User Uploaded Images in PeopleTools

PT 8.58
Randall Groncki

Introduction

PeopleTools has the power to deliver an image rich, modern UI for your enterprise. Whether branding, employee photos or the millions of images for the catalog's widgets, we can deliver the modern UI you need within the PeopleSoft architecture.

There are two primary image categories in PeopleSoft: Design Time images and those images uploaded by users. This document will specifically cover the User Uploaded images.

User Images are those images uploaded by the user while using the system. For example, employee photos and asset images. The PeopleSoft developer creates tables to store and pages to display these images but has no idea of what the actual images are at design time.

PeopleTools supports users uploading their own image to the application. These images can be thought of more as displayable attachments than design elements. These images are data and related to the data in the system.

Design time images are those uploaded into the App Designer when creating and modifying pages. These images are managed objects and migratable. Design Time images are discussed in our paper "Using Design Time Images in PeopleTools" and out of scope of this document.

Though extensively used by PeopleSoft, the image handling objects and documentation functionally haven't been updated since the original version 8 tools debuted in 2000.

Using images in fluid pages will be addressed it its own document and not in the scope of this document.

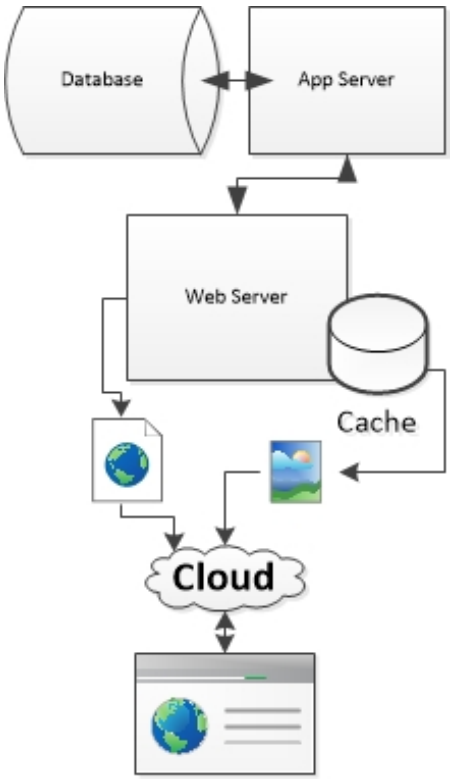
How PeopleTools delivers images to the browser

A primary challenge with images is that the application does not provide the image to the user's browser in the HTML as it can with most other elements such as text, JavaScript and CSS. Rather the application places the image in a known location of the web server and provides the browser a link to that location.

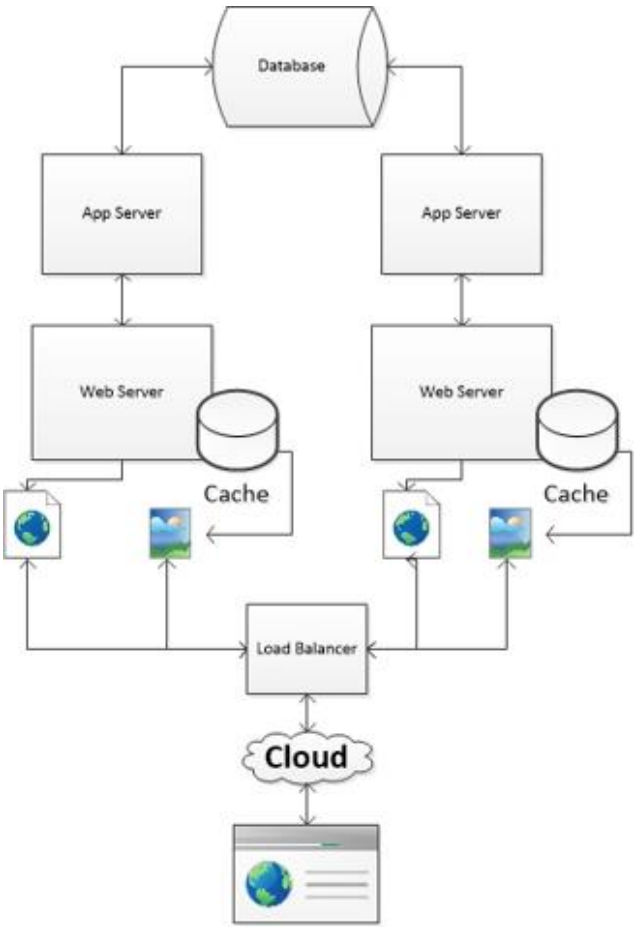
If the architecture contains multiple Web and App servers behind a load balancer, we can't know ahead of time where that image is going to be for the user… and the next user after that.

Browser gets Images and objects from the Web Server's Cache

Simple Network – Single Web Server



Complex Network - Multiple Web Servers



When an image is called for at runtime, the App Server places the designated image in the web server's cache and then provides a link to that image in the HTML delivered to the user's browser.

As developers, we never need to pre-stage images on web or file servers. This is bad practice since it would create a versioning and ongoing maintenance tasks. Instead, the architecture handles image management for us.

The Image Field

Image fields are defined by the type of image they contain. When creating a new image or reusing a previous image, consider the file type of the image.

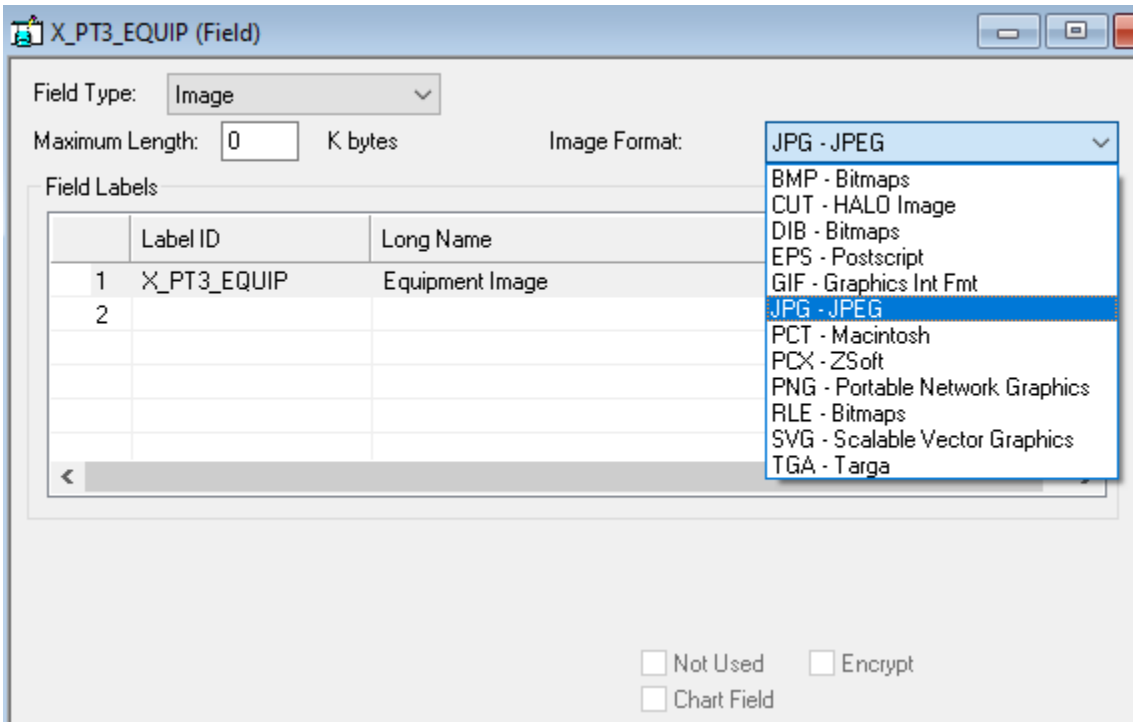


Image Field Attributes

Maximum Length

Leave this field "0" to allow images to stored as per the database maximum. On the Oracle Platform, Peopletools does not limit the file size to the value specified here.

Database	Max Size Stored if field left zero
Oracle SQL Server	2 Gigabytes
DB2	32,700 KB. This depends on how the database is configured

Image upload size is better controlled through the InsertImage() function (covered later in this doc)

Image Format

As shown in the example above, the image field can accommodate 12 different image types. This will restrict the user to that defined type of image they can upload to this field.

Currently, the most common type of user image uploaded is a JPEG.

The Image Reference Field

Use the Image Reference Field to display static, design time images that are easily controllable by PeopleCode. These field are usually attached to a Derived/Work record used on the page buffer.

An example of an image reference field use would be to display a default image for a data element where the user has not yet uploaded an image. We would like a consistent UI without repeatedly storing a common, default image for every data element without an image. Use this field to display or hide the default image depending on if the user image is populated.

Image field displayed
User uploaded image

Reference image displayed
No user uploaded image

User Image Record Definition

The record definition for the table to contain the user images are like any other PeopleTools record definition.

Along with your image field, you must also include the PSIMAGEVER field on the record definition (see image below). PeopleTools will auto-populate this field when a new image is uploaded through the UI.

This field must be placed on the page in the same scroll level as the image. Usually, this field is defined as invisible to not distract the users. But it must be there.

In the example below

- "X_PT3_EQUP" is the image field that will hold the image file in the database.
- "PSIMAGEVER" is our required image version field.

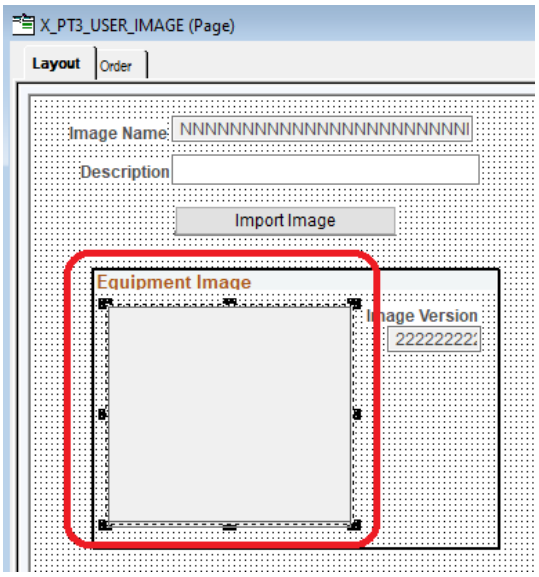
X_PT3_USR_IMAGE (Record)

Record Fields | Record Type

Num	Field Name	Type	Len	Format	Short Name	Long Name
1	IMAGE_NAME	Char	30	Upper	Image Name	Image Name
2	DESCR	Char	30	Mixed	Descr	Description
3	X_PT3_EQUIP	Img	0K	JPG	Equipment	Equipment Image
4	PSIMAGEVER	Nbr	10	Raw B	Image Version	Image Version

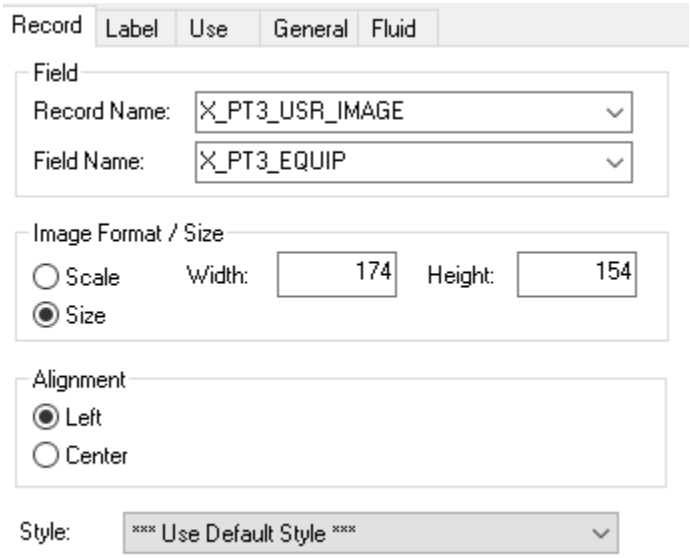
Placing the User Image on a page

Add an image field to the page at the correct scroll level. This can be done by dragging the image field from the record definition in the Project Object pane onto the page or by clicking on the image icon on the toolbar 



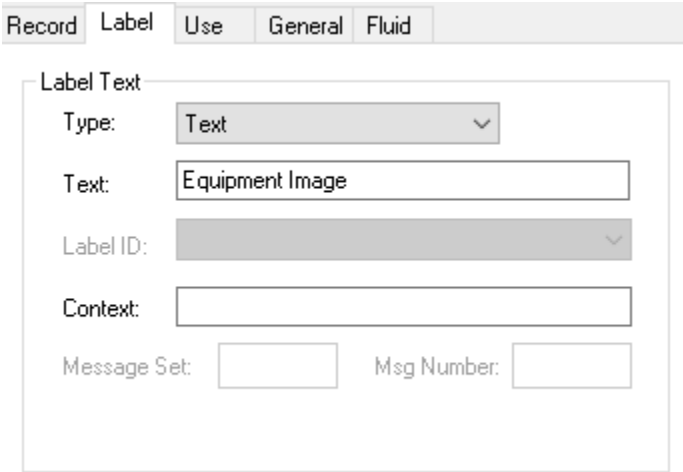
Open up the properties of the page image object and adjust.

Record tab: choose between Scale and Size:

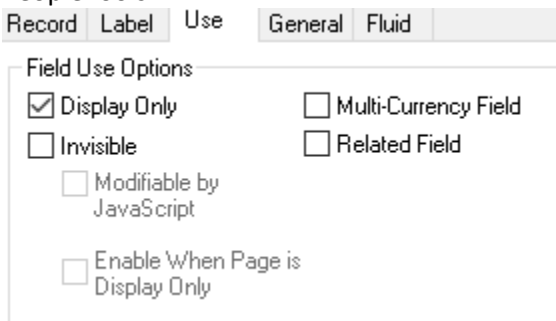


- **Scale**: display the image to the width and height specified here. **Warning: PeopleTools will force this image into the width/height size. If this ration does not match the original image* ratio, *the image will be visually skewed*
- **Size**: display the image at the native width/height

Label tab: The label text field becomes the mouse-over hover text content.

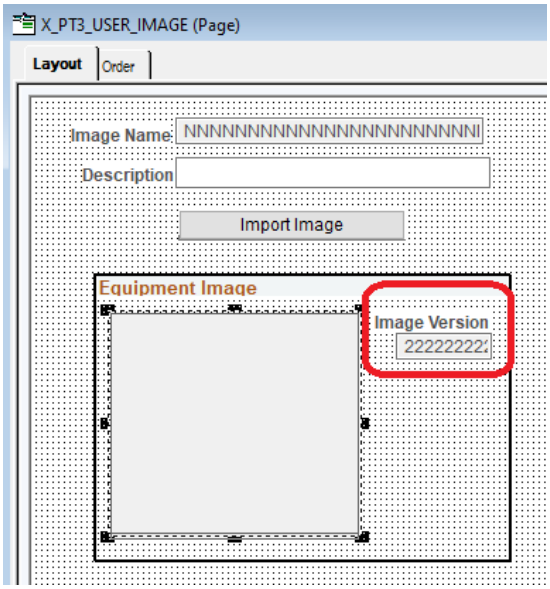


Use tab: The image should be display only since the images are not directly manipulatable by the user in PeopleTools.



PSIMAGEVER field

Add the PSIMAGEVER field from that same record definition on the page at the same scroll level as the image. The PSIMAGEVER field can be hidden through the field's "Use" properties tab, but it must be there at that scroll level. Image functions ***MAY*** work correctly if the image is not in the page but present in the page buffer if all the records keys are in the buffer. Better coding is just to put it on the page and mark it as invisible rather than assuming it will be in the buffer at runtime.



PeopleTools Functions for Manipulating User Images

PeopleCode provides several functions for uploading and manipulating images within the PeopleSoft Application Environment. These functions leverage Java and JavaScript behind the scenes to provide a base level image functionality to the user.

Both InsertImage() and CropImage() functions are "Think Time", meaning they are expecting user input. They can't be used in a batch process for example to import multiple employee photos. They are designed for real time interaction with the user through a PeopleTools Component and Page.

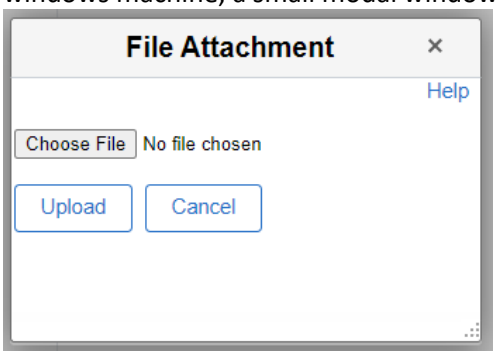
Be warned: the image functions have not been functionally updated for many PeopleTools versions. For example, they still use the scroll path notation for image location rather than the current row/Rowset/record/field notation. Some do not have all the original functionality.

As a note, I have found applications are more stable and reliable when all image manipulation work is done at the Buffer Level 0 level

Adding an image

We enable a user to upload an image into PeopleTools via the "**InsertImage()**" command.

This function prompts the user to choose an image through the browser's host OS file functions. On a windows machine, a small modal window will prompt the user to choose a file.



Clicking the "Choose File" button invokes a Windows File Explorer window enabling the user to navigation and choose an image to upload. I assume this works the same way leveraging the OS Services in other OS's such as Macs and Linux too.

```
InsertImage([scrollpath, target_row,] [recordname.]fieldname [, max_size])
```

The function returns an integer as a status.

Insert Image Code Example

```
Local Record &Rec_X_PT3_IMG_WRK = GetLevel0()(1).GetRecord(Record.X_PT3_IMG_WRK);
Local Record &Rec_X_PT3_USR_IMAGE =
GetLevel0()(1).GetRecord(Record.X_PT3_USR_IMAGE);

Local integer &Int_Image_Insert;

&Int_Image_Insert = InsertImage(X_PT3_USR_IMAGE.X_PT3_EQUIP);

Evaluate &Int_Image_Insert
When = %InsertImage_Success
   &Rec_X_PT3_USR_IMAGE.X_PT3_EQUIP.Visible = True;
   &Rec_X_PT3_IMG_WRK.HR_DUMMY_IMAGE.Visible = False;
   /* success */
   Break;
When = %InsertImage_Cancelled
   MessageBox(0, "", 0, 0, "User Canceled Image Upload");
   Break;
When = %InsertImage_ExceedsMaxSize
   MessageBox(0, "", 0, 0, "Image Exceeded Max Size");
   Break;
When-Other
   MessageBox(0, "", 0, 0, "Virus Scan Issue");
End-Evaluate;
```

The only required parameter of the function is the fieldname from the database table to store the retrieved image.

Since this is a function where the user can experience an error or even choose not to import an image, we need to check the status of the completed function and react appropriately. The responses are either a success (0) or an image was not uploaded (return > 0). See PeopleBooks for a complete list of return values.

The InsertImage() function is also the best place to limit the size of the user uploaded image. The final parameter of the function limits the upload file size in Kilobytes. For example, the below function would return the "`%InsertImage_ExceedsMaxSize`" error if the user uploads an image larger than 20K.

```
&Int_Image_Insert = InsertImage(X_PT3_USR_IMAGE.X_PT3_EQUIP, 20);
```

Deleting an image

The DeleteImage() function deletes the image from a database field.

```
DeleteImage(scrollpath, target_row, [recordname.]fieldname)
```

The function returns a Boolean

- True: Image successfully deleted
- False: Image not deleted

```
Local Record &Rec_X_PT3_IMG_WRK = GetLevel0()(1).GetRecord(Record.X_PT3_IMG_WRK);
Local Record &Rec_X_PT3_USR_IMAGE =
GetLevel0()(1).GetRecord(Record.X_PT3_USR_IMAGE);

Local boolean &Bl_Image_Delete;

&Bl_Image_Delete = DeleteImage(X_PT3_USR_IMAGE.X_PT3_EQUIP);

If &Bl_Image_Delete Then
   /* success */
   &Rec_X_PT3_USR_IMAGE.PSIMAGEVER.Value = 0;
   MessageBox(0, "", 0, 0, "Image Deleted");
Else
   /* fail */
   MessageBox(0, "", 0, 0, "Error Deleting Imiage");
End-If;
```

The code above deletes the image contained in the `X_PT3_USR_IMAGE.X_PT3_EQUIP` field.

Cropping

*Note: PeopleTools 8.58 currently has a bug causing the CropImage() function generating an "OnReadyState" Error. (Doc ID 2769073.1, Bug: 32773424). Oracle solution is a fix in PeopleTools 8.59 or 8.60.

The CropImage() function enables a user to Crop an existing image producing a new image. This function does not destroy the original image.

```
CropImage(src_recfield, dst_recfield, [width, height])
```

The function returns…

- PeopleTools prior to 8.58: Boolean
- PeopleTools 8.58: Integer?

This function is better done at Buffer Level Zero (0) rather than somewhere in the child scrolls for simplicity and reliability.

Optional Parameters

The CropImage() function has two optional parameters: Width & Height. However, this description is misleading. The width and height parameters define the **ASPECT RATTIO** of the new image, not the size. The user's selection box will be locked to this aspect ratio. The size and portion of the image is still selected by the user, just locked into that defined aspect ratio.

If these parameters are not specified, the user will decide the aspect ration of the new cropped image by their selection.

CropImage() Code example

```
Local Record &Rec_X_PT3_IMG_WRK = GetLevel0()(1).GetRecord(Record.X_PT3_IMG_WRK);
Local Record &Rec_X_PT3_USR_IMAGE =
GetLevel0()(1).GetRecord(Record.X_PT3_USR_IMAGE);

Local boolean &Bl_Image_Crop;
Local integer &Int_Crop_Result;

Local integer &Int_Width = 3;
Local integer &Int_Heitht = 2;

&Int_Crop_Result = CropImage(X_PT3_USR_IMAGE.X_PT3_EQUIP,
X_PT3_IMG_WRK.X_PT3_EQUIP_TMP, &Int_Width, &Int_Heitht);


If None(&Int_Crop_Result) Then
   &Rec_X_PT3_IMG_WRK.GROUPBOX1.Visible = True;
Else
   MessageBox(0, "", 0, 0, "Error Cropping Image: %1", &Int_Crop_Result);
End-If;
```

This example allows the user to crop an image to using a fixed, 3:2 aspect ratio. The resulting image is placed in a work record's image field. The PSIMAGEVER field is also on the same work record as the receiving image and placed on the page next to the image.

A groupbox containing the new image and PSIMAGEVER field is unhidden showing the new, resulting image if the cropping is successful.

Resizing

The ResizeImage() function creates a new, different size image from the original image. This function is limited to JPG & BMP files. It will not resize a PNG.

As with the CropImage() function, the original image is not modified or deleted.

```
ResizeImage(URL.source_URL, URL.dest_URL, array_of_sizes [, type][, aspect_ratio])
```

Parameters

Source and Destination	The "Source" and "Destination" parameters are URLs instead of field references. These can either be a direct field reference in the UI or a file reference on the server. I have not tested the server file references. When using the field references, they must exist as direct references to fields in the buffer rather than object variable references.
Array of sizes	Depending on the TYPE parameter, the array will hold the new images dimensions or change percentage. Originally, this function could take an array of multiple dimensions and create multiple new images in the PS_PT_IMG_TMPSTORE table. I have not been able to get this multiple resize functionality working.
Type	%Resize_ByDimensions (0) or %Resize_ByPercentage (1). Use %Resize_ByDimensions (0) to force the new image into the required dimensions regardless of the original image size. Since PeopleTools does not give us an easy way to determine the parameters of the original image, resizing by percentage is not a reliable option.
Aspect Ratio	This Boolean parameter is only for %Resize_ByDimensions choice. If you leave one of the array size parameters as zero (0), then the new image will resize in the correct aspect ratio if this value is true. If you provide non-zero values in both height and width, this parameter will be ignored. However, your new image may be visually skewed.

ResizeImage() Code Example

```
Local integer &Int_ResizeImage;

Local array of number &Ar_ImgDimensions;
&Ar_ImgDimensions = CreateArray(0);

&Ar_ImgDimensions.Push(200); /* wdith */
&Ar_ImgDimensions.Push(0); /* height */

/*  %Resize_ByDimensions or  %Resize_ByPercentage , maintain aspect ratio */
&Int_ResizeImage = ResizeImage(X_PT3_USR_IMAGE.X_PT3_EQUIP,
X_PT3_IMG_WRK.X_PT3_EQUIP_TMP, &Ar_ImgDimensions, %Resize_ByDimensions, True);


If &Int_ResizeImage = 0 Then
   &Rec_X_PT3_IMG_WRK.GROUPBOX1.Visible = True;
Else
   MessageBox(0, "", 0, 0, "Error Resizing Image: %1", &Int_ResizeImage);
End-If;
```

This example resizes the source image into a page work field using the %Resize_ByDimensions parameter. The Aspect Ratio of the original image is maintained in the copy as the new image is 200 pixels wide and the height is set proportionately due to the zero (0) parameter.

Moving images

Moving images is a simple process of moving field values just like any other field types.

When moving or copying an image into another image field, also copy the PSIMAGEVER field.

```
/* Move resized image into the source image field (overwrite source image) */

&Rec_X_PT3_USR_IMAGE.X_PT3_EQUIP.Value = &Rec_X_PT3_IMG_WRK.X_PT3_EQUIP_TMP.Value;
&Rec_X_PT3_USR_IMAGE.PSIMAGEVER.Value = &Rec_X_PT3_IMG_WRK.PSIMAGEVER.Value;
```

Image Properties

PeopleTools does not have a facility to determine the properties of a user uploaded image other than type and size limits when uploading.

Java does have extensive file and image functions usable through PeopleTools Java objects.