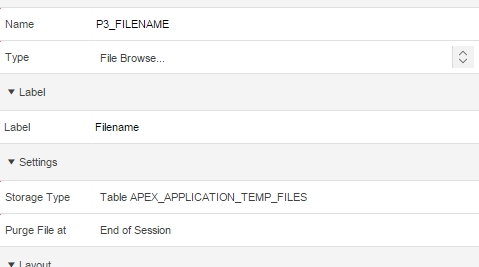
Uploading an image to an ORDImage object using APEX

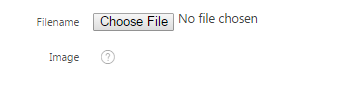
This is the third part of a series of 3 to be followed in order.

In **APEX**

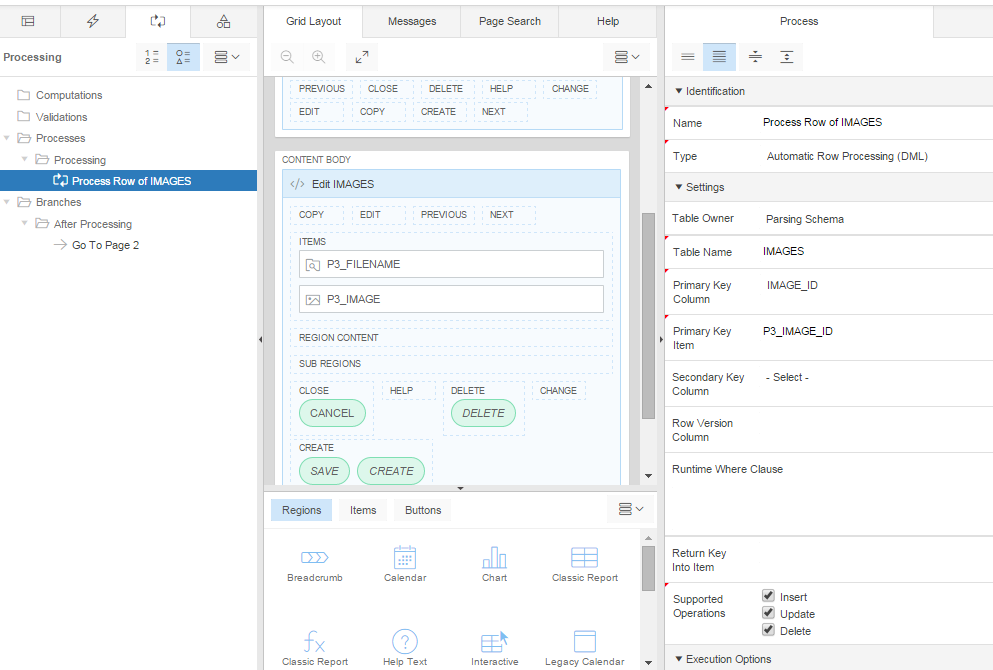
Edit Page 3 and change the Type of P3\_FILENAME to File Browse … and the Storage Type to Table\_APEX\_APPLICATION\_TEMP\_FILES



Run the Page and click on the Choose File button to browse for a file but you can’t do anything with it



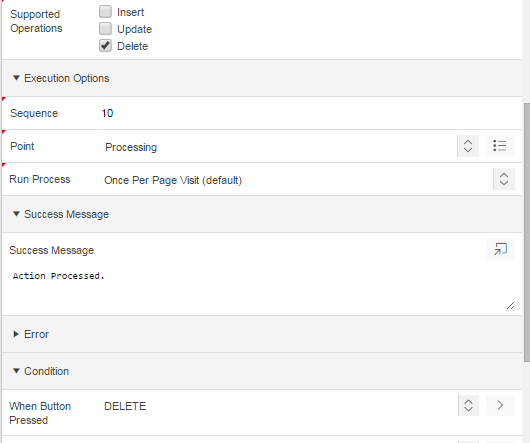
When the Form with Report pages were created an Automatic Row Processing (DML) process (Process Row of IMAGES) was created so that table rows could be INSERTed, UPDATEd and DELETEd.



Whilst APEX will happily work with BLOB data, ORDImage is more complicated …

We need to replace the default behaviour so that an image can be uploaded and saved in the table.

To do this, edit the Process Row of IMAGES process and uncheck INSERT and UPDATE and set the process to run when the DELETE button is pressed.



Save and Run

Having removed the ability to INSERT a new row, we need to create a custom process to replace it[[1]](#footnote-1).

Create a new process, name it Insert New Image, change the Type to PL/SQL Code and enter as the Source

DECLARE

l\_upload\_size INTEGER;

l\_upload\_blob BLOB;

l\_image\_id NUMBER;

l\_image ORDSYS.ORDImage;

BEGIN

--

-- Get the BLOB of the new image from the APEX\_APPLICATION\_TEMP\_FILES (synonym for WWV\_FLOW\_TEMP\_FILES)

-- APEX 5.0 change from APEX\_APPLICATION\_FILES which has been deprecated

-- APEX\_APPLICATION\_TEMP\_FILES has fewer columns and is missing doc\_size

--

SELECT

blob\_content

INTO

l\_upload\_blob

FROM

apex\_application\_temp\_files

WHERE

name = :P3\_filename;

--

-- Insert a new row into the table, initialising the image and

-- returning the newly allocated image\_id for later use

--

INSERT

INTO

images

(

image\_id,

filename,

image

)

VALUES

(

seq\_image\_id.nextval,

:P3\_filename,

ORDSYS.ORDImage()

)

RETURNING

image\_id, image

INTO

l\_image\_id, l\_image;

-- find the size of BLOB (get doc\_size)

l\_upload\_size := dbms\_lob.getlength(l\_upload\_blob);

-- copy the blob into the ORDImage BLOB container

DBMS\_LOB.COPY( l\_image.SOURCE.localData, l\_upload\_blob, l\_upload\_size );

-- set the image properties

l\_image.setProperties();

UPDATE

images

SET

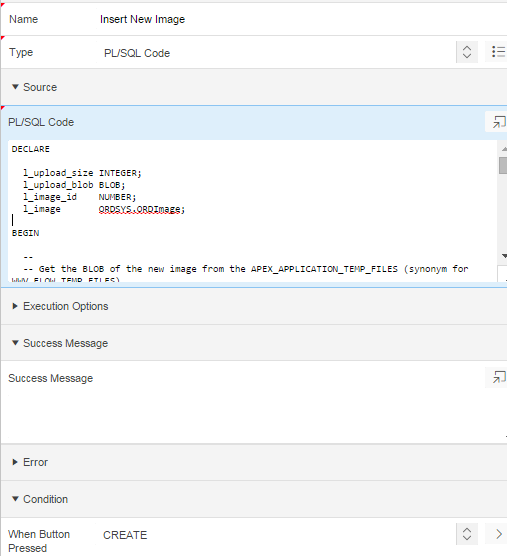
image = l\_image -- original ORDImage image

WHERE

image\_id = l\_image\_id;

END;

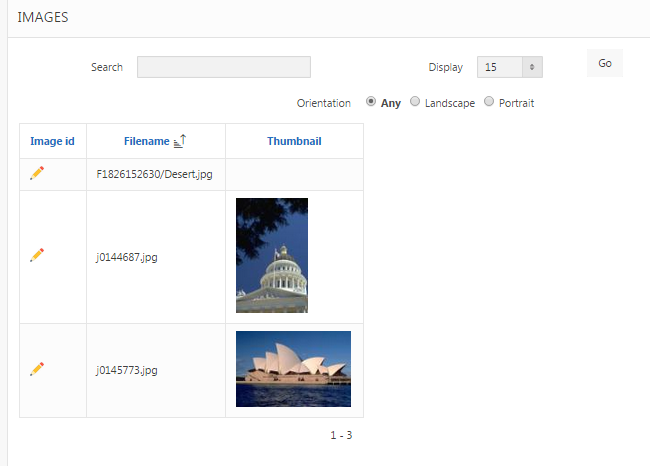
And set the When Button Pressed condition to CREATE



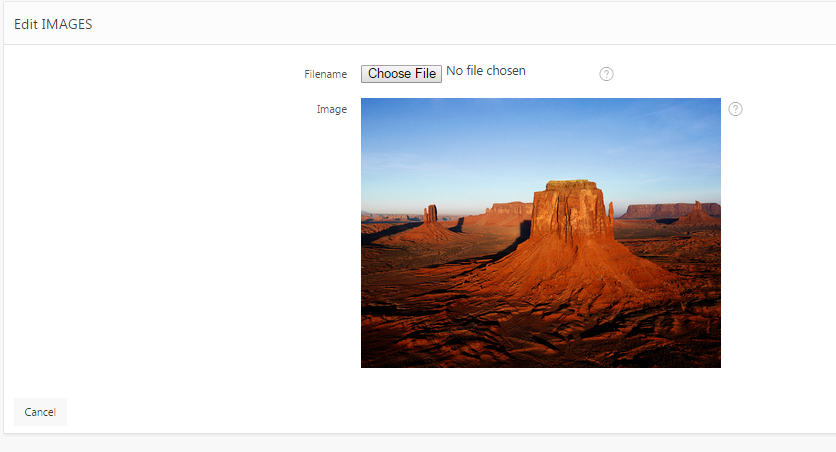
Save and Run, browse to an image file and click Create

Note that the thumbnail is not shown. Why?

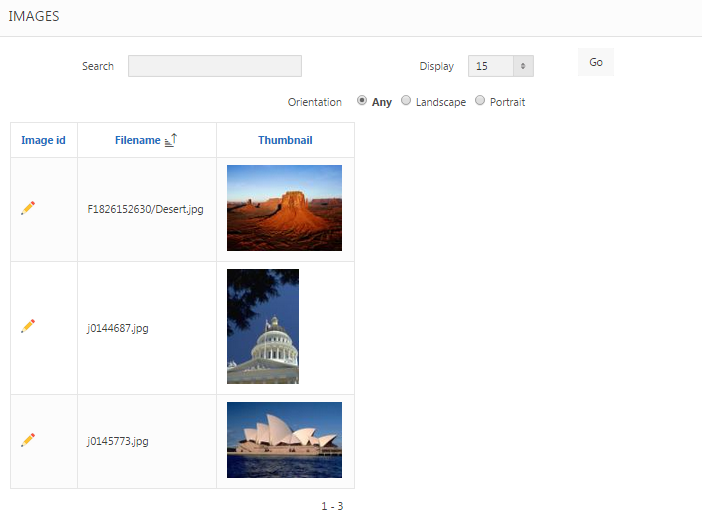
Note also the format of the filename in the report. The prefix is used to uniquely identify the file.



Clicking on the Edit symbol (the pencil) shows the full size image on Page 3.



**So, what about the missing thumbnail? This is what you want eventually**



**Over to you…**

1. One for UPDATE will also be needed at some point [↑](#footnote-ref-1)