Configuring the Application

The next step in the process is to add a new TAB to the application where you want the summary to appear.

NOTE: AT THE CURRENT TIME DO NOT USE APPLICATION DESIGNER TO ADD A NEW TAB TO AN APPLICATION.

The reason for this is that doing so corrupts the application XML such that the contents of the new tab appear at the bottom of the list tab when you display the application.

Before you manually edit the XML and import it back into Maximo create a new signature option for the application with the same name as your automation script (and it is CASE Sensitive).



Configuring the Application

Here are the steps to manually add the tab to the application:

1. Open the application in the application designer.
2. Export the application XML and save the exported file to your desktop.
3. Edit the exported XML using Notepad ++ or your favorite text editor
4. Paste in the new tab information after the very last tab of the application and before he </tabgroup> tag
5. Save the changes and import the modified XML into the application designer in Maximo

```
  </tab>
▼<tab id="summary" label="WO Summary">
  ▼<section id="summary_s1">
      <pushbutton id="1520435426663" label="Generate Summary" mxevent="BPDWO001"/>
      <richtexteditor dataattribute="BPDSUMMARY" extra_plugins="[]" height="1000" hidelabel="true" id="summary_s1_wosummary" plugins="[]" width="900"/>
    </section>
  </tab>
</tabgroup>
```

Configuring the Application

A note about what you are pasting into the new tab. You are adding a section and at the top of the section will be a pushbutton to trigger the automation script.

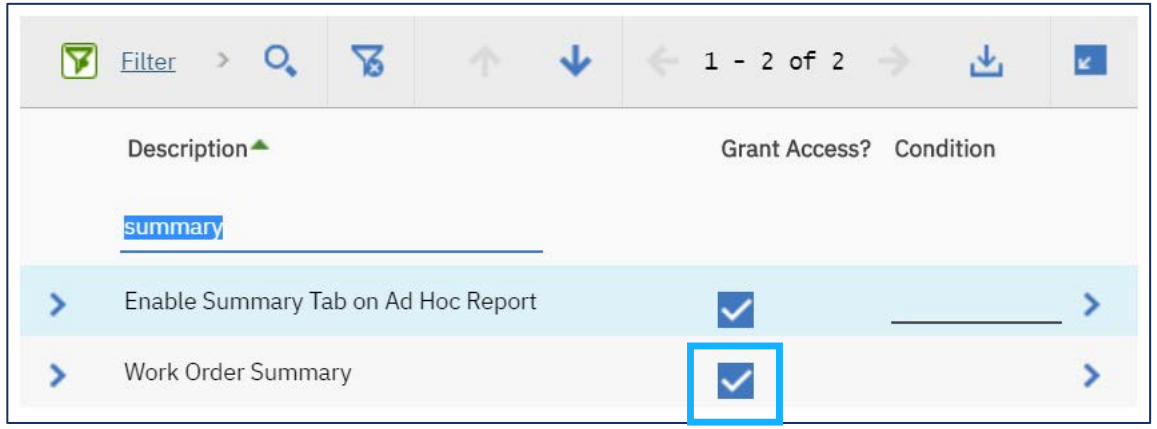
Under that button will be the field to display the summary attribute.

Once this is manually added through editing the XML, you can modify the size and position of things.

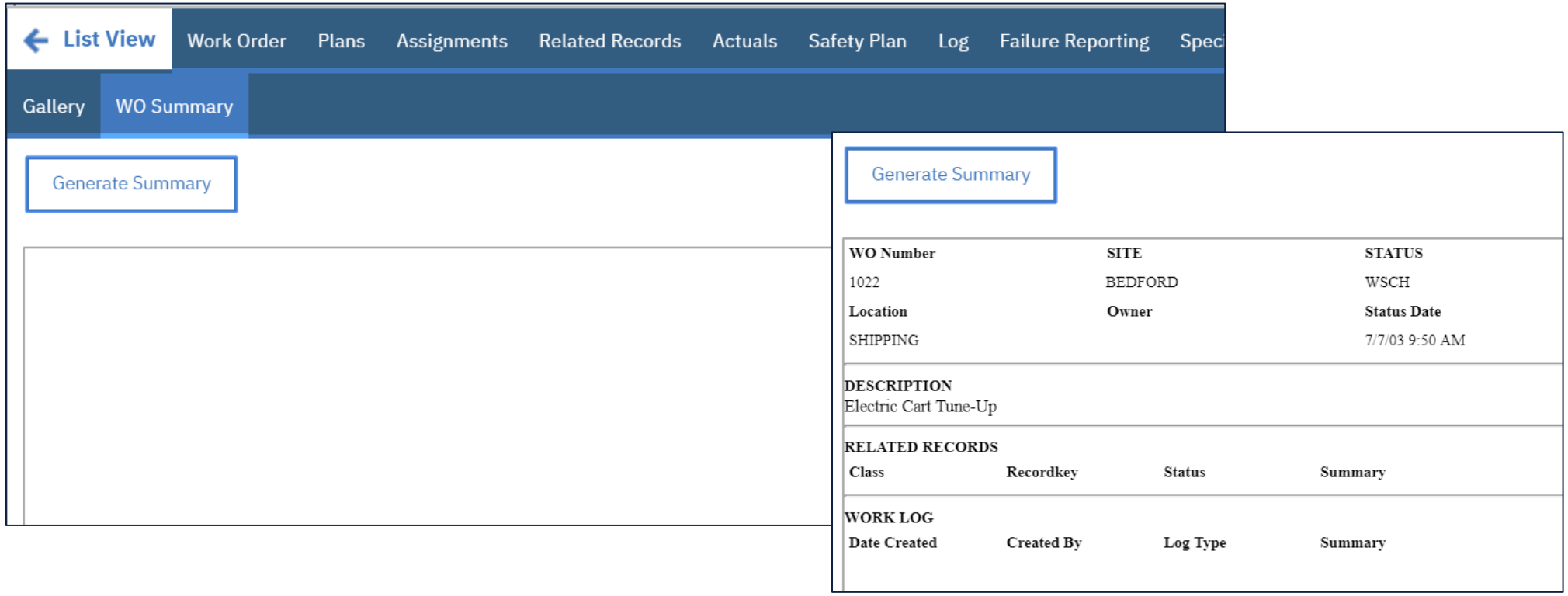
```
  </tab>
▼<tab id="summary" label="WO Summary">
  ▼<section id="summary_s1">
     <pushbutton id="1520435426663" label="Generate Summary" mxevent="BPDWO001"/>
     <richtexteditor dataattribute="BPDSUMMARY" extra_plugins="[]" height="1000" hidelabel="true" id="summary_s1_wosummary" plugins="[]" width="900"/>
   </section>
  </tab>
</tabgroup>
```

Security Settings

- To test the new push button go to the Security Groups application and bring up the security record for the MAXADMIN security group.
- Then go to the Work Order Management application and filter the options until you see Work Order Summary. Make sure this security option is checked on.
- Log out of Maximo, Log Back in and then go test the button.



Give it a Whirl!



Go to the Work Order Tracking application, open any work order, click on the WO Summary Tab. Then click the Generate Summary button. If it all works your summary should appear in the area below immediately.

If you get an error read what it says, and check the automation script for errors.

Adding to The Summary

Once the summary is working, you can always add more data to it, but going to the automation script and adding new tables, get new data from other relationships, or add additional fields to the tables you have already created.

Once it is working you can add comments to the automation script by placing a hash-tag # at the start of the line.

If you want specific information for the summary and a relationship does not exist, you can create the relationship.

In the database configuration tool for the workorder database table and then use that relationship to get the data for your summary. (example, get the name of the LEAD for the work order).

Other Potential Uses in Maximo

The example given is for a Work Order Summary, but you can build summaries for any application/database table you wish.

Examples:

- **Asset Summary** – including ownership information, asset move history, where used information.
- **SR Summary** - including worklogs, related records MOC Summary (for Maximo HSE or Oil and Gas) showing all approval records, action items, related records.
- **Inventory Summary** – showing vendor data, and transaction data.

What You Have Learned

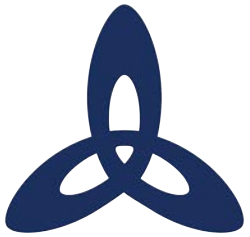
- Non-BIRT solution for various summaries
- Uses Automation Script
- Runs lightening fast
- Users Love it wherever it has been implemented

This concept can be applied to almost any application in Maximo.

THANK YOU!

Success! You should have learned some cool new reporting techniques in Maximo.

MAXTEACH



Ask us a technical question:

maxtech@bpdzenith.com

www.bpdzenith.com/maxtech

MaxTECH is the first ever dedicated Maximo Technical User Group aimed at Maximo Administrators, Developers and Technical Support staff.

It is a great place for users to ask and answer technical questions, learn from each other, collaborate and help improve Maximo in your organization.

MaxTECH was founded in 2017 by BPD Zenith and is chaired by Maximo Consultant Stephen Hume. We host several events every year (Calgary, St. Louis, MaximoWorld, Houston, MUWG, Northern California MUG, Maximo UK & Ireland User Group) including digital events.

MaxTEACH is a free online user group designed to go in depth into a Maximo topic.