



Document Imaging System Administration Guide

Table of Contents

Using this manual	5
Designates a section	5
Designates a sub-section	5
How to Begin	6
Create a document inventory list	6
Conversion Approach	8
ES Imaging Anatomy and Features	9
Anatomy	9
Multiple Item Selection	10
Shortcuts.....	10
Folder Configuration.....	11
Preparation for Folder Configuration.....	11
Folder Structure	11
Preparation for Capture and Index Configuration	13
Static and Dynamic Folders	14
Common Folder Needs	15
Search Needs	16
Folder Naming Scheme	17
Indexing Folder Destination	17
How Indexes Impact Indexing	22
Indexing Behavior Settings.....	25
Scanned Documents “Inbox”	25
Capture and Index Users Groups	27
Establishing Folder Structure	28
Creation of the System Root	28
Creation of Folder Types	28
Creation of Folders	30
Capture and Index Configuration	31
Creation of Keyword Types	31
Creation of Dynamic Names	32
Updating Folder Type with Index Settings	32
Creation of Capture and OCR Queues.....	33

Administration Guide

Updating a Capture Queue Name	33
Creation of a Folder Index.....	34
Creation of a Document Index	35
Folder History	36
Enabling Folder History	36
Disabling Folder History	37
Viewing Folder History.....	37
File Encryption	38
File Versioning	38
Security Administration	39
Preparation for Security Administration	39
Functions	39
Types of Permission Settings	40
Group Permissions.....	41
User Permissions.....	41
Permission Levels.....	41
System Default Permissions.....	42
Object Type Permissions.....	42
Object Permissions	42
Permission Hierarchy	43
Permission Hierarchy when User in Multiple Groups	43
Preset Group Permissions.....	44
Establishing User Accounts	45
Establishing Security	46
Setting System Default Permissions.....	46
Creation of Groups.....	47
Adding Users to Groups	47
Setting Object Type Permissions.....	48
Setting Object Permissions	49
User Interface	50
Preparation for User Interface Configuration	50
User Interface Overview	50

Administration Guide

User Interface Configuration	53
Setting User Interface Permissions	53
Workflow, Events, and Triggers	54
Workflow	54
Workflow Queue.....	55
Creation of a Workflow Queue	56
Workflow Item	57
Creation of a Workflow Item (Manually)	58
Creation of a Workflow Item (With Events)	59
Events and Triggers.....	59
Event Types.....	60
Create an Event.....	83
Triggers	83
Action Types	84
Schedule Types	86
Transaction History	87
Trigger Conditions.....	87
Trigger Sequence	88
Creating an Action Trigger (Events Perspective)	89
Creating an Action Trigger (Action Triggers Perspective).....	91
Creating a Schedule	92
Creating a Schedule Trigger (Schedules Perspective)	93
Creating a Schedule Trigger (Events Perspective)	94
Dynamic Tags.....	95
Universal Tags.....	96
Object Context Tags.....	98
System Deployment.....	105
Executable and AIR Application Runtime Parameters	105

Using this manual

This guide contains both the preparation steps and technical guidance needed to configure and customize ES Imaging. Before proceeding with the configuration, please be certain to review the preparation sections outlined within this document. These sections should be followed sequentially, since there are many areas that are dependent upon others. The preparation sections will reduce the time needed to setup the folder/file structure and security permissions and will eliminate rework later.

The color scheme and technique used throughout this manual are described below:

Designates a section

Designates a sub-section

***ES Imaging
Terms***

Used for terms and notes which are specific to ES Imaging.

***Quick Tips and
Shortcuts***

Used for alternate methods to perform the designated function.

How to Begin

Before delving into setting up the folder/file structure of ES Imaging, it is important to understand the *business problems of your organization* that prompted the decision to purchase imaging software. If the business problem is “unable to locate files in an efficient manner”, then identify and document the specific shortcomings, so they can be addressed. If the business problem is “inability to access files from offsite location”, then be certain to understand those needs, so consideration can be given in this area.

Create a document inventory list

Identify the scope of the initial effort to convert paper into electronic documents. Be certain to know what is included and what is not. Consider starting with one department at a time, to enable a smooth, more manageable transition period.

Once the in-scope documents are identified, elaborate further and document the following for each type of document. See the “*Preparation for Folder Configuration*” and “*Preparation for Capture and Index Configuration*” sections for additional guidance.

- **How created?**
Is the document created at your organization or is it received from an external organization?
- **How labeled?**
How is the document named for filing purposes? Is it named with client/customer name or account number?
- **How filed?**
Are the documents filed by year, category and client, or is some other filing system used? Is this current filing structure sufficient?
- **How to categorize?**
What type of document is it? This may result in multiple levels of categories, such as Insurance, Homeowner’s Insurance.
- **How retrieved?**
How are the current documents searched? What information is available when looking for a document? Are there situations that a document is not found, simply because the filing method does not provide for this need?
- **Who can access?**
Are there limited people that can view the document? Who is the document routed to for approval and are notations necessary at each step? What role will each person require? Global viewing or limited

Scan/Import

Dynamic Name

**Folder
Structure**

**Index/
Keywords**

**Security
(Permissions)**

viewing (only designated folders)? Administrative functions, indexing or capturing?

- ***What is the rough count of paper documents?***

For each type of document, what is the volume of paper received/sent?
What is the volume of the historical and current filed documents?

***Conversion
Approach***

Conversion Approach

Once the above analysis is complete, decisions can be made as to whether to convert all old paper files into electronic documents. There are three basic approaches to choose when making this decision.

- First, a *scan forward* approach allows for a quicker startup, and is suitable when historical files are not routinely accessed. If this approach is chosen, an effort to convert these at a later time might also be appropriate.
- Second, a *hybrid* approach of only converting specific document types (such as invoices) based on the business needs of the organization. This is a good approach, when some of the historical files may be critical to the organization, while other files are not.
- Third, a *full conversion* of all historical files may be the best approach for the organization. A decision will need to be made as to whether to convert the files prior to implementing ES Imaging, or to have this process happening concurrently.

If the *hybrid* or *full conversion* approach is selected, there are many companies available that offer file scanning services to back scan paper documents. This service assists with back scanning so the organization can continue their routine business and adjust to the imaging system more easily.

For documents that already exist in electronic form, the *import* and *print routing features* of ES Imaging makes this task simple. See the ES Imaging User Manual for detailed instructions.

ES Imaging Anatomy and Features

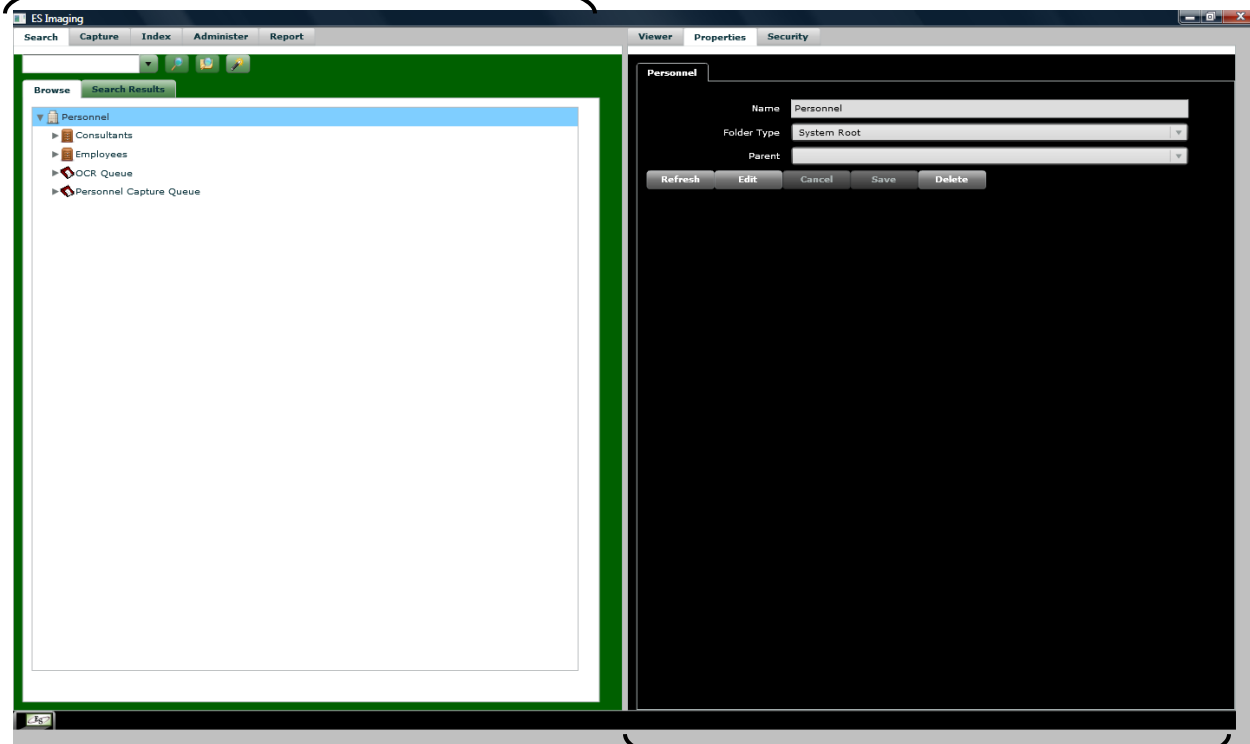
Anatomy

A general understanding of the structure of ES Imaging is necessary before beginning administrative tasks. ES imaging uses a familiar tree structure and dual panels to provide an intuitive user interface and experience. Tabs and buttons are used to easily navigate from page to page. User interface is controlled with a combination of permissions (see “*Preparation for Security Administration*” section) and user interface settings (see “*Preparation for User Interface Configuration*” section).

Quick Tip – see the “*Preparation for User Interface Configuration*” section for an explanation of each tab.

Left Panel

- Search/Browse
- Copy/Cut/Paste
- Import/Export

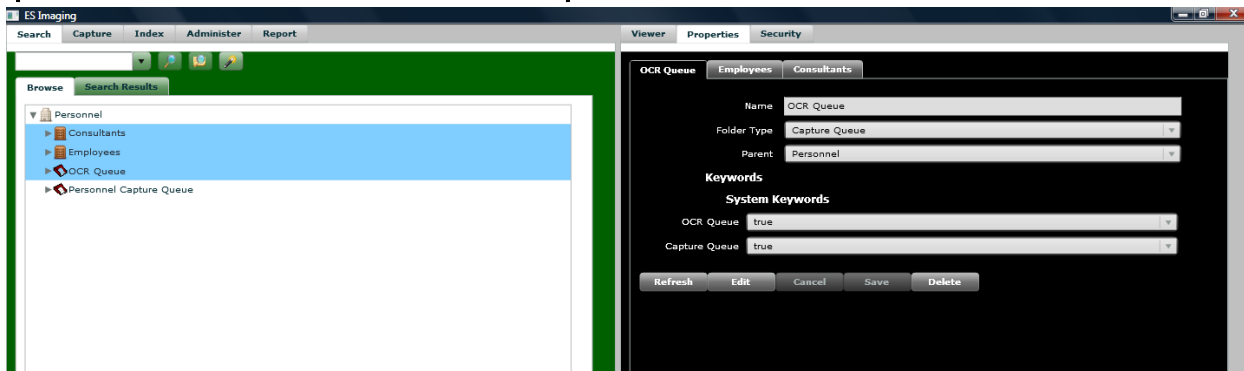


Right Panel

- View/Markup/Print
- View/Edit Properties
- View/Edit Security

Multiple Item Selection

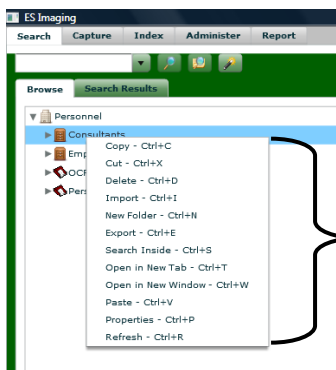
- To select consecutive items: Hold the Shift key and click on required items.
- To select non-consecutive items: Hold the Ctrl key and click on required items.



When selecting multiple items, the right portion of the screen will contain multiple tabs which are named with the selected items. This allows toggling back and forth from item to item. If changes are made to one of the items, a message is displayed prompting for saving or cancelling the changes.

Shortcuts

Several functions have built-in shortcuts to eliminate extra keystrokes or provide for the inability to right click (right click is unsupported within both a web browser and for one button mice).



Copy – **Ctrl + C**
Cut – **Ctrl + X**
Delete – **Ctrl + D**
Import – **Ctrl + I**
New Folder – **Ctrl + N**
Export – **Ctrl + E**
Search Inside – **Ctrl + S**
Open in New Tab – **Ctrl + T**
Open in New Window – **Ctrl + W**
Paste – **Ctrl + V**
Properties – **Ctrl + P**
Refresh – **Ctrl + R**

Folder Configuration

Preparation for Folder Configuration

Once ES Imaging is installed, there will be a temptation to rush into the configuration so that the organization can benefit from the use of electronic documents. However, the more completely the needs of the organization are understood the more seamless the transition.

One of the first considerations is the *structure of folders/files* and the *searching needs* of the organization. A clear and thorough knowledge of the organizational needs will make subsequent steps much easier for the administrator (see the “*How to Begin*” section).

Folder Structure

In the “*How to Begin*” section, the creation of a document inventory list was suggested. This list is essential in the creation of the folder structure within ES Imaging. From the steps where documents were categorized and the filing systems were defined, the hierarchy and relationship of folders begin to take shape.

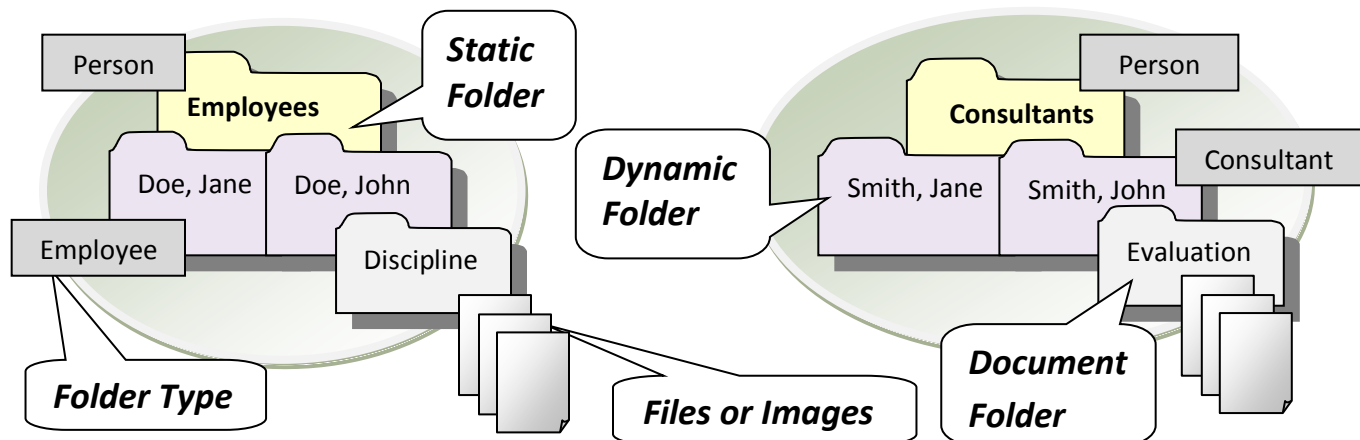
Folders

Identify and document the structure of the categories and the sub-categories until all security and search needs have been considered. If any of the categories or sub-categories have conflicting security or search needs within themselves, this would imply that another category/sub-category should be created.

Folders that are created by the administrator are referred to as *static folders* and those created during indexing are called *dynamic folders*. When there are several static or dynamic folders with identical search and index needs, these folders should be set up with the same *folder type*.

Folder Types

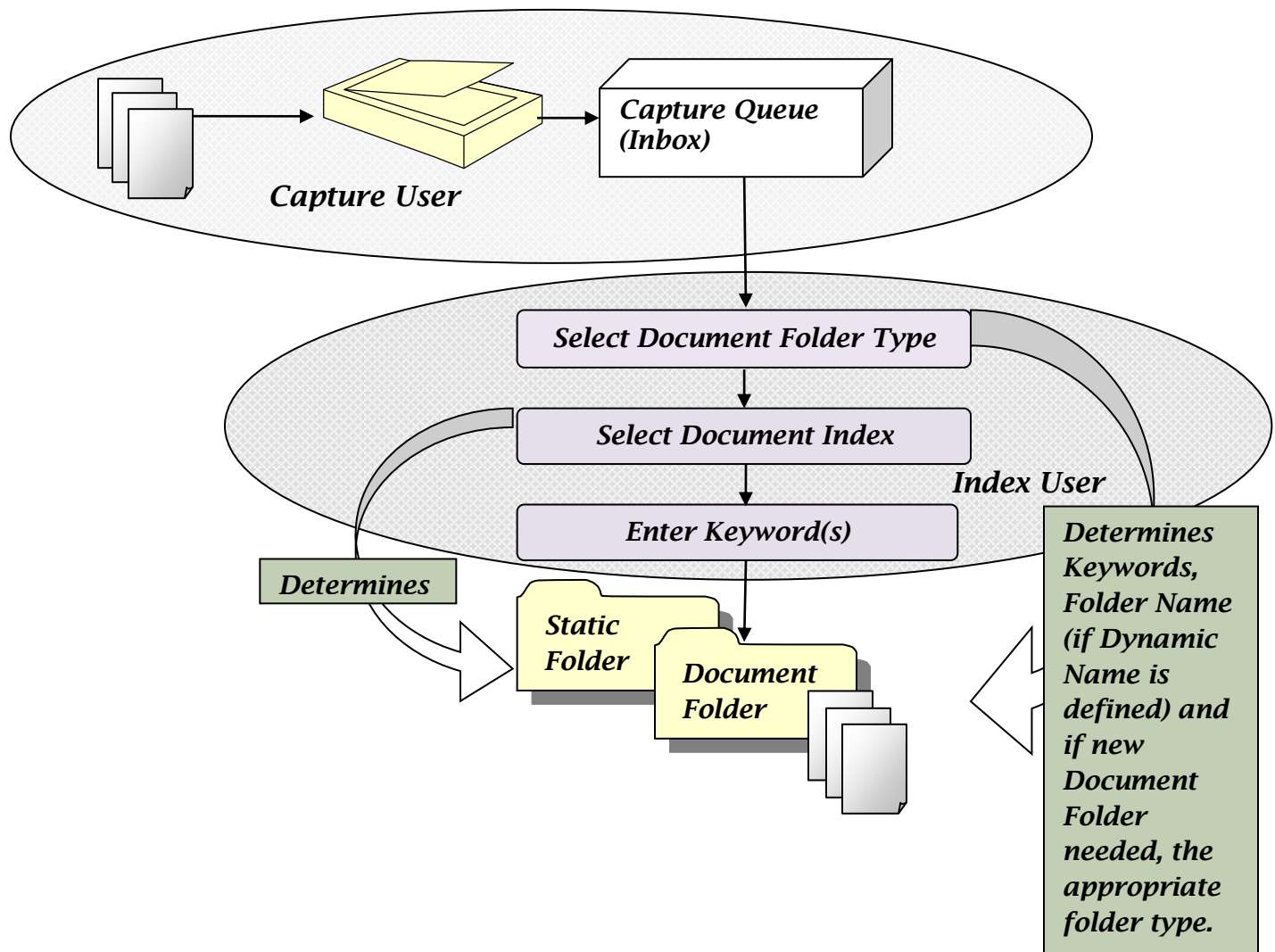
As shown below, the lowest folder level is called a *document folder* and is referred to as such within ES Imaging. Pages that are placed into the *document folder* are called *files* or *images*.



Preparation for Capture and Index Configuration

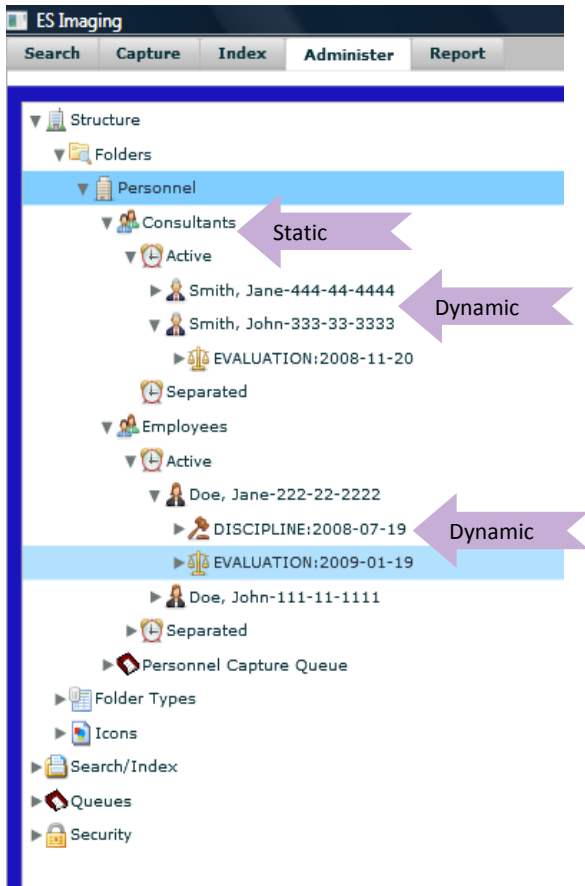
The *capture* and *index* functions within any imaging system are vital to the effectiveness of locating and retrieving folders/files. While the administrators are responsible for setting up the folder/file structure, the individuals scanning and indexing have an equally important role. The folders/files will be difficult to retrieve if not indexed accurately. In order to index effectively, the administrator needs to take special care to establish index settings correctly.

Before moving on to the individual components of capture and index, see the below overview of this process.



Static and Dynamic Folders

In order to adequately explain the index structure, let's begin with a sample folder structure. The administrator may want to begin with the end in mind when establishing the initial folder structure in ES Imaging.



As shown, the Personnel folder consists of a *static* “Consultants” and “Employees” folder. Within each of these are “Active” and “Separated” *static folders*. Within each of these is the actual folder for a specific person. And within each person’s folder is an “Evaluation” and/or “Discipline” document folder.

In structure shown, the “Consultants”, “Employees”, “Active” and “Separated” folders are all *static* and are created by the administrator. The individual person’s folder and the “Evaluation” and “Discipline” folders are *dynamic* and are created during indexing.

Refer to the folder structure previously identified for your organization. Label each folder with whether it is *static* (known and setup by the administrator) or if it is *dynamic* (*created during indexing*). If not previously done, label each folder with *keywords* useful for searching and for each *dynamic* folder, the *dynamic naming* scheme to be used when folders are created.

Common Folder Needs

Several folders may share common needs and therefore can use a common *folder type*. For each static *folder*, determine whether there are others at the same folder level that are similar.

Folder Types

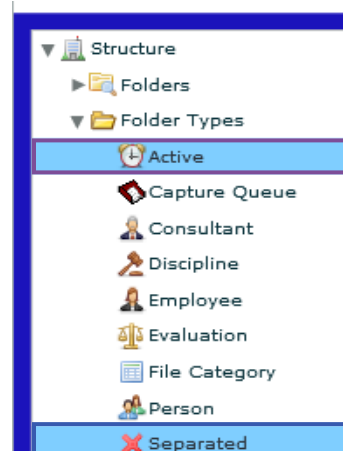
The screenshot shows the configuration window for the 'Person' folder type. The 'Name' field is set to 'Person'. The 'Icon' dropdown is set to 'Group'. The 'Keyword Types' section has two columns: 'Available' and 'Assigned'. The 'Available' column lists 'Date', 'Discipline Type', 'Full Name', 'Hire Date', and 'SSN'. The 'Assigned' column is empty. A purple arrow points from the 'Assigned' column to the text 'No Keywords'. At the bottom, there are fields for 'Dynamic Name' (set to '--SELECT--'), 'Use in Capturing' (set to 'false'), and 'Use in Indexing' (set to 'false'). Buttons for 'Refresh', 'Edit', 'Cancel', 'Save', and 'Delete' are at the bottom.

Folder Type

The “Consultants” and “Employees” folders are both similar and neither will require *keywords* associated to them. Additionally, these folders share the same icon. Therefore a folder type of “Person” will be created, so that settings can be shared by both static folders. Since files will not be placed directly into this folder level, the “Use in Capturing” and “Use in Indexing” are set to “false”.

On the other hand, the “Active” and “Separated” static folders (shown below) do not require keywords but they do require different icons to be more user-friendly. Consequently, both of these static folders will have a unique folder type.

Like the “Person” folder type, these folders do not have keywords and will not have files indexed to them.



The screenshot shows the configuration window for the 'Active' folder type. The 'Name' field is set to 'Active'. The 'Icon' dropdown is set to 'Clock'. The 'Keyword Types' section has two columns: 'Available' and 'Assigned'. The 'Available' column lists 'Date', 'Discipline Type', 'Full Name', 'Hire Date', and 'SSN'. The 'Assigned' column is empty. At the bottom, there are fields for 'Dynamic Name' (set to '--SELECT--'), 'Use in Capturing' (set to 'false'), and 'Use in Indexing' (set to 'false'). Buttons for 'Refresh', 'Edit', 'Cancel', 'Save', and 'Delete' are at the bottom.

Folder Type

The screenshot shows the configuration window for the 'Separated' folder type. The 'Name' field is set to 'Separated'. The 'Icon' dropdown is set to 'Red X'. The 'Keyword Types' section has two columns: 'Available' and 'Assigned'. The 'Available' column lists 'Date', 'Discipline Type', 'Full Name', 'Hire Date', and 'SSN'. The 'Assigned' column is empty. At the bottom, there are fields for 'Dynamic Name' (set to '--SELECT--'), 'Use in Capturing' (set to 'false'), and 'Use in Indexing' (set to 'false'). Buttons for 'Refresh', 'Edit', 'Cancel', 'Save', and 'Delete' are at the bottom.

Folder Type

Different Icons

Administration Guide

The image shows two side-by-side screenshots of the administration interface. The left screenshot is for the 'Discipline' folder type, and the right is for the 'Evaluation' folder type. Both forms have a 'Folder Type' label at the bottom. A purple arrow points from the 'Discipline' form to the 'Evaluation' form, labeled 'Different Icons, Keyword Types & Dynamic Names'.

***Discipli...**

***Name** Discipline

Icon Hammer and Gavel

Keyword Types

Available

- Full Name
- Hire Date
- SSN

Assigned

- Date
- Discipline Type

Dynamic Name Discipline

Use in Capturing false

Use in Indexing true

Refresh Edit Cancel Save Delete

Folder Type

***Evaluati...**

***Name** Evaluation

Icon Scale

Keyword Types

Available

- Discipline Type
- Full Name
- Hire Date
- SSN

Assigned

- Date

Dynamic Name Evaluation

Use in Capturing false

Use in Indexing true

Refresh Edit Cancel Save Delete

Folder Type

For each dynamic folder (such as “Discipline” and “Evaluation”) a different *folder type* will be necessary, since these folders have different *icons*, *keywords* and possibly *dynamic naming* schemes (as shown above). It is important for the administrator to note that these *folder types* created for *dynamic folders* will not be associated to a folder until indexing.

Search Needs

In order for files to be searched by appropriate criteria, *keyword types* will need to be set up and attached to the appropriate *folder type(s)* by the administrator. Without *keyword types*, only the contents of a *folder name* can be searched.

Keyword Types

The image shows two side-by-side screenshots of the administration interface. The left screenshot is for the 'Discipline Type' keyword type, and the right is for the 'Discipline' folder type. A purple arrow points from the 'Discipline Type' form to the 'Discipline' form, labeled 'Keyword Type associated to a folder type'.

***Discipline Ty...**

***Name** Discipline Type

***Value Type** string

***Control Type** EditableDropDownList

Refresh Edit Cancel Save Delete

Keyword Type

***Discipli...**

***Name** Discipline

Icon Hammer and Gavel

Keyword Types

Available

- Full Name
- Hire Date
- SSN

Assigned

- Date
- Discipline Type

Dynamic Name Discipline

Use in Capturing false

Use in Indexing true

Refresh Edit Cancel Save Delete

Folder Type

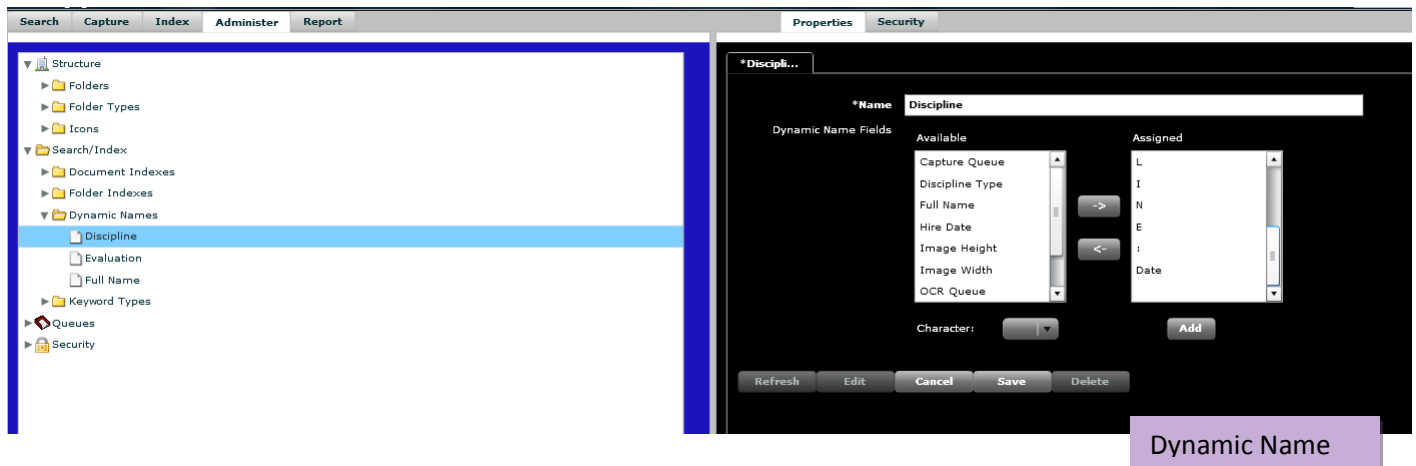
All files placed into the “Discipline” folder need to provide for searching by “date” and “discipline type”. These search attributes are all considered *keyword types* and are associated to the “Discipline” *folder type*. By doing so, when documents are indexed, each of the values for the “date” and “discipline type” *keywords* should be entered. This process ensures that each document placed in the system, can later be efficiently accessed.

Folder Naming Scheme

The *dynamic name* scheme is built by the administrator using a combination of *keywords* and *special characters*. When *keywords* are assigned during indexing, the *document folder* is created according to the established *dynamic name* scheme, if the document folder does not already exist.

Dynamic Name

The “Discipline” folder has a dynamic name that appends the *date* (entered during indexing) to the word “Discipline”. This is accomplished with a combination of a *keyword type* and *dynamic name*. A *dynamic name* scheme may also consist of a character string without any keywords.



Indexing Folder Destination

Both *document* and *folder indexes* are used during indexing to direct the scanned files to the appropriate location. These indexes assist the *index user* by reducing the typing necessary during the index process.

The *document index* contains the *static folder* (or path) destination where files will be placed. This index can also be thought of as the predetermined folder for indexed file, as it is a known folder and is established by the administrator during folder configuration.

Document Index

Folder index(es) are established and then associated to the appropriate *document index(es)* to provide further direction as to where the files will be placed during indexing.

Folder Index

Administration Guide

The first technique that can be used for establishing indexes is to set up a separate *document index* for each static path.

As shown below, there are four *document indexes* that correspond to the “Active” and “Separated” folders within both “Consultants” and “Employees”. This technique is suitable when the number of static paths is limited to less than eight possibilities.

1) Where to search or place *document folders*

2) See below Folder Index

Document Index

Each folder that can be created during indexing must have an associated *folder index*. The *folder index* is used when a *dynamic folder* is created during indexing so the system knows the correct *folder type* designation.

Used to create new *document folders*

Folder Index

The “How Indexes Impact Indexing” section will cover indexing in more detail.

Administration Guide

The second technique that can be used for establishing indexes is to set up a shared *folder index* for each static path level.

As shown below, there are two *document indexes* that correspond to the “Consultants” and “Employees” static folders. This technique is suitable when the number of static paths is eight or more, or when this number is expected to increase with time.

The screenshot shows the Indexing software interface. On the left, the 'Structure' pane displays a hierarchy of folders: Personnel (Consultants, Employees), Folder Types, Icons, and Search/Index (Document Indexes). Two purple arrows point from the 'Consultants' and 'Employees' folders to the right-hand configuration pane. The right-hand pane is titled '*Consultant' and contains fields for *Name (Consultant), *Static Folder (Personnel\Consultants\), *Folder Name Control (DropDownList), and After Indexing (Retain Last Folder). It also has checkboxes for Visible, Search Enabled, and Folder Name Override. Below these are two lists: 'Available' (Employee) and 'Assigned' (Status, Consultant). At the bottom, there are buttons for Refresh, Edit, Cancel, Save, and Delete. A purple label 'Document Index' is positioned at the bottom right of the configuration pane.

1) Where to search or place *document folders*

2) See below Folder Indexes

Document Index

With this technique, a *folder index* for each folder that can be created during indexing there must also be a common *folder index* for each level in the *static folder* path destination. The *folder indexes* must be added in the order which they should appear in the folder hierarchy.

The first screenshot shows the '*Status' folder index configuration. It has fields for *Name (Status), *Folder Type (Status), *Folder Name Control (Text), and After Indexing (Retain Last Folder). It also has checkboxes for Visible, Search Enabled, and Folder Name Override. At the bottom are buttons for Refresh, Edit, Cancel, Save, and Delete. A purple label 'Folder Index' is positioned at the bottom right.

Folder Index

The second screenshot shows the '*Consultant' folder index configuration. It has fields for *Name (Consultant), *Folder Type (Consultant), *Folder Name Control (DropDownList), and After Indexing (Retain Last Folder). It also has checkboxes for Visible, Search Enabled, and Folder Name Override. At the bottom are buttons for Refresh, Edit, Cancel, Save, and Delete. A purple label 'Folder Index' is positioned at the bottom right.

Folder Index

The first technique requires more administration, since each static folder combination will need to be created by the administrator. However, the *Index User* will save time during indexing, since one or more folder level selection/entry is eliminated.

Administration Guide

The second technique will create less administration and allows for greater scalability. However, the *Index User* will need to enter/select additional index information, as shown below.

The image displays two side-by-side screenshots of a software interface, both featuring a purple tab labeled "Index Tab".

Left Screenshot (First technique):

- Document Folder Type: Evaluation
- Document Index: Consultant - Active
- Consultant Name: Smith, Jane-444-44-4444
- Full Name: Smith, Jane
- SSN: 444-44-4444
- Evaluation Name: EVALUATION:2009-02-10
- Date: 2009-02-10

Right Screenshot (Second technique):

- Document Folder Type: Evaluation
- Document Index: Consultant
- Status Name: Active
- Consultant Name: --SELECT--
- Full Name:
- SSN:
- Evaluation Name: EVALUATION:2007-04-29
- Date:

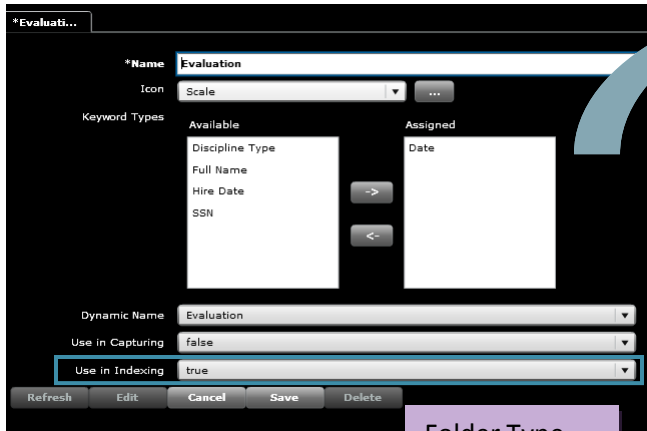
First technique, creating separate *document indexes* for each static folder path.

Second technique, creating a common folder index for each level within the static folder path.

How Indexes Impact Indexing

Now let's map the *document* and *folder index* that the administrator establishes to the indexing area (*Index Tab*) that the *Index User* uses.

Administrator View

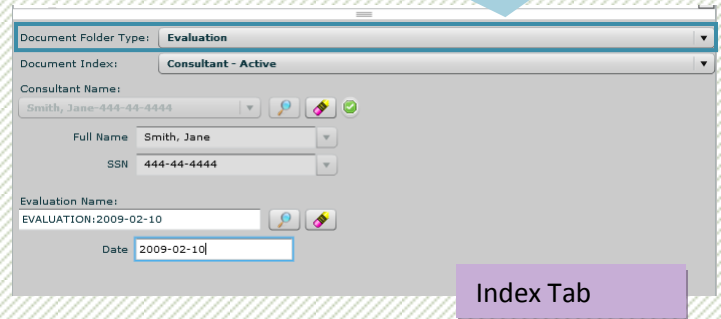


The screenshot shows the 'Evaluation' folder type configuration in the Administrator View. The 'Use in Indexing' checkbox is checked. A blue arrow points from this checkbox to the 'Index User View'.

Folder Type

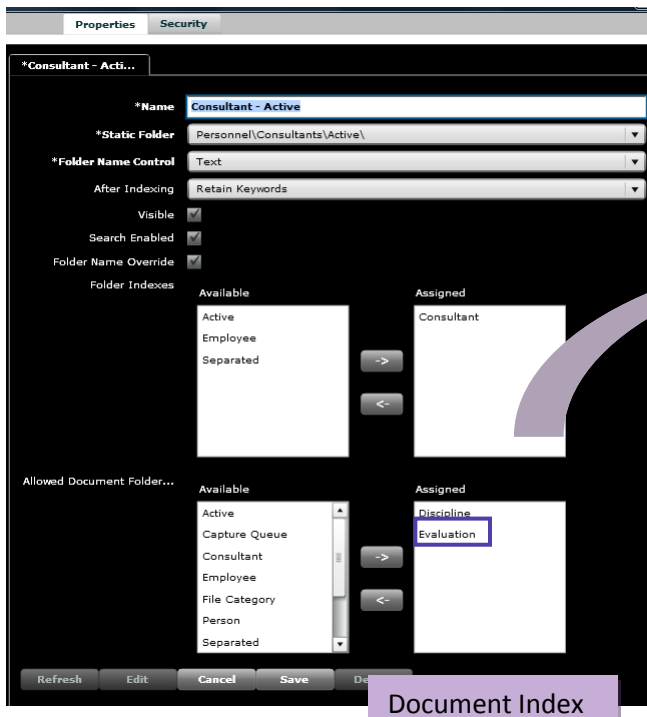
Index User View

Contains all *folder types* with "Use in Indexing" set to "true"



The screenshot shows the 'Evaluation' folder type configuration in the Index User View. The 'Document Index' is set to 'Consultant - Active'. A purple arrow points from this selection to the 'Index Tab'.

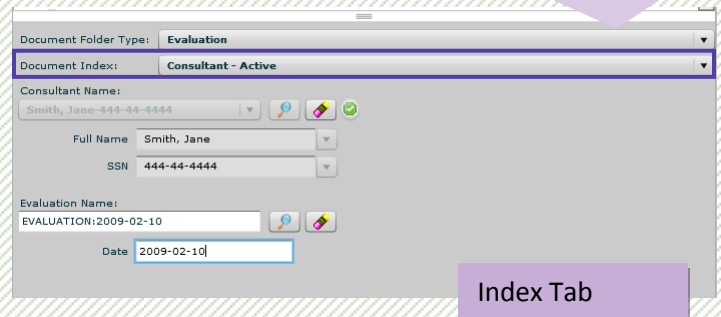
Index Tab



The screenshot shows the 'Consultant - Active' document index configuration in the Administrator View. The 'Allowed Document Folder...' list includes 'Evaluation'. A blue arrow points from this list to the 'Index User View'.

Document Index

Contains all *document indexes* that contain an *allowed document folder type* = "Evaluation"

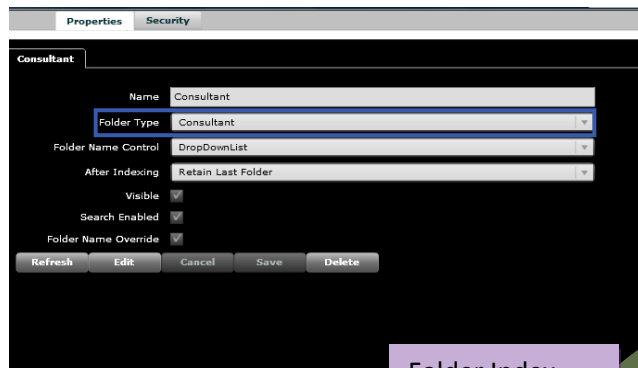


The screenshot shows the 'Consultant - Active' document index configuration in the Index User View. The 'Document Index' is set to 'Consultant - Active'. A purple arrow points from this selection to the 'Index Tab'.

Index Tab

Since the *Index User* selected the "Consultant - Active" *document index* and it contains the "Consultant" *folder index*, the "Consultant" *folder type* is evaluated for the *keyword types* and *dynamic name* used during indexing.

Administrator View



Properties Security

Consultant

Name: Consultant

Folder Type: Consultant

Folder Name Control: DropDownList

After Indexing: Retain Last Folder

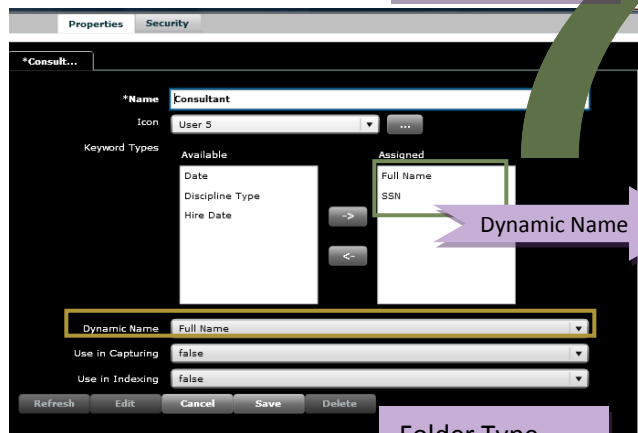
Visible: ☒

Search Enabled: ☒

Folder Name Override: ☒

Buttons: Refresh Edit Cancel Save Delete

Folder Index



Properties Security

*Consult...

Name: Consultant

Icon: User 5

Keyword Types

Available	Assigned
Date	Full Name
Discipline Type	SSN
Hire Date	

Dynamic Name: Full Name

Use in Capturing: false

Use in Indexing: false

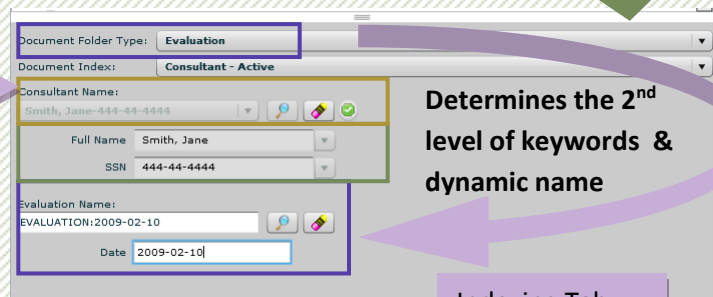
Buttons: Refresh Edit Cancel Save Delete

Dynamic Name

Folder Type

Index User View

Determines the 1st level of keywords & dynamic name scheme



Document Folder Type: Evaluation

Document Index: Consultant - Active

Consultant Name: Smith, Jane-444-44-4444

Full Name: Smith, Jane

SSN: 444-44-4444

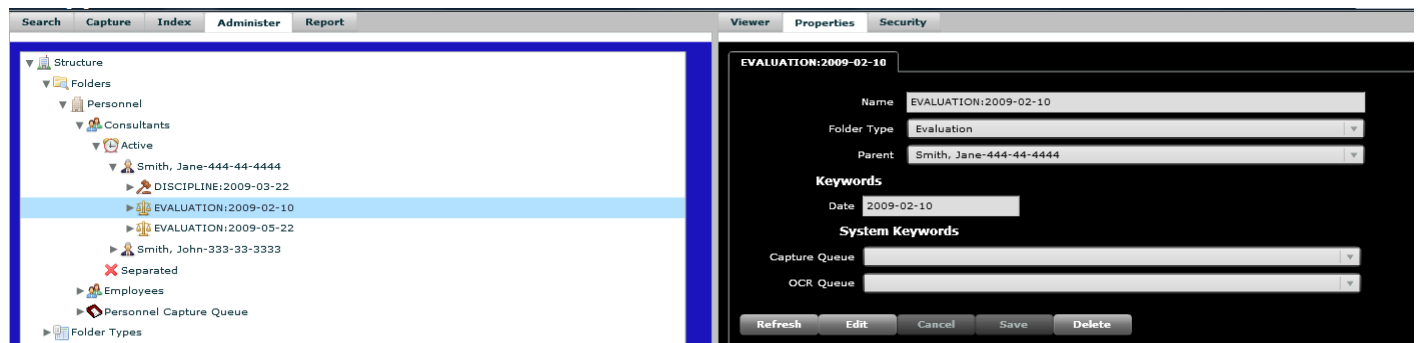
Evaluation Name: EVALUATION:2009-02-10

Date: 2009-02-10

Determines the 2nd level of keywords & dynamic name

Indexing Tab

After files are indexed with the above index values, the below *document folder* and its contents are available in ES Imaging.



Search Capture Index Administer Report

Viewer Properties Security

Structure

- Folders
 - Personnel
 - Consultants
 - Active
 - Smith, Jane-444-44-4444
 - DISCIPLINE:2009-03-22
 - EVALUATION:2009-02-10
 - EVALUATION:2009-05-22
 - Smith, John-333-33-3333
 - Separated
 - Employees
 - Personnel Capture Queue

- Folder Types

Properties

EVALUATION:2009-02-10

Name: EVALUATION:2009-02-10

Folder Type: Evaluation

Parent: Smith, Jane-444-44-4444

Keywords

Date: 2009-02-10

System Keywords

Capture Queue:

OCR Queue:

Buttons: Refresh Edit Cancel Save Delete

Indexing Behavior Settings

There are various administrator controlled index settings that greatly reduce the time needed for indexing. Most of these behavior settings are associated to both *document* and *folder indexes*.

- *Folder Name Control* – Allows for either a *text* (typed name), *drop down list* (choose from list) or *editable drop down list* (can type in a new list entry)
- *After Indexing* – Provides option to *clear* all contents, *retain last folder* (keeps both folder and keywords) or *retain keywords*, after indexing a document
- *Visible* – Hides or shows indexing values
- *Search Enabled* – Allows for the *Index User* to search for a folder that is already present, so that duplicate folders are not created. This will search for the folder name entered.
- *Folder Name Override* – Allows for a folder name to be entered, in cases where there is not a *dynamic name* scheme assigned
- *Variable Depth (folder indexes only)* – Allows the search for a folder in a given level to be at a variable depth in the hierarchy. This is useful if you have the same folder type embedded inside each other with no hard set structure. The following example demonstrates how folder types of 'Photo Album' may be embedded inside each other to have a variable hierarchy. If checked, the *Index User* can select *any* of the folders below the 'Pictures' level in the following example. If not checked, the *Index User* would *only* be able to choose 'Family Album'.

```
My Home
  Pictures
    Family Album
      Dad's Album
        Reunions
          2010
            Grandparents
              Mom's Album
                Grandma
```

Scanned Documents "Inbox"

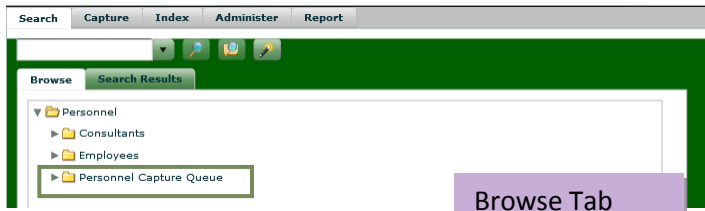
The capture and OCR queue are used during the capture process to hold scanned/imported documents prior to indexing. It is basically an "inbox" for the *index user/group* assigned to the capture queue.

Capture / OCR Queues

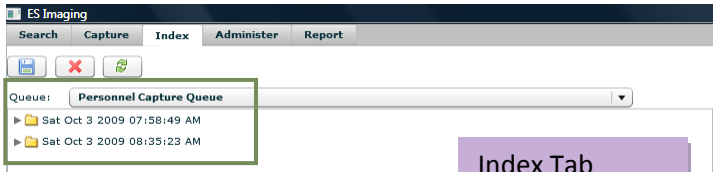
When a new *root folder* is created a corresponding *capture queue* is also created. This *capture queue* folder is named with the *root folder* name appended with the words "capture queue". If there are multiple individuals who will be indexing, it might be helpful to create personalized capture queues for each *index user/group*.

Quick Tip – A *root folder capture queue* is automatically created upon installation of ES Imaging.

Administration Guide



Browse Tab



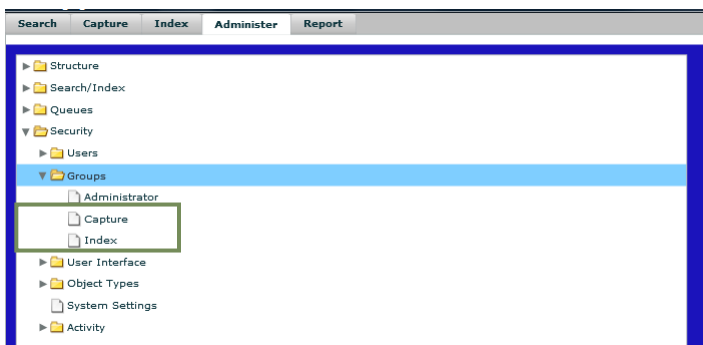
Index Tab

The "Personnel Capture Queue" contains scanned documents with a date and time stamp.

Capture and Index Users Groups

The *Capture User* scans/imports files into batches and routes them to the appropriate *capture queue*. This *capture queue* serves as an inbox for the *Index User*. In order for the Capture and Index Users to process documents, each will need specific permissions. The Capture User will need the ability to save to the appropriate capture queue. The Index User will need to *view* the appropriate capture queue and will also need *save* permission to the appropriate folders/files. Upon installation of ES Imaging, generic *Capture User* and *Index User* security groups are established and are granted the appropriate *view* and *save* permissions for the capture queue. However, the *Index User* security group will need additional permission granted to the appropriate folders/files.

Quick Tip The *Capture and Index User* groups are automatically created upon installation of ES Imaging.



Individual users will need to be assigned to each of the *capture* and *index* user groups. As mentioned in the security section of this manual, the use of groups eases the administration of security, specifically when multiple people require similar permissions. Refer to the “*Preparation for Security Administration*” and “*Establishing Security*” sections of this manual for guidance on security. Depending on the size of the organization, more than one individual may be necessary for capturing and indexing documents. If this is the case, additional groups might be necessary. The existing “Capture User” and “Index User” may need to be renamed to include the functional area they represent. For instance, there may be a need for a “Human Resource Capture User” and an “Insurance Capture User”. The more meaningfully the group names describe the area and the role, the easier it will be to administer permissions.

Capture and Index User Groups

It is important to note that many times an individual will belong to multiple groups, especially in a small organization. ES Imaging does allow this feature, but it is imperative to understand the rules on how permission conflicts are resolved (see “*Preparation for Security Configuration*”, “*Permission Hierarchy when User is in Multiple Groups*” subsection).

Establishing Folder Structure

Considerable analysis should be performed to establish the most effective folder structure for the organization. Prior to beginning these steps, review the “*How to Begin*”, “*Preparation for Folder Configuration*” and “*Preparation for Capture and Index Configuration*” sections.

Creation of the System Root

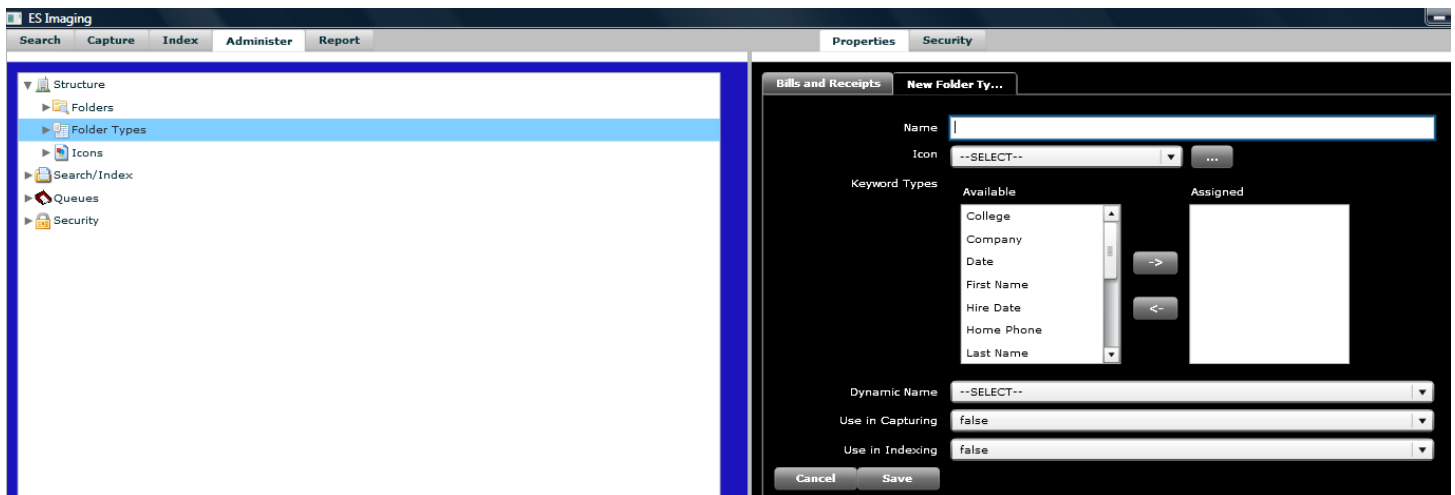
- Logon to ES Imaging
- At prompt to create a new root folder, type in the appropriate name.

Quick Tip – The “system root” is typically the company or organization name. Multiple root folders are allowed.

Creation of Folder Types

See “*Preparation for Folder Configuration*” and “*Preparation for Capture and Index Configuration*” section for guidance.

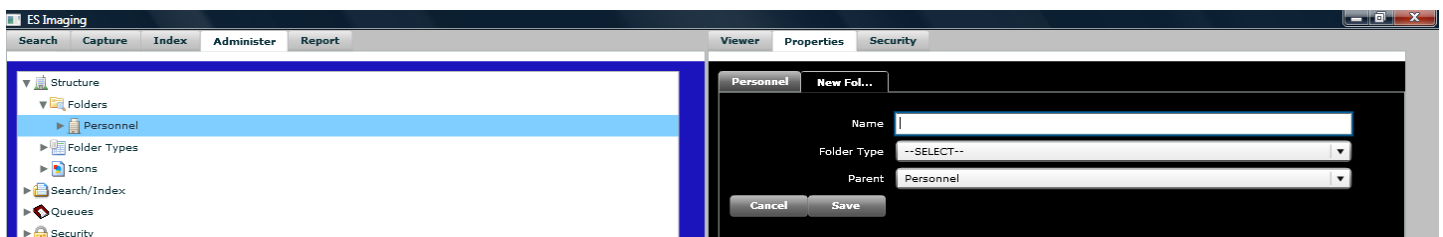
- Within the “Administer” tab, expand “Structure”
- Right click on “Folder Type”
- Click on “New Folder Type”
- Type an appropriate folder type “Name” and select the appropriate “Icon” (using the ...)
- Highlight the appropriate “Keyword Type(s)” and click on the right arrow (can be added later, see “*Capture and Index Configuration*” section)
 - It is possible to have multiple values for the same Keyword Type by using the ∞-> button. Can even be used in conjunction with Panel Keyword Types. The user will see a button with a ‘+’ symbol in order to allow them to add more values to a folder of this type.
- Select the appropriate “Dynamic Name” (can be added later, see “*Capture and Index Configuration*” section)
- Select true or false for “Use in Capturing” and “Use in Indexing”
- Click on “Save”



Creation of Folders

See “*Preparation for Folder Configuration*” section for guidance.

- Within the “Administer” tab, expand “Structure”
- Expand “Folders”
- Right click on the appropriate root or parent folder
- Click on “New Folder”
- Type an appropriate folder “Name”
- Select the appropriate “Folder Type”
- “Parent” defaults to the highlighted folder, but may be changed
- Click on “Save”



Capture and Index Configuration

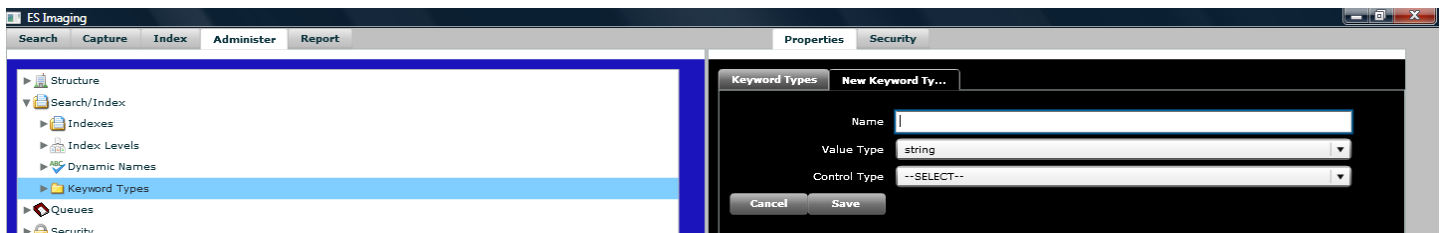
Prior to beginning these steps, complete the “*How to Begin*” and “*Preparation for Folder Configuration*” and “*Preparation for Capture and Index Configuration*” sections.

Creation of Keyword Types

See “*Preparation for Capture and Index Configuration*”, “*Search Needs*” subsection for guidance.

Quick Tip – The “system root” is typically the company or organization name. Multiple root folders are allowed.

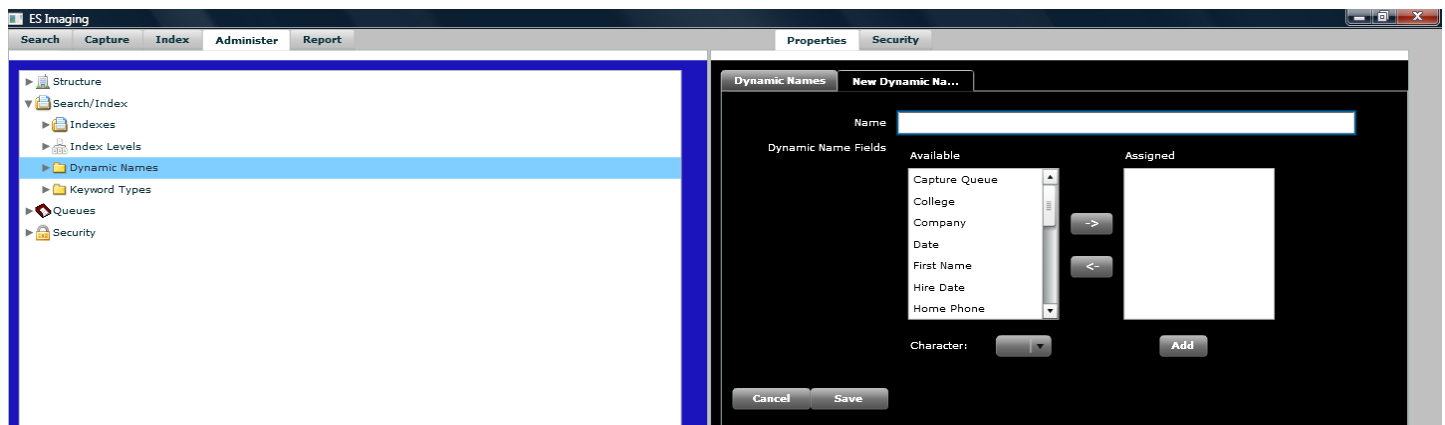
- Within the “Administer” tab, expand “Search/Index”
- Right click on “Keyword Types”
- Click on “New Keyword Type”
- Type an appropriate name
- Select the appropriate “Value Type”
- Select the appropriate “Control Type”
 - Panels allow grouping of keywords, such as an Address. Example:
 - Address may have Line One, Line Two, City, State, and Zip.
 - Panels may be searched in Advanced Searching with any of the values of the sub-keywords. The internal value stored in a Panel is an aggregate of all these values, separated by commas.
- Click on “Save”



Creation of Dynamic Names

See “*Preparation for Capture and Index Configuration*”, “*Folder Naming Scheme*” subsection for guidance.

- Within the “Administer” tab, expand “Search/Index”
- Right click on “Dynamic Names”
- Click on “New Dynamic Name”
- Type an appropriate name
- Select the appropriate “Keyword Type” needed for the name
- If necessary, select the appropriate “Character” to append to the name
- Continue to build the dynamic name until complete
- Click on “Save”



Updating Folder Type with Index Settings

- Within the “Administer” tab, expand “Structure”
- Click on the appropriate “Folder Type”
- Select the appropriate “Keyword Type(s)” that are required for searching
- Select the appropriate “Dynamic Name” scheme
- Select true or false for “Use in Capturing” and “Use in Indexing”
- Click on “Save”

Creation of Capture and OCR Queues

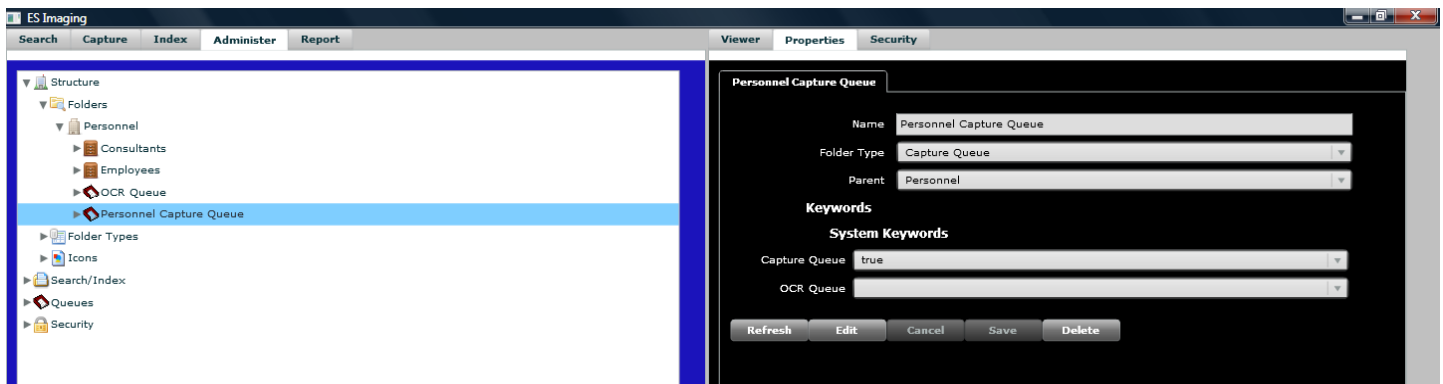
See “Preparation for Capture and Index Configuration”, “Scanned Document Inbox” subsection for guidance.

- Within the “Administer” tab, expand “Structure”
- Expand “Folders”
- Right click on the root folder
- Click on “New Folders”
- Type an appropriate capture/ocr queue “Name” (recommend name contains “capture queue” or “OCR queue”)
- Select the appropriate “Icon”
- For “Folder Type” select “Capture Queue” (used for both capture and OCR queues)
- Select “true” for the “Capture Queue” system keyword (for capture queues)
- Select “true” for the “OCR Queue” system keyword (for OCR queues)
- Click on “Save”

Quick Tip – If documents require full-text searching, “OCR queue” system keyword must be set to “true”.

Updating a Capture Queue Name

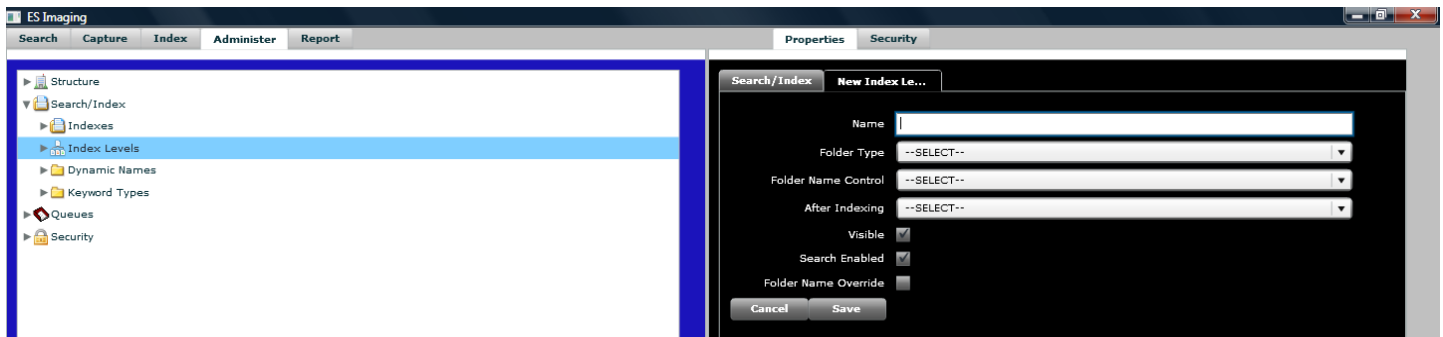
- Within the “Administer” tab, expand “Structure”
- Expand both “Folders” and the root folder
- Click on the appropriate capture queue folder
- Click on “Edit”
- Update the folder “Name”
- Click on “Save”



Creation of a Folder Index

See “*Preparation for Capture and Index Configuration*”, “*Indexing Folder Destination*” subsection for guidance.

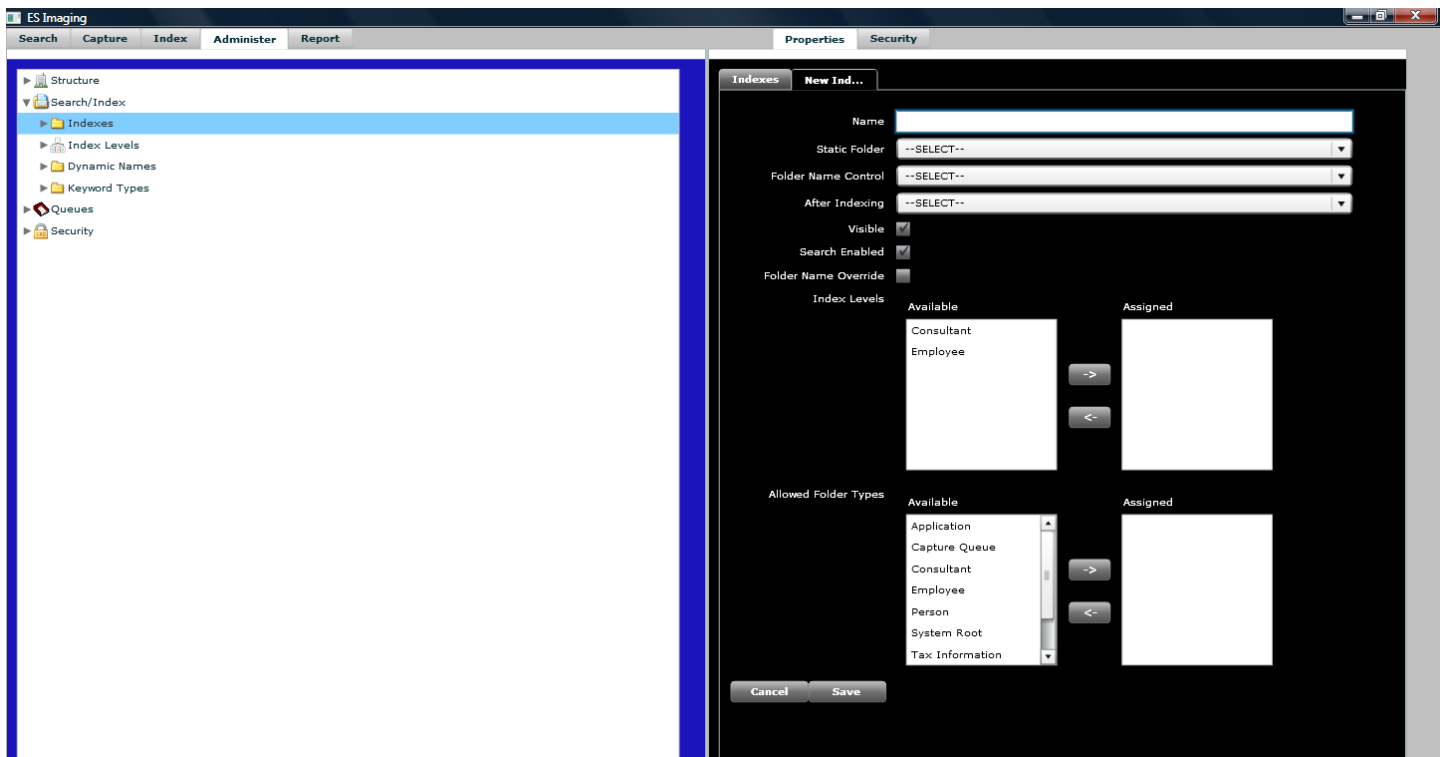
- Within the “Administer” tab, expand “Search/Index”
- Right click on “Folder Index”
- Click on “New Folder Index”
- Type an appropriate folder index “Name”
- Select the other appropriate values/options
- Click on “Save”



Creation of a Document Index

See “*Preparation for Capture and Index Configuration*”, “*Indexing Folder Destination*” subsection for guidance.

- Within the “Administer” tab, expand “Search/Index”
- Right click on “Document Index”
- Click on “New Document Index”
- Type an appropriate document index “Name”
- Select the appropriate “Static Folder”
- Select the other appropriate values, including “Folder Indexes” and “Folder Types”
- Click on “Save”

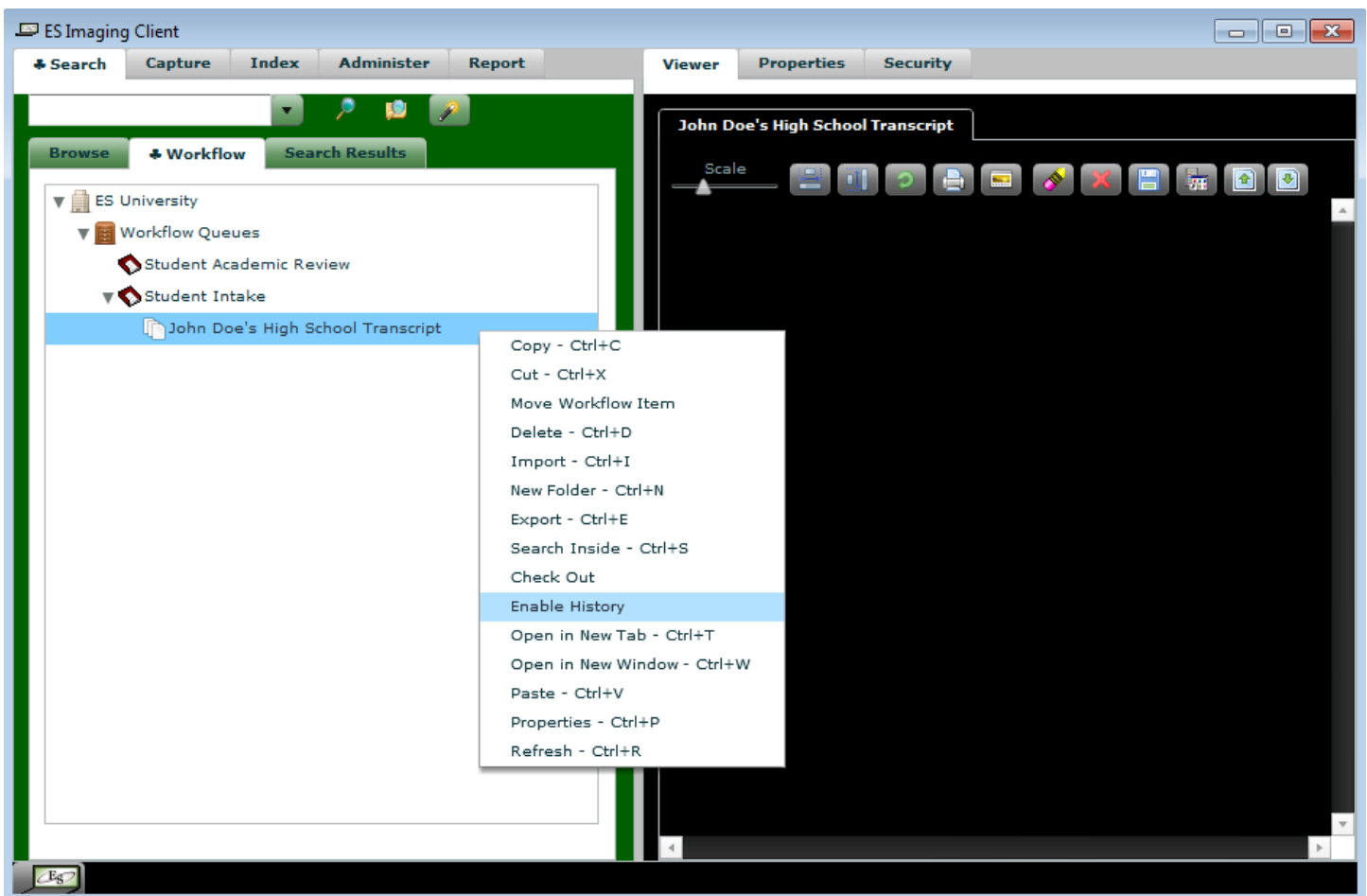


Folder History

History can be enabled on any folder in ES Imaging. This allows someone to see who changed what on a given folder, and at what time.

Enabling Folder History

- Navigate to the desired folder in any of the ES Imaging trees.
- Right click on the desired folder.
- Left click on the “Enable History” menu option. See the figure below.
- You will be presented with a dialog saying that history is now enabled. Simply click “Ok”.

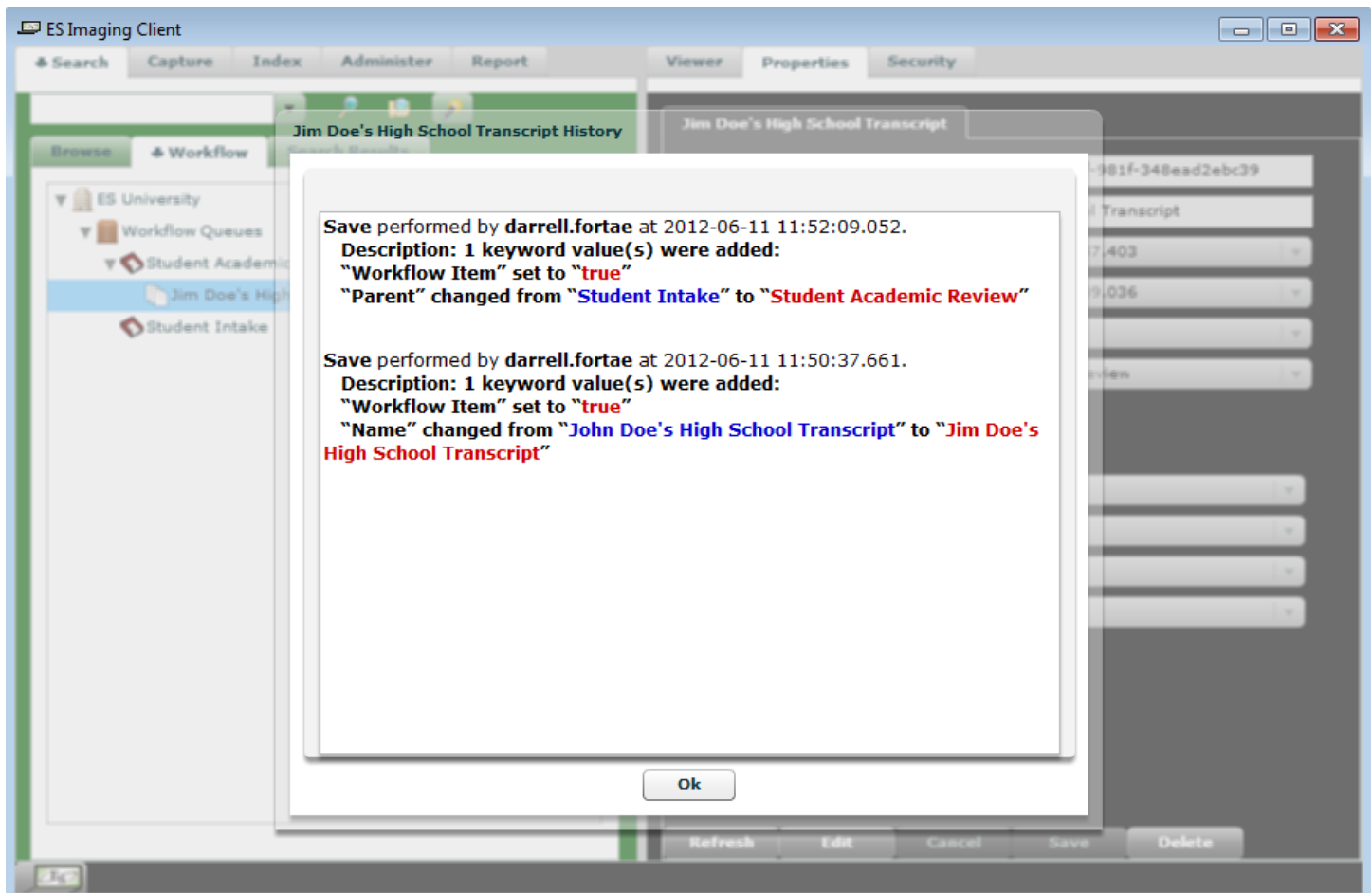


Disabling Folder History

- Navigate to the desired folder in any of the ES Imaging trees.
- Right click on the desired folder.
- Left click on the “Disable History” menu option.
- You will be presented with a dialog saying that history is now disabled. Simply click “Ok”.

Viewing Folder History

- Navigate to the desired folder in any of the ES Imaging trees.
- Right click on the desired folder.
- Left click on the “View History” menu option.
- You will be presented with a dialog showing the full history for the folder. See the figure below for an example.
- Click “Ok” when you have finished examining the history.



Note: “View History” will only show up if you have permission, history is enabled, and the folder has had at least 1 change since history was enabled.

File Encryption

Individual files can be encrypted/decrypted from within ES Imaging. The encryption cannot be reversed in the case of a lost/forgotten password, so please use great caution.

Please refer to the “Encrypting” and “Decrypting” sections of the ES Imaging User Manual for more information.

File Versioning

Versioning can be enabled for individual files. Versioned files have a tag name associated so it is easy for a person to identify a specific one. Security can actually be controlled for each version, since each is its own file.

Please refer to the “Create New Version” and “Replacing Current Version” sections of the ES Imaging User Manual for more information.

Security Administration

Preparation for Security Administration

ES Imaging security is designed to allow for the most specific needs of an organization. In order to provide maximum flexibility, security is managed with functions (*view, save, delete, copy, etc.*), system permission defaults and group permission levels.

Functions

Some functions allow/protect file content from being viewed, while others allow/protect files from being inadvertently edited or removed. The list of functions below briefly describes each.

- *View* – see images and properties
- *Save* – change/save properties
- *New Image Markup* – add/change/remove image markups
- *View Security* – see security details
- *Save Security* – change/save security details
- *New* – create a new object
- *Delete* – permanently remove the object
- *Copy* – copy folders and files
- *Cut* – cut folders and files
- *Paste* – paste folders and files
- *Import* – import files into a folder (usually from a local/network disk)
- *Export* – export folders/files to a local/network disk
- *Encrypt* – encrypt files
- *Decrypt* – decrypt files
- *New Version* – create a new version of a file
- *Check Out* – check out an object (gives the user exclusive update rights)
- *Assign Check Out* – check out an object to a specific user
- *Check In* – check in an object (removes exclusive update rights)
- *Enable History* – enables history on folders
- *Disable History* – disables history on folders
- *View History* – see the history on folders

Types of Permission Settings

There are three types of permissions to grant within the system. The “allow” permission grants access to the designated item(s) for the designated individual(s). The “deny” permission prevents access to the designated item(s) for the designated individual(s). Both the “allow” and “deny” permissions are known as “explicit” permission settings.

The “unspecified” permission implies moving to the next higher permission level to determine explicit permissions. This is explained further in the *“Permission Hierarchy”* and *“Permission Hierarchy when User in Multiple Groups”* subsections.

Copy	<input type="radio"/> Allow	<input type="radio"/> Deny	<input type="radio"/> Unspecified
Cut	<input type="radio"/> Allow	<input type="radio"/> Deny	<input type="radio"/> Unspecified
Delete	<input type="radio"/> Allow	<input type="radio"/> Deny	<input type="radio"/> Unspecified

Group Permissions

Several *users* of the system may require identical permissions. *Group* permission bundles *users* together so permissions can be administered as a whole for these individuals.

Groups

User Permissions

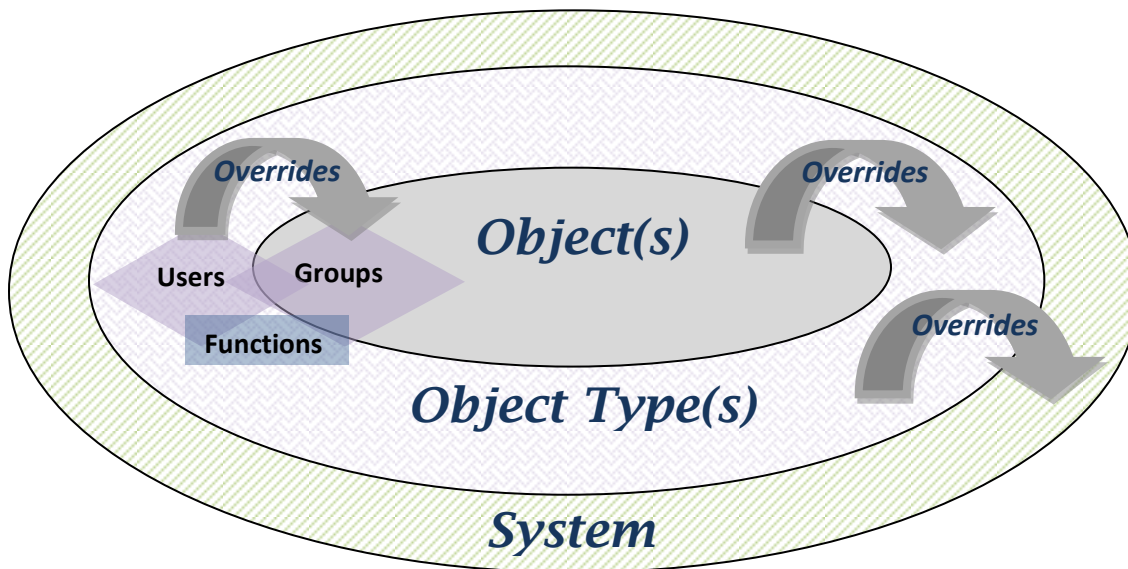
At times, it may be necessary to manage security at an individual level. For example, when an individual requires access to a restricted, confidential folder/file then the user can be granted access to the appropriate folder/file.

Users

Permission Levels

Permissions can be established at three levels: the *system level* (for all users), an intermediate level (*object type*) or a more specific level (*object*). Each level allows for common explicit (allow/deny) permissions to be used by other levels, as shown below.

Allow/Deny



System Default Permissions

System level permission allows each of the previously mentioned functions to be set to “allow” across all users. However, *system level* permissions are overridden by security granted at the *object type* and *object levels* (as shown above). Please note that ES Imaging automatically sets all of the function permissions to “deny”, except for the designated administrator which is granted global “allow” permission.

Object Type Permissions

An *object type* is a logical grouping of *objects* (see below for object explanation) within the system. These *object types* are used to more effectively maintain security within the system by administering permissions across all *objects* within the *object type*. *Users* and *group(s)* and the appropriate functions are designated as “allow”, “deny” or “unspecified” for each *object type*.

Object Types

- *Admin Data* – includes administrative items (*structure, search/index, queues and security*)
- *User Interface* – the graphical features, like buttons and tabs
- *File* – a page of a scanned/imported image, text document, spreadsheet, etc.
- *Folder* – grouping of files and other folders
- *Folder Type* – templates for folders
- *Icon* – graphical symbols that can be associated to folder types
- *Document Index* – defines behaviors and properties during indexing for documents
- *Folder Index* - defines behavior and properties during indexing for a specific folder type
- *Keyword Type* – templates for folder keyword values to allow finding/searching
- *Report* – various administrative views of ES Imaging activity , logs, data, etc.
- *Security Group* – established by administrator to group similar users
- *Security User* – individual using the system

Quick Tip – this permission level is useful to allow a user or group **all folder/file** contents and nothing more.

Permissions set at this level are especially useful if all users of the system or a specific group(s) are allowed to view **all** file/folder content and nothing more. The “allow” permission set for the “view” function for the “file” and “folder” *object types* would easily accomplish this.

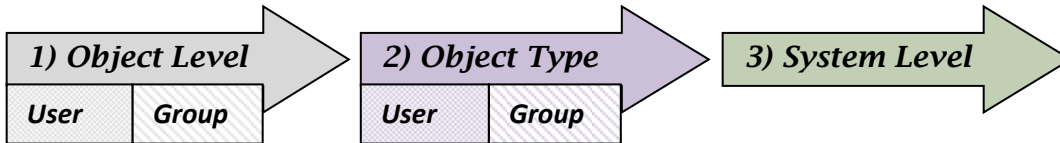
Object Permissions

An object within the system is simply a node of the tree structure. For instance, while “folder” is considered an *object type* (designating **all** folders), a specific named folder or file, is an *object*. *Users* and *group(s)* along with the appropriate functions are designated as “allow”, “deny” or “unspecified” for each object.

Objects

Permission Hierarchy

Earlier we discussed the three levels of permissions that can be managed within the system. In addition to these permissions, it is important to understand the hierarchy of permissions. The below diagram shows how permissions are determined:



Explicitly set allow/deny *user* permission attached to the *object level* overrides all others. If permissions are not explicitly set at the *object level* ("unspecified" permission), then the explicitly set *group* permission at the *object level* is used. This same evaluation repeats from left to right, as shown above.

Quick Tip – *user explicit permissions at an object level override all other permissions. System level permissions are only used when no other explicit permissions are set.*

Permission Hierarchy when User in Multiple Groups

An explanation is necessary for a scenario involving a user with "unspecified" permission who belongs to multiple groups, specifically when these groups have conflicting explicit security settings (for example one group is "deny" and the other is "allow" for a specific folder). The following evaluation applies:

- If the *system level* default is "deny", then the permission will result in "deny"
- If the *system level* default is "allow", then the permission will result in "allow"

Preset Group Permissions

Individuals perform a variety of functions when using ES Imaging. Some of these functional *groups* have preset permissions upon installation of ES Imaging. Each *group* carries with it different security settings as described below:

- *System Administrator* – Can perform **any function** in the system for **all objects**.
- *Administrator* – Same as system administrator, except they **cannot alter system settings**.
- *File/Folder Administrator* – Can perform **any function** on **all folders and files**.
- *Capture User* – Can **scan/import files** into the default capture queue.
- *Index User* – Can **specify keyword values** and **move documents** in the default capture queue to the **proper location** in the system.
- *File/Folder Viewer* – Can view **any folder and file**.
- *Report Viewer* – Can **view any report**.

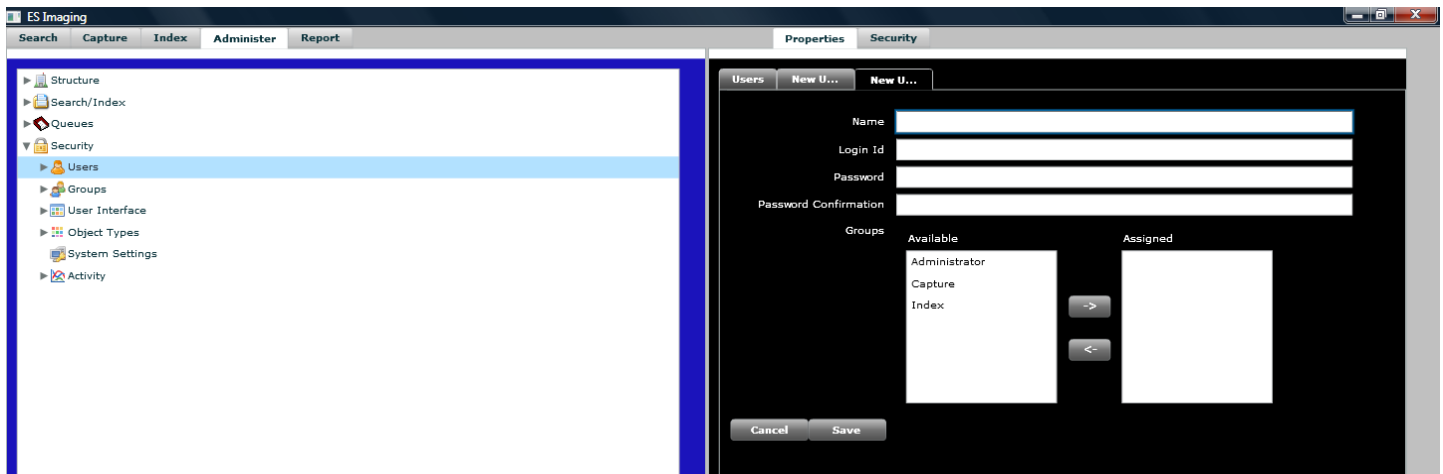
Although these *groups* are preset during ES Imaging installation, these can be changed by the *System Administrator* and *Administrator*. *Users* will need to be attached to the appropriate *group(s)* by the *Administrator*.

A *user* can belong to multiple *groups*. See the “*Permission Hierarchy when User in Multiple Groups*” subsection.

Establishing User Accounts

Before security settings can be established, *user* accounts must be established.

- Within the “Administer” tab, expand “Security”
- Right click on “Users”
- Click on “New User”
- Type in the individual’s “Name” (recommend last, first name format)
- Type in the appropriate “Login Id”, “Password” and “Password Confirmation”
- “Groups” will be updated later (see “*Establishing Security*” section)
- Click on “Save”

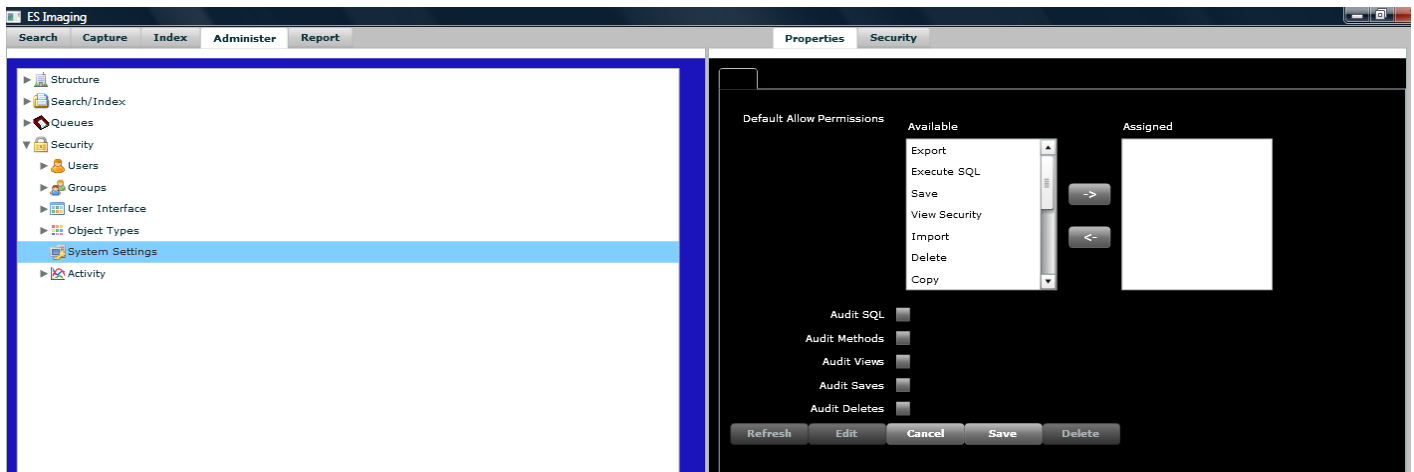


Establishing Security

See “*Preparation for Security Administration*” section, “*System Default Permissions*” and “*Permission Hierarchy*” subsections for guidance.

Setting System Default Permissions

- Within the “Administer” tab, expand “Security”
- Click on “System Settings”
- Click on the “Properties” tab
- Click on “Edit”
- Select the functions that require “allow” as a default
- Click on the right arrow
- Click on “Save”

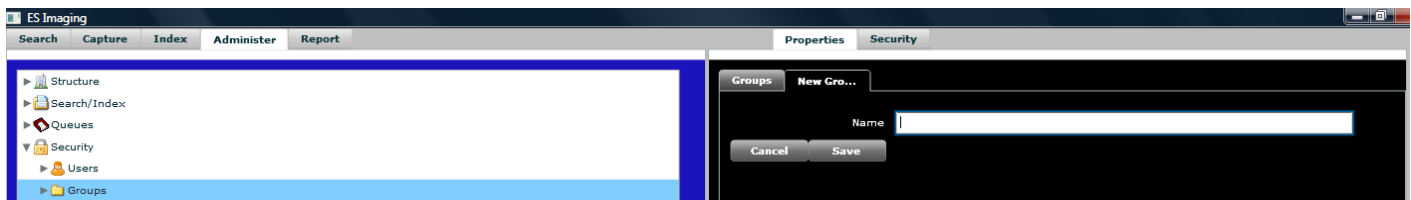


Creation of Groups

See “*Preparation for Security Administration*” section, “*Group Permissions*” and “*Preset Group Permissions*” subsections for guidance.

- Within the “Administer” tab, expand “Security”
- Right click on “Groups”
- Click on “New Group”
- Type in the group “Name” (a meaningful name is recommended)
- Click on “Save”

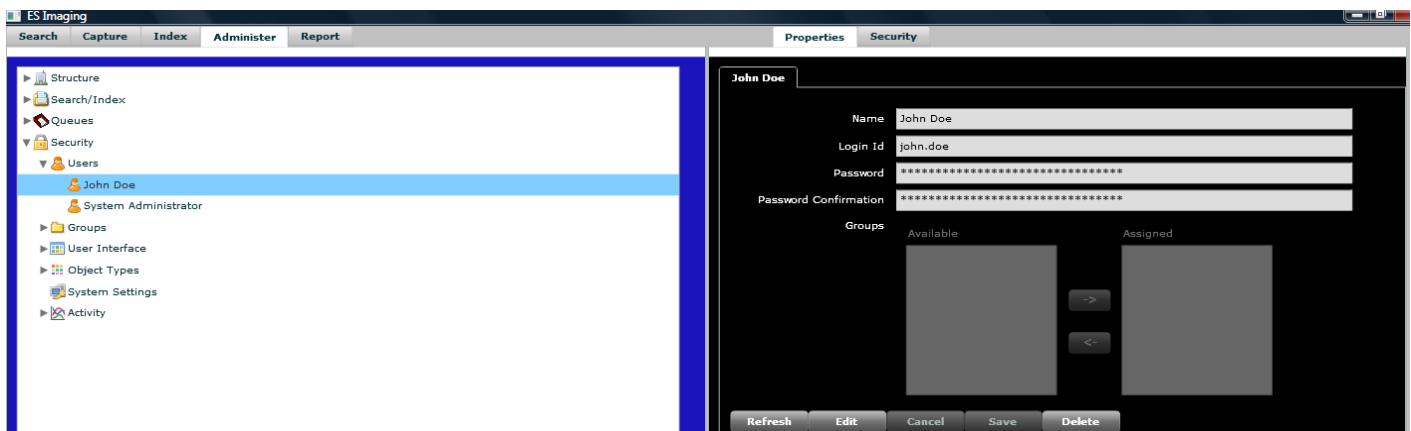
Quick Tip –
groups are used to help simplify security administration, when individuals require identical security levels.



Adding Users to Groups

See “*Preparation for Security Administration*” section, “*Group Permissions*” subsections for guidance.

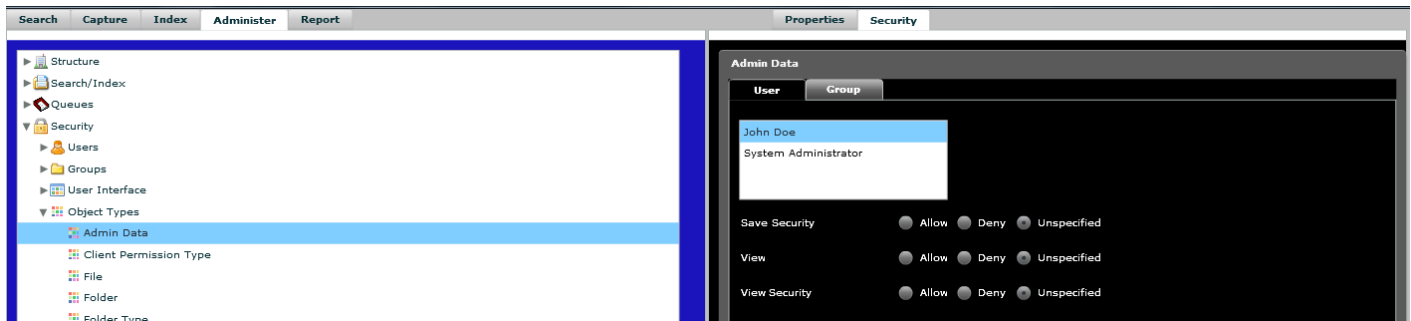
- Within the “Administer” tab, expand “Security”
- Expand “Users”
- Right click on the user to be added to a group(s)
- Click on “Edit”
- Select the groups to assign to the user
- Click on the right arrow
- Click on “Save”
-



Setting Object Type Permissions

See “*Preparation for Security Administration*” section, “*Permission Levels*”, “*Object Type Permissions*” and “*Permission Hierarchy*” subsections for guidance.

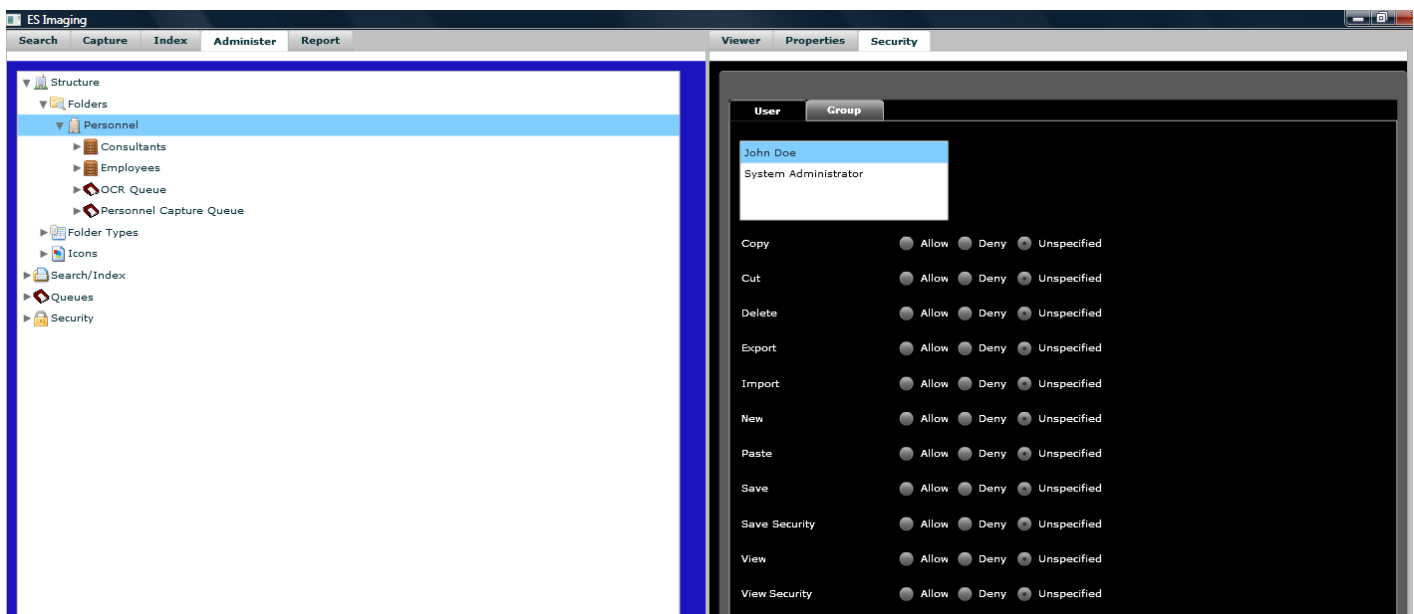
- Within the “Administer” tab, expand “Security”
- Expand “Object Types”
- Click on the appropriate “Object Type”
- Click on the “Security” tab
- Select the appropriate user(s) or group(s)
- Click on “Edit”
- Select the “allow”, “deny” or “unspecified” permission
- Click on “Save”



Setting Object Permissions

See “*Preparation for Security Administration*”, “*Permission Levels*”, “*Object Permissions*” and “*Permission Hierarchy*” subsections for guidance.

- Within the “Administer” tab, expand the tree node until the appropriate “object” (a file, folder or another specific item) is highlighted
- Click on “Security” tab
- Select the appropriate user(s) or group(s)
- Click on “Edit”
- Select the “allow”, “deny” or “unspecified” permission
- Click on “Save”



User Interface

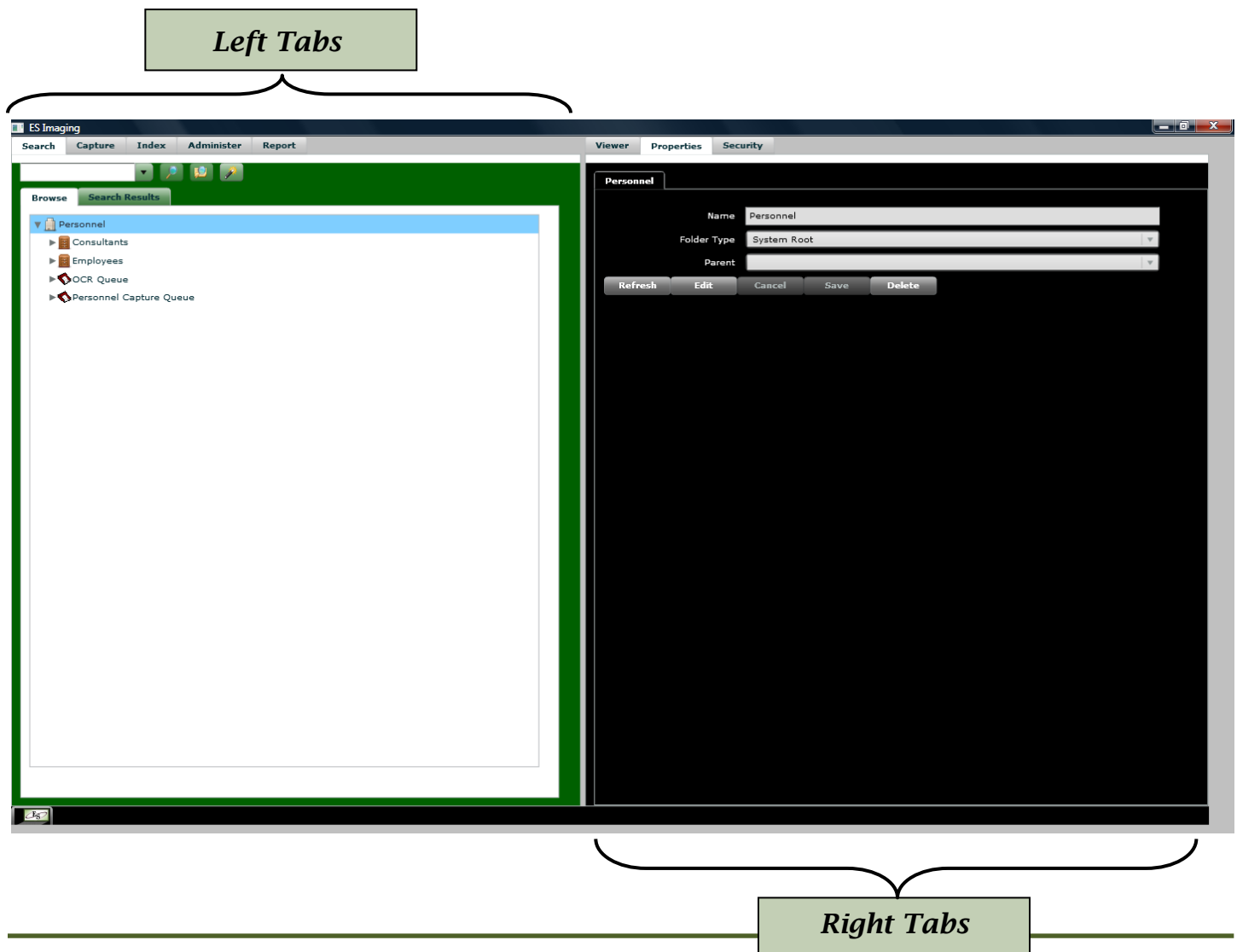
Preparation for User Interface Configuration

ES Imaging allows for customization of user interface to control what each user can view. The Administrator can hide tabs and buttons for user and/or group accounts, based on the functions an individual can perform. Additionally, icons can be imported to accurately represent various folders within ES Imaging.

User Interface Overview

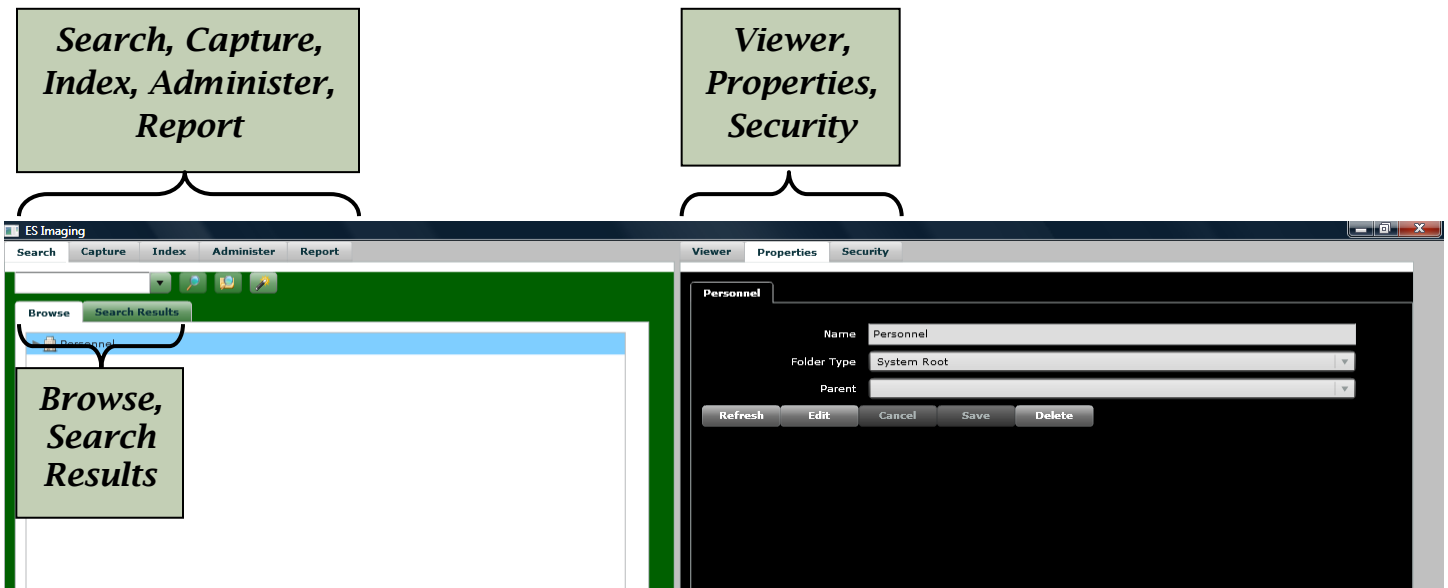
ES Imaging is designed with side by side panels that allow the user to view and navigate the tree structure while simultaneously looking at images, properties and security.

User Interface

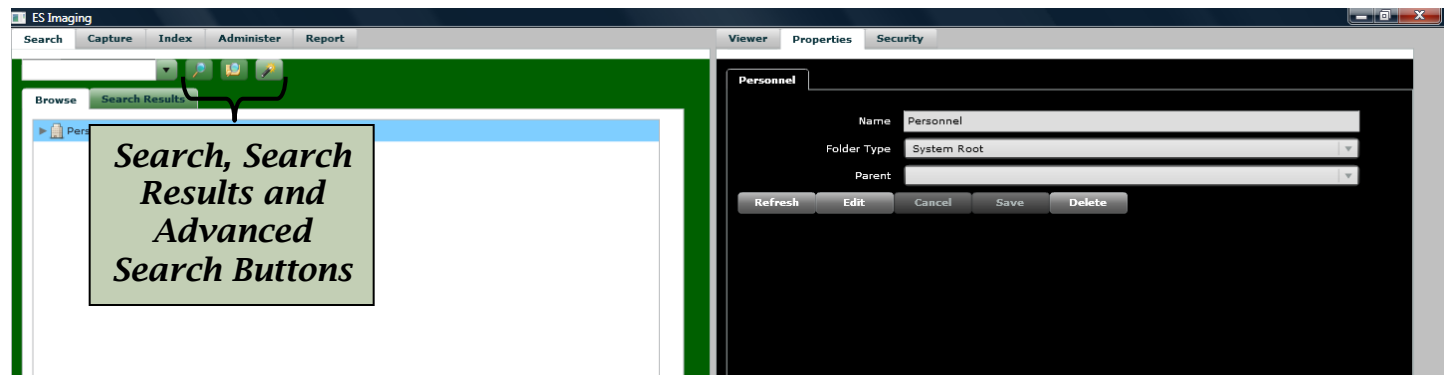


Administration Guide

Within the left and right panels, there are several *tabs* and *sub tabs*.



Within these panels, there are *buttons* that perform a variety of functions.



As mentioned within the “*Preparation for Security Administration*” section, consideration should be given as to the individual(s) performing the various functions within ES Imaging. The control of the *user interface* settings along with the permission settings will provide each user exactly what they need to perform their tasks within the system. Here are the installed *user interface* settings for the various *Preset Groups* (see the “*Preparation for Security Administration*” section, “*Preset Group Permissions*” subsection for an explanation of permissions for these groups):

- *System Administrator* – allowed to view **all tabs and buttons**.
- *Administrator* – allowed to view **all tabs and buttons**.
- *File/Folder Administrator* – allowed to view **all tabs and buttons except the capture and index tabs**.
- *Capture User* – allowed to view the **search, viewer, properties and capture tabs**.
- *Index User* – allowed to view the **search, viewer, properties and index tabs**.
- *File/Folder Viewer* – allowed to view the **search, viewer and properties tabs**.
- *Report Viewer* – allowed to view the **reports, viewer and properties tabs**.

User Interface Configuration

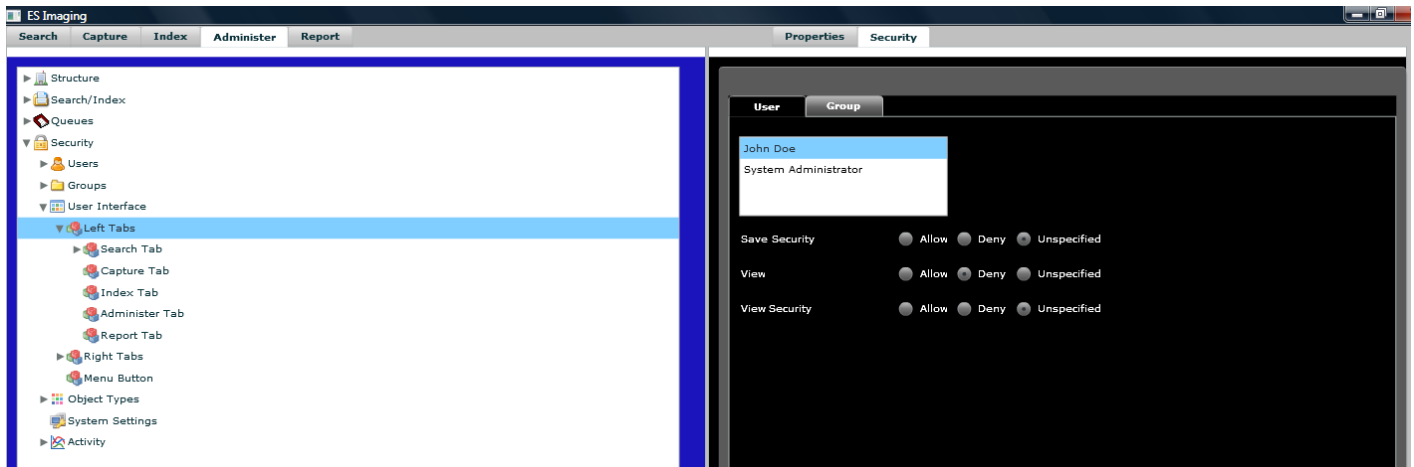
See “Preparation for User Interface Configuration” section for guidance.

Setting User Interface Permissions

- Within the “Administer” tab, expand “Security”
- Expand “User Interface”
- Click on the appropriate item
- Click on the “Security” tab
- Click on the appropriate user(s) or group(s)
- Click on “Edit”
- Select “allow” for the “view” function
- Click on “Save”

Note that if the “unspecified” permission is selected that the *user interface object type’s* permission will be used.

Quick Tip –
permissions set
at the parent
item will be used
for all children,
when
permissions are
set to
“unspecified” at
the children
level.



Workflow, Events, and Triggers

Workflow

Most businesses have a series of processes they follow in order to ensure work gets done. These processes typically involve someone being notified of some work, the individual performing the actual work, and managers approving the work. This is a typical workflow process in a nutshell. In other words, workflow in a computer system is just an abstraction of the steps taken in real life.

ES Imaging supports, not only the above scenario, but also very complicated processes through the use of its events and triggers. In other words, workflow is just a subset of events and triggers. In order to further understand how one might use workflow in ES Imaging, we will look at an example.

Suppose a college wants to review and approve students to attend. They may have a workflow process setup such as:

Prospective student's high school transcript records are scanned into ES Imaging.

Mary looks in her inbox to see if there are any new students to review.

Mary determines the student's eligibility and either denies or approves. She looks at the student's academic capabilities.

John is the next person to review the student's eligibility. John looks at the student's financial capabilities and denies or approves.

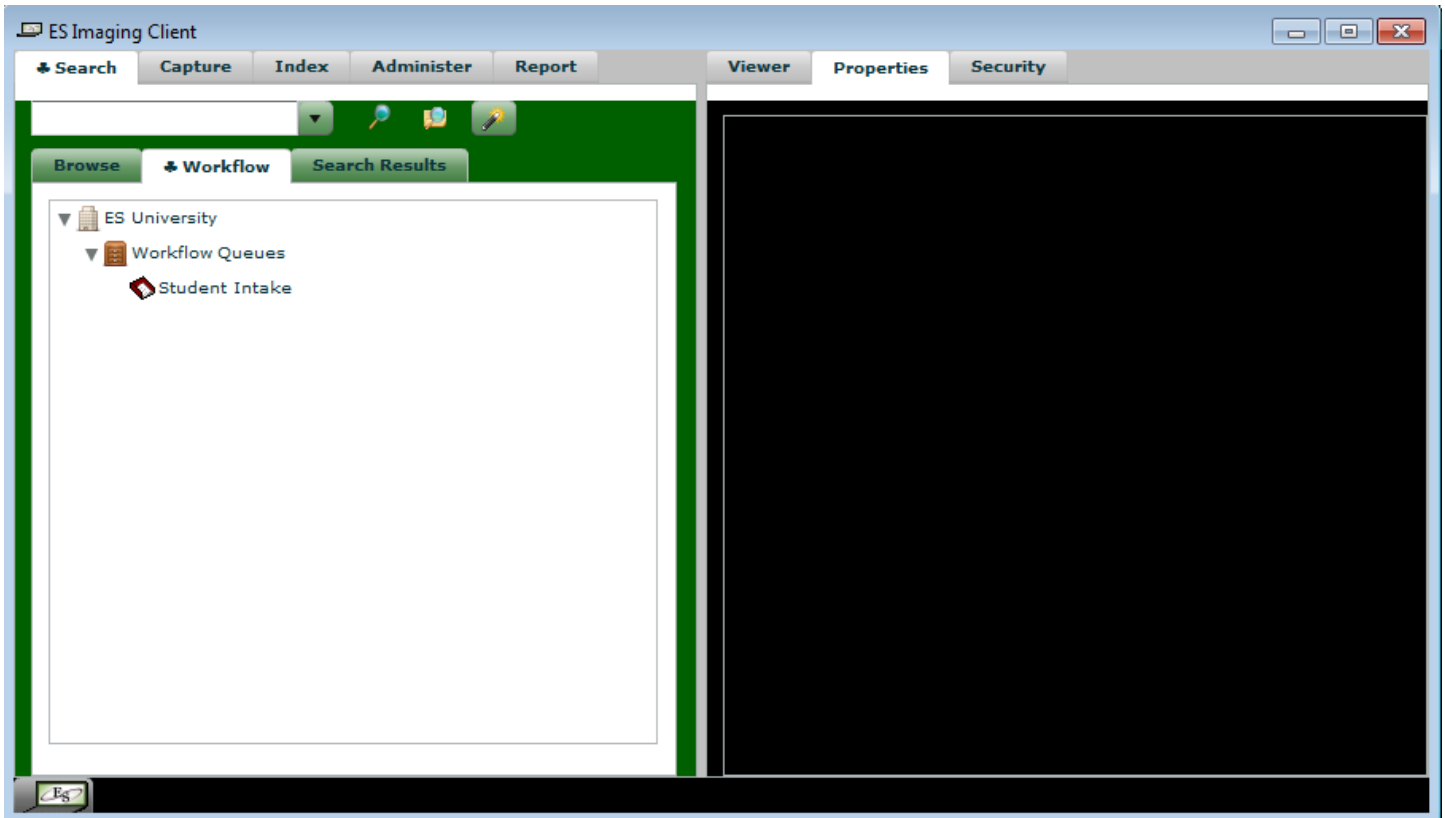
If the student was approved, Carol sends out an acceptance letter via mail.

If the student was denied (at any point in the process), Brian sends out a rejection letter via mail.

How exactly would this situation be handled in ES Imaging? We still have a couple concepts to discuss before the actual implementation.

Workflow Queue

The “inbox” discussed above is actually a workflow queue in ES Imaging. These are simply folders that have their “Workflow Queue” property set to “true”. Workflow queues show up in the special “Workflow” tab when a user logs into the system (see the figure below).



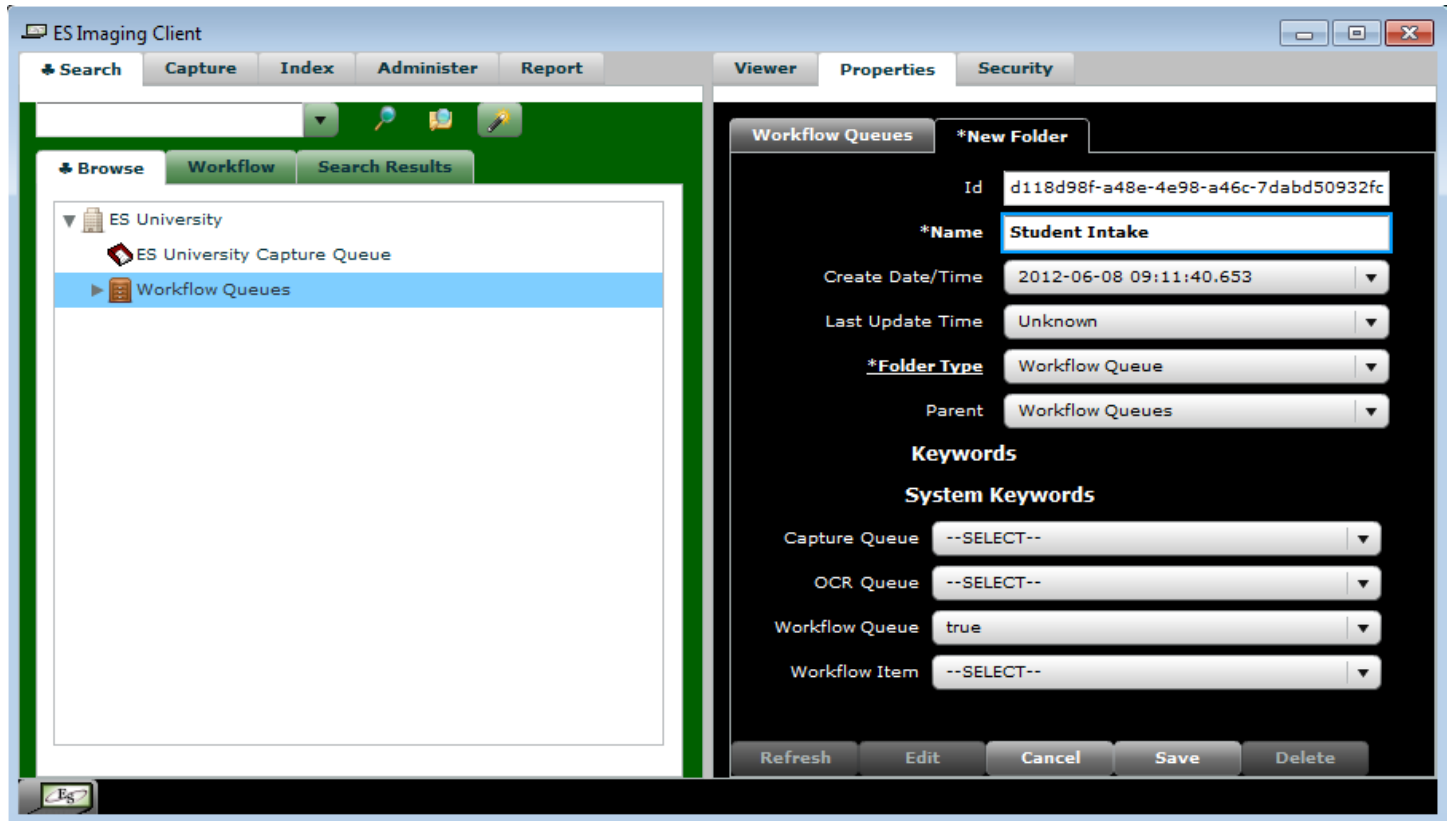
A user can have access to 0, 1, or many workflow queues based on security. Conceptually, there are 2 types of workflow queues: Group or Personal. A group queue would be one that is intended to be viewed by multiple people. A personal queue would be one that only a given person would take care of. The system doesn't really distinguish between the two, only the administrators do based on how they set security and events/triggers. Some businesses may decide to only have group queues (larger businesses typically). Smaller businesses, where there is only 1 person doing the work, may decide they'd rather have their workflow queues be 1 for 1 per user. They may know that Brian always handles denials, so they just route every denial to Brian's queue.

Again, there is no real distinction in ES Imaging, so the way an administrator would implement group vs personal is with the name.

Examples of group queues would be: Student Academic Approval, Student Financial Approval, Student Academic Denial, Student Financial Denial, and Student Acceptance.

Creation of a Workflow Queue

- Select a folder you wish to make a workflow queue in the tree.
- Select the “Properties” tab.
- Click the “Edit” button.
- Change the “Workflow Queue” drop down list to “true”, as indicated in the figure below.



Workflow Item

Now that we have queues for items to appear in, how does something get there? This happens through a few types of events, namely: Create Workflow Item and Move Workflow Item.

Conceptually, the Create Workflow Item event should fire when it has been indexed in the system (sits at its final resting place). The reason it normally happens then is all the appropriate keyword values would be filled in by the person doing the indexing. They may fill in keywords such as: First Name, Last Name, Home Phone, Address, etc. All this information is important and useful for the people involved in the workflow processes to get their work done in the real world.

So how does the “Create Workflow Item” event work? It creates a “packet”, around the item that was indexed. The packet goes to 1 or more workflow queues (determined by the administrator). In the case above, the packet would be created around a high school transcript that was scanned. The reason this packet is created is that the related workflow keywords (an Approval Status, and Approved By in this case) are tied to it instead of the real document. People just looking at the real high school transcript don’t care about workflow information. In fact, it may confuse them. By tying the Approval Status and Approved By to this packet, the clutter can be eliminated.

Another important thing to point out above the packet is that it allows for multiple items to be bundled together. If there is a case where several things need to be reviewed all at once, this is possible since the packet can have 1 or more child folders. The administrator can setup events/triggers to create this situation.

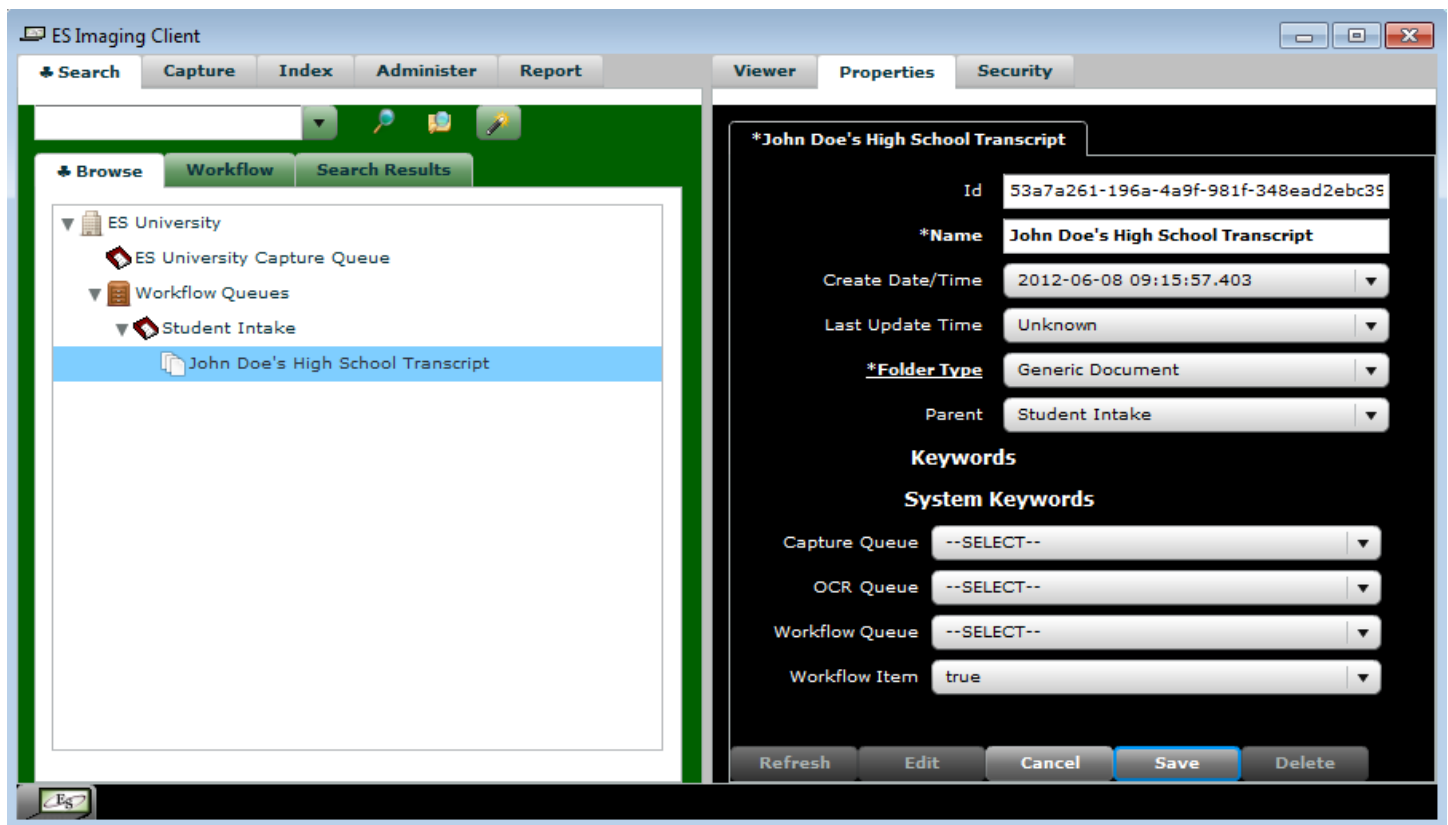
The “Move Workflow Item” event should fire when an item is ready to move on to a different workflow queue. In the scenario above, it may be going from “Student Academic Approval” to “Student Financial Approval”.

It’s also worth mentioning that the only thing that distinguishes a workflow item from a regular item is that it has the properly “Workflow Item” set to “true”. This allows the events and triggers to move the item to various workflow queues.

Creation of a Workflow Item (Manually)

- Select a folder you wish to make a workflow item in the tree.
- Select the “Properties” tab.
- Click the “Edit” button.
- Change the “Workflow Item” drop down list to “true”, as indicated in the figure below.

NOTE: This technique is a “manual” way to force a folder to be a workflow item, and is certainly not the norm. Typically you will use events and triggers via the “Create Workflow Item” event.



Creation of a Workflow Item (With Events)

This is the normal way to create workflow items. Typically the process is setup to occur after indexing is complete.

Refer to the “*Create Workflow Item*” event type under the “*Event Types*” section of the manual.

Events and Triggers

As mentioned earlier, workflow is just a subset of the events and triggers in the system. There are many types of events an administrator can utilize to perform business functions. Conceptually, there are several categories:

- 1) Notification of data
- 2) Rejection/Approval of data
- 3) Modification of data
- 4) Deletion of data
- 5) Creation of data
- 6) Searching for data
- 7) Auditing of data
- 8) Moving of data
- 9) Processing of data

As you can see, all of them involve data in one way or another. That leads us to the type of events that exist in ES Imaging. The following section is a brief description for each of them. We will start with the simpler ones and work our way up to the more complex ones.

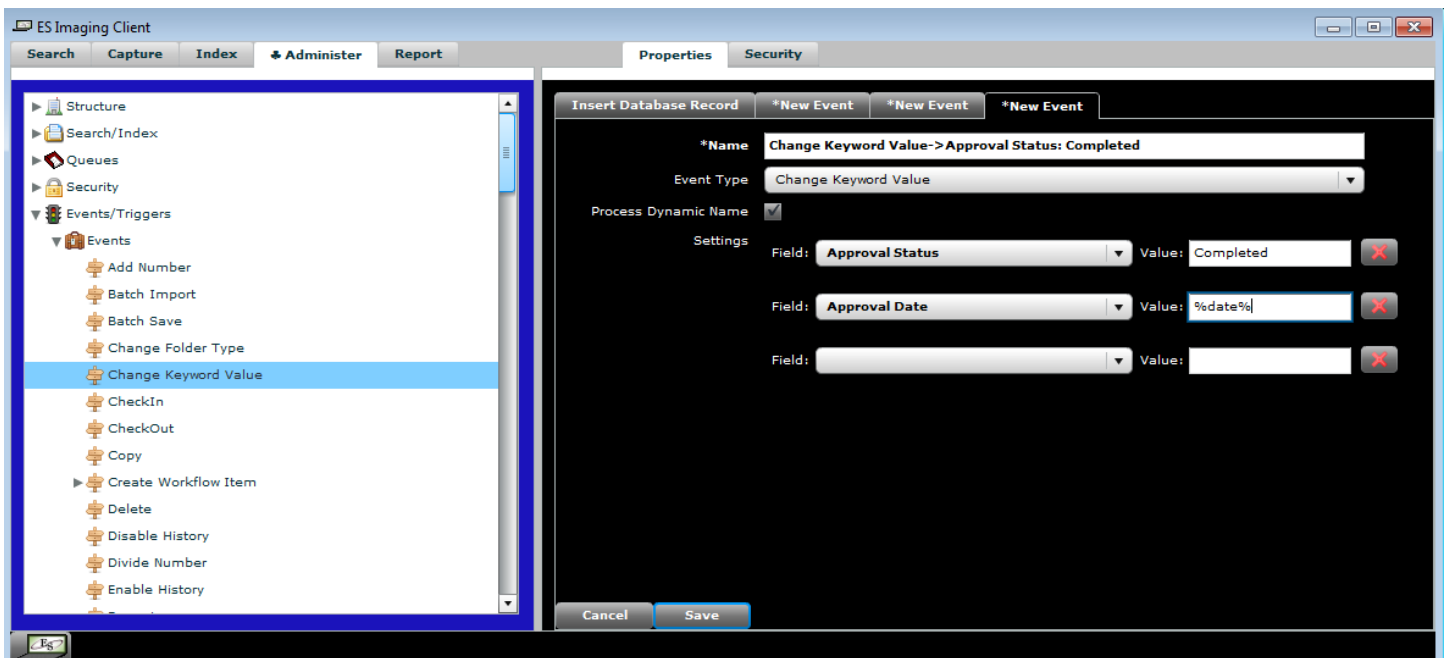
It is important to note that most event types are designed to work off an Action Trigger (see the “Triggers” section). Some exceptions are: Sleep, Send Email, Throw Error Message, Batch Import, Batch Email Import, Insert Database Record, Search, Batch Save, Execute Multiple Events, Synchronize Records, and Replicate Volume Engine. These can be triggered via an action or schedule. Since action triggers require a context (object that an action just happened on), the event types described below in the “Event Types” section will indicate what type of objects. Sometimes it is specific to only folders or files, other times it can be any type of object (folder type, folder, user, keyword type, etc).

There is one more thing to mention before discussing event types. Dynamic tags (refer to the “Dynamic Tags” section) can be utilized for most of the parameters in order to give even more power to administrators in ES Imaging.

Event Types

- 1) Delete – Deletes the supplied objects.
- 2) Copy – Copies the supplied objects to a given folder.
 - a. Destination Folder – An id or unique name of a folder in the system that will receive the copied object(s).
 - b. Source – An id of an object to copy. If blank, the supplied object in context will be copied.
 - c. Fire Events – If checked, save action triggers may fire during the copy event. Use caution to avoid recursive firing of copying folders with many children.
- 3) Move – Moves the supplied objects to a given folder.
 - a. Destination Folder – An id or unique name of a folder in the system that will receive the copied object(s).
- 4) Rename – Renames the supplied object to a given value.
 - a. New Name – The new name that will be utilized
- 5) Move Volume – Moves all the files on the supplied volume to a given volume. Only the System Administrator can save this event type.
 - a. Destination Volume Name – The name of the destination volume to move all the files to. This must be one of the volume names in the “es_volume” table. Only System Administrators can setup this type of event.
- 6) CheckIn – Checks in the supplied object.
- 7) CheckOut – Checks out the supplied object to a given user.
- 8) Enable History – Enables history on the supplied folder.
- 9) Disable History – Disables history on the supplied folder.
- 10) Add Number – Adds the supplied number to a keyword value on a folder or file.
 - a. Process Dynamic Name - If checked, causes the system to recalculate the name on the supplied object after the keyword value(s) have been changed.
 - b. Settings – The keywords to change.
 - i. Field – The specific keyword value to add to.
 - ii. Value – The number to add.
- 11) Divide Number - Same as “Add Number” except it performs division.
- 12) Multiply Number – Same as “Add Number” except it performs multiplication.
- 13) Subtract Number - Same as “Add Number” except it performs subtraction.
- 14) Encrypt – Encrypts the supplied file with a given password.
 - a. Password – The password you wish to utilize to encrypt the file.
 - b. Verify Password – Must match the “Password” parameter. This is just to verify there wasn’t a typo.
- 15) ReEncrypt – Re-encrypts the supplied file with a given password (changes the password). It will only do so if the provided “Old Password” is correct and if the file was already encrypted.
 - a. Old Password – The existing password for the file to re-encrypt.

- b. New Password – The new password you wish to utilize to re-encrypt the file.
 - c. Verify New Password – Must match the “New Password” parameter. This is just to verify there wasn’t a typo.
- 16) Change Folder Type – Changes the folder type on the supplied folder.
- a. Folder Type Name – The specific folder type to change the supplied folder to.
- 17) Change Keyword Value – Changes a given keyword value on the object. See the figure below.
- a. Process Dynamic Name – If checked, causes the system to recalculate the name on the supplied object after the keyword value(s) have been changed.
 - b. Save Value – If checked, causes the updated keyword value to be saved in the database. If false, the value is merely temporarily updated. Most of the time you want this true. An example where you may want it to be temporary is when viewing encrypted keywords/fields. In most cases, you want the user to be able to see the decrypted value, but keep it stored as an encrypted string.
 - c. Change For Parent – If checked, causes the updated keyword value to be set on the object’s parent, instead of the object itself. This is useful if a file is the source but the parent folder has the desired keyword type.
 - d. Settings – One or more keywords can be specified to change.
 - i. Field – The keyword to change.
 - ii. Value – The new value you want to set the keyword to. Dynamic tags, like “%date%” used in the figure below, can be used to make evaluations at the time an event fires. Refer to the “Dynamic Tags” section for more information.



- 18) Process Dynamic Name – Causes the system to recalculate the dynamic name on the supplied object.
- 19) Flatten Image – Causes the supplied File's markups to be permanently embedded in the image. It will only work on Image (JPEG, TIFF, PNG, BMP, etc.) files. This cannot be undone, so be careful when utilizing this event type. This is useful for redaction to be permanent, or other markings to be printable, exportable, etc.
 - a. Create New Version – If checked, a new version will be created of the flattened image.
 - b. Version Name – The name for the new version to create (if 'Create New Version' is checked).
- 20) Rotate Image – Causes the supplied File to be rotated by a specified number of degrees.
 - a. Degrees – The number of degrees to rotate the image (integer).
- 21) Scale Image – Causes the supplied File to be scaled/zoomed.
 - a. Percentage – The percentage to scale the image by. This must be greater than 0 to perform scaling.
 - b. Max Pixels – An optional parameter that forces the image to a maximum pixel size based on which dimension is larger (width or height). The size forcing only occurs if the image is larger than the specified max pixels. The 'Percentage' parameter is first used to scale the image before applying 'Max Pixels'. If 0, no forcing will occur.
- 22) Propagate Security – Causes security to propagate on the supplied object.
 - a. Propagate to Children – If checked, the security is propagated down to all children (applies only to a folder).
 - b. Override Inherited Security – If checked, the values provided in the 'Permissions' section will be applied, instead of the object's parent. Otherwise, the parent's security is propagated.
 - c. Merge Existing Security – If checked, the already existing permissions on the object will be merged with the provided 'Permissions'. This also requires the 'Override Inherited Security' to be checked. The provided security will override any previous security, except when the provided has 'Unspecified' set for a given permission type. For example: Previous Permissions=Group A – View = Deny, Group A – Save = Allow merging with Provided Permissions=Group A – View = Deny. The result (after propagation) would be Group A – View = Deny, Group A – Save = Allow.
 - d. Permissions – This section allows an administrator to set any type of permission for any user/group. However, the provided permission types will only apply if the object(s) in context have that permission type. For example, a 'Folder' does not have the 'Execute' permission type, so setting that would have no effect in this example.
- 23) Sleep – Causes the system to pause for the supplied amount of milliseconds. This can be useful when you have multiple events chained together and you need a pause for whatever reason.
 - a. Milliseconds – The number of milliseconds to wait after the event has fired.

- 24) OCR – Causes the supplied folder or file to be OCR processed. This event automatically happens by default if a folder or file is saved in an OCR queue. This is a system level event type and should not be altered.
- 25) Send Email – Sends an email to 1 or more email addresses.
- a. From – The email address that will appear in the “From” field when received by others.
 - b. Subject – The subject of the email.
 - c. Message – The message/body of the email.
 - d. To – One or more email addresses to receive the email.
 - e. CC – Optional email addresses to carbon copy.
 - f. BCC – Optional email addresses to blind carbon copy.
- 26) Write to Log – Writes a text message to a log file. Only the System Administrator can save this event type.
- a. Log File Name – The full path to the log file.
 - b. Text to Write – The text message to append.
- 27) Execute Program – Executes a program on the server. Only the System Administrator can save this event type. Also, the ‘allowProgramExecution’ flag must be set to true in the Event Monitor Engine’s XML file in order to execute a program.
- a. Program File Name – The full path to the executable.
 - b. Parameters – These are the parameters/arguments to provide to the program. This is optional.
- 28) Call URL – Calls the provided URL, and returns the results. It can be any http or https address that the server can reach. Only the System Administrator can save this event type. Also, the ‘allowURLCalling’ flag must be set to true in the Event Monitor Engine’s XML file in order to call a url.
- a. URL – The URL to call. Examples: ‘http://www.mydomain123.com/status.txt’ or ‘https://www.securesite456.com/UpdateEmployeeServlet’.
 - b. URL Type – If left blank, the default is an HTTP Post. Otherwise, ‘get’ can be provided to perform an HTTP Get instead of a Post.
 - c. Login Id – An option authentication login id.
 - d. Password – An optional authentication password.
 - e. Parameters – These are the names of the parameters/arguments to provide to the URL. Examples: ‘First Name’, ‘Last Name’.
 - f. Values – These are the matching values for the parameters/arguments. Examples: ‘John’, ‘Doe’.
- 29) Throw Error Message – This causes an error message to be generated. The purpose of this is to abort/prevent a function from taking place. If multiple events were chained together to fire, this will abort the proceeding events.
- a. Message – The message that the user will see (if an Action Trigger fires the event) or that will appear in the transaction history (if a Schedule Trigger fires the event).

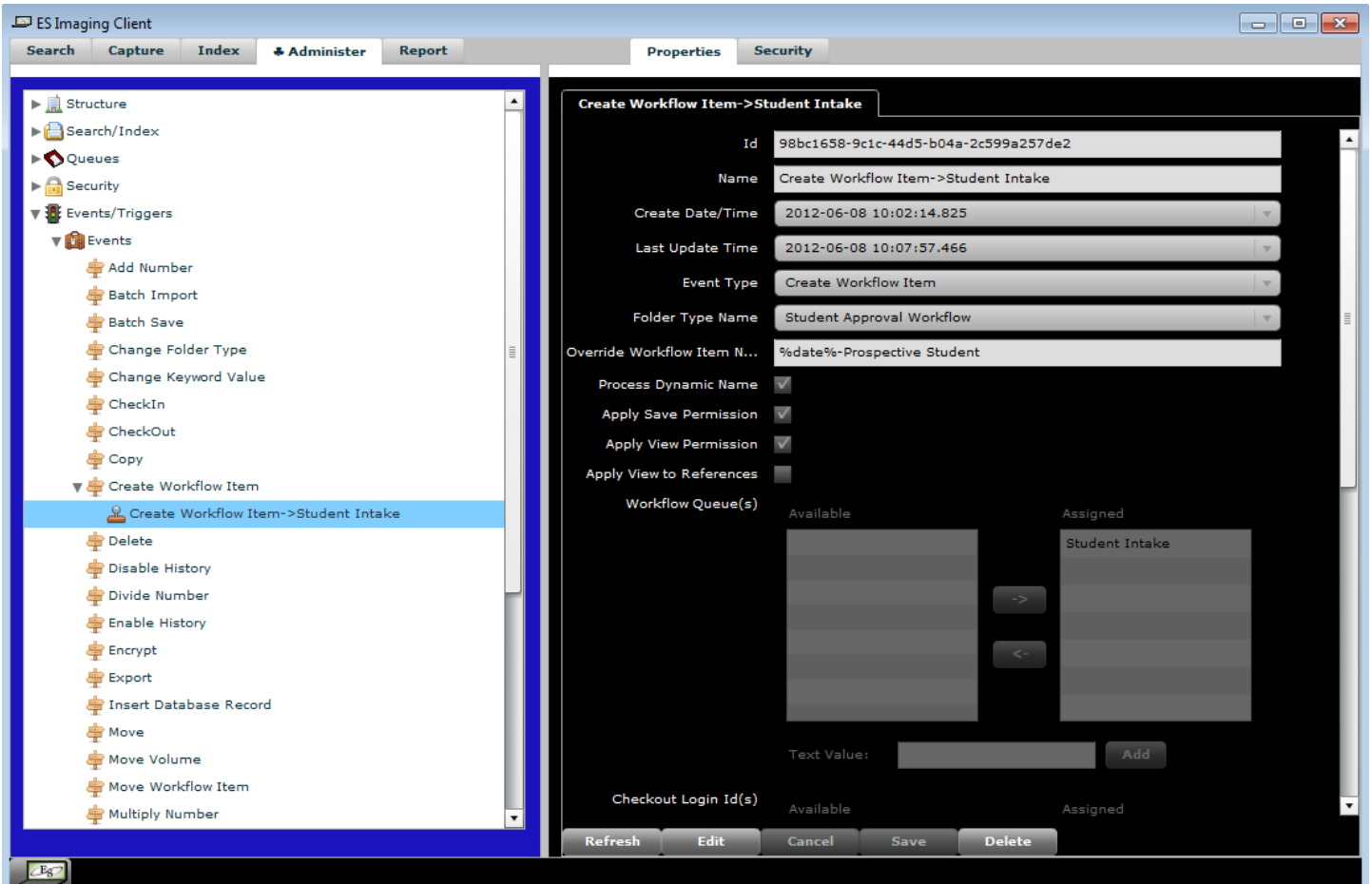
- 30) Export – Causes the supplied files to be exported to a directory on the server. Only the System Administrator can save this event type.
- a. Destination Path – The directory on the server to receive the supplied files.
 - b. Append Folder Hierarchy – If checked, the folder structure in ES Imaging will be appended to the destination path. If unchecked, all files will be dumped directly in the destination path.
 - c. Overwrite Existing Files – If checked, existing files with the same name will automatically be overwritten.
- 31) Batch Import – Causes all the files in a given directory on the server to be imported into a specified folder in ES Imaging. Only the System Administrator can save this event type.
- a. Source Path – The directory on the server to process.
 - b. Destination Folder – The id or unique name of a folder in ES Imaging to import into.
 - c. Abort On Single File Import Failure – If checked, and any file fails to import, the entire batch import will abort.
 - d. Create Folder Hierarchy – If checked, the directory structure utilized on the server will be imported as “Unknown” folders in ES Imaging.
 - e. Volume Engine Name – If specified, this specific Volume Engine will be used during the import. Otherwise the “Default” volume engine will be used.
 - f. Create Shortcut To Source – If checked, the named Volume Engine will be utilized (or created if it doesn't exist) and its “Mount Name” will be pointing to the source path. Basically this means the files are not directly copied, just the database records created pointing to the original files. This option is useful when you already have existing files on a server (local or remote) in a given hierarchy, and you want to transition over to using ES Imaging for storage. Users can still use the server's file system since changes made to the files from within ES Imaging will be reflected in the original location.
 - g. Allow Deletes From Source – If checked (“Create Shortcut To Source” must be checked as well), a deletion performed from ES Imaging will delete the actual file on the source drive. Please ensure you want this behavior before enabling this flag.
 - h. Phantom Write – If checked, the files aren't actually imported into ES Imaging, but the folders and file stubs are created. This can be helpful when trying to test an import and check the hierarchy/structure.
- 32) Extract Email – The purpose of this event type is to process already saved email files that one has produced from an email client, such as Outlook. It causes the provided file (must be a '.eml' file structure) to have its email contents extracted into attachments and keyword values. A new folder of type 'Email' is created in the same folder as the file being processed. This new folder will have the system “email” keyword values filled (subject, from, to, message body, etc). The new folder will also contain any attachments of the email.
- 33) Evaluate – The purpose of this event type is to replace all the tags in the supplied 'Value' field. This event type is executed any time an 'Evaluate' tag is utilized.

- a. Value – All the tags supplied in the string will be evaluated. The value returned can then be used in subsequent events with the 'previousResult' field type. If the event is fired from a tag, the 'Value' must be in 'EVENTNAME,%parameter[1]%,%parameter[2]%,...' format, where 'n' is the index number of the parameters provided from the 'Evaluate' tag. The 'EVENTNAME' argument must be the unique name of the Evaluate Event to fire.
- b. Example:
 - i. If the tag was: %evaluate('My Eval','john','doe')%, and the 'Value' of the 'My Eval' Evaluate event was 'the name is: %parameter[2]%, %parameter[1]%', the end result of the tag replacement would be: 'the name is doe, john'.

- 34) Batch Email Import – Causes all the emails in a given set of email boxes (or inbox) on a POP email server to be imported into a specified folder in ES Imaging.
- a. Email Engine – This can only be utilized by the System Administrator. It allows a specific Email Engine to be utilized instead of the explicit parameters below. Typically this will be blank.
 - b. Destination Folder – The id or unique name of a folder in ES Imaging to import into.
 - c. Abort On Single File Import Failure – If checked, and any email fails to import, the entire batch import will abort.
 - d. POP Server – The name or IP address of the POP email server.
 - e. Port – The port to connect to on the POP server.
 - f. Use TLS – If checked, a secure connection will be utilized to retrieve the emails (if available).
 - g. User Name – The user name to use when reading from the POP server.
 - h. Password – The password to use when reading from the POP server.
 - i. Phantom Write – If checked, the files aren't actually imported into ES Imaging, but the folders and file stubs are created. This can be helpful when trying to test an import and check the hierarchy/structure.
 - j. Delete After Import – If checked, the email on the POP server will be permanently deleted after successfully being imported into ES Imaging. Use extreme caution with this!
 - k. Create Sub-Folders – If checked, sub-folders will be created that have the email box name. For example, 'Inbox' may be created in the 'Destination Folder', and then all the emails will go inside 'Inbox'.
 - l. Save Raw Email – If checked, the full bytes of the email will be saved with an attachment called 'raw.eml'. This is useful if you ever want to be able to recreate/export the original email in some external program.
 - m. Mail Folders – If specified, the supplied email boxes will be read on the POP server. If excluded, the default inbox will be read.

- 35) Batch Email Delete – Causes all the emails in a given set of email boxes (or inbox) on a POP email server to be deleted if any of the provided criteria is matched.
- a. Email Engine – This can only be utilized by the System Administrator. It allows a specific Email Engine to be utilized instead of the explicit parameters below. Typically this will be blank.
 - b. Abort On Single File Import Failure – If checked, and any email fails to import, the entire batch import will abort.
 - c. POP Server – The name or IP address of the POP email server.
 - d. Port – The port to connect to on the POP server.
 - e. Use TLS – If checked, a secure connection will be utilized to retrieve the emails (if available).
 - f. User Name – The user name to use when reading from the POP server.
 - g. Password – The password to use when reading from the POP server.
 - h. Phantom Delete – If checked, the emails aren't actually deleted, but the status returned specifies how many would have been deleted. This can be helpful when trying to test and make sure you have the deletion criteria correct.
 - i. Mail Folders – If specified, the supplied email boxes will be read on the POP server. If excluded, the default inbox will be read.
 - j. Safe Email Addresses – If specified, the supplied "From" will be checked for any "like" values. If a match is found, the email is automatically flagged as safe and will not be deleted.
- 36) Insert Database Record – Inserts a record into the specified table and columns of the database. This is useful for auditing and recording of data during system usage. For example, you may want to know any time a particular file is viewed, and who did it. Or, you may want to insert a record showing any time an email event is fired. Only the System Administrator can save this event type.
- a. Table Name – The name of the table to insert into.
 - b. Columns – The columns to insert values into.
 - c. Values – The values to insert into the supplied columns.

- 37) Create Workflow Item – Creates a workflow item (packet) out of a supplied folder or file. The supplied folder or file will appear as a child (shortcut) under this newly created workflow item. The original location of the folder or file will not be changed since it is a shortcut. The workflow item is created in the specified workflow queue(s). They item can optionally be automatically checked out to specified people. Finally, keyword values can automatically be set on the newly created workflow item. See the figure below for an example.



The follow explains the specific parameters:

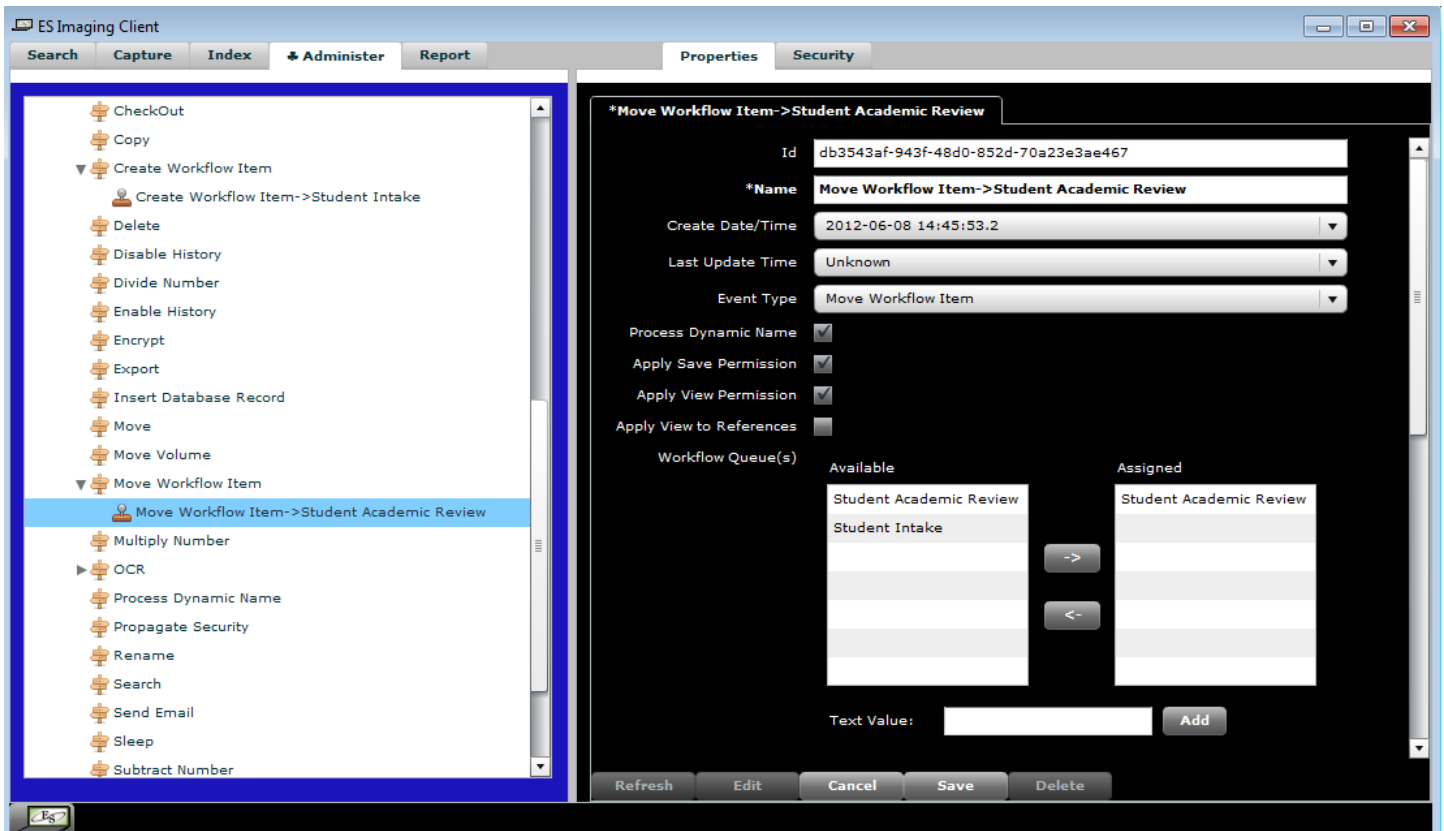
- Name – The name is VERY important when setting up workflow related events. The reason is it can get confusing very quickly because there are so many paths workflow items may take during a workflow process. Typically, you will want to use some sort of naming convention, like “Create Workflow Item->XXX” where XXX is the workflow queue or step in the process.
- Folder Type Name – The specific folder type to use for the workflow item. Typically this will be a type that has keywords such as “Approve Date”, “Approval Status”, “Approved By”, etc. Any type of information specifically related to the workflow process should be assigned to this folder type.

- c. Override Workflow Item Name – This will override any dynamic name set on the specified folder type. You can use any of the tags listed in the “Dynamic Tags” section of the manual. If used, it will typically be the create date and useful information such as the type of workflow process.
- d. Process Dynamic Name – If checked (and if no “override workflow item name” is supplied), the server will utilize the dynamic name on the folder type when the item is created. Any “Initial Keyword Values” specified will also process correctly, if they are in the dynamic name.
- e. Apply Save Permission – If checked, any users able to view the parent workflow queue will automatically be able to “Save” the item as well. This is a convenient way to have security be dynamic for the items so people can change status or cause other triggers to fire in order to move the workflow item along in the workflow process.
- f. Apply View Permission – If checked, any users able to view the parent workflow queue will automatically be able to “View” the workflow item as well.
- g. Apply View to References – If checked, any users able to view the parent workflow queue will automatically be able to “View” the original documents/files underneath the workflow item. It is important to note that this basically gives the user rights to view anything underneath this workflow item, regardless of where it is saved in the system. These rights will continue to stick even after the workflow item is removed or moved to a different workflow queue. It is up to the administrator to setup a new trigger or process to strip this security from the user when appropriate. Basically, this is an easy way to give view rights to the original documents/files for any user able to view the workflow item.
- h. Create For Each Child – If checked, the child folders/files directly under the source folder will get a separate workflow item created. This happens for each workflow queue specified as a destination. In other words, if there are 6 files as children and 2 workflow queues as destinations, a total of 12 workflow items will be created. This is useful if you want to split a folder into pieces for workflow. Typically this will be when each file will be worked on by different people or the files will move through workflow in parallel. If this property is not checked, only 1 workflow item for the source folder will be created per workflow queue destination.
- i. Workflow Queues – Any “Assigned” workflow queues will get a copy of the new workflow item. Typically you will only want to create a workflow item in 1 queue at a time, but ES Imaging does allow for multiples if desired. It’s important to note these are COPIES and not REFERENCES to the same workflow item. Each workflow item will act independently with keyword values and workflow related information.
- j. Checkout Login Ids – Any “Assigned” checkout login ids will automatically have the newly created workflow item checked out to them. This is optional and only needed if you want the workflow item to automatically be assigned to a specific person. This is particularly useful if you are using personal workflow queues (only 1 person monitoring

the queue) as it will save a step for the user having to check out the item. This also guarantees that others can't check out the item in a group workflow queue situation. If you are using multiple "Workflow Queues", you must either have only 1 person specified here, or a 1 to 1 ratio for each queue. In other words, if you had 3 workflow queues, you could either specify 1 person (John Doe for example), or you could specify 3 people (John Doe, John Doe, and Mary Sue). The order of the "Workflow Queues" must match the order of the "Checkout Login Ids" if specifying multiple people to check out to.

- k. Initial Keyword Values – This allows you to set specific keyword values on the supplied folder type when the workflow item is created. For example, you may want to set the "Approval Status" to "Pending". To do this, you would select "Approval Status" from the "Field" drop down list and then type in a value of "Pending" in the "Value" text box. You can use any dynamic tag from the "Dynamic Tags" section of the manual as well.

38) Move Workflow Item – Similar to the “Create Workflow Item” event type, except the supplied workflow item is actually moved to the supplied workflow queue(s). See the figure below.

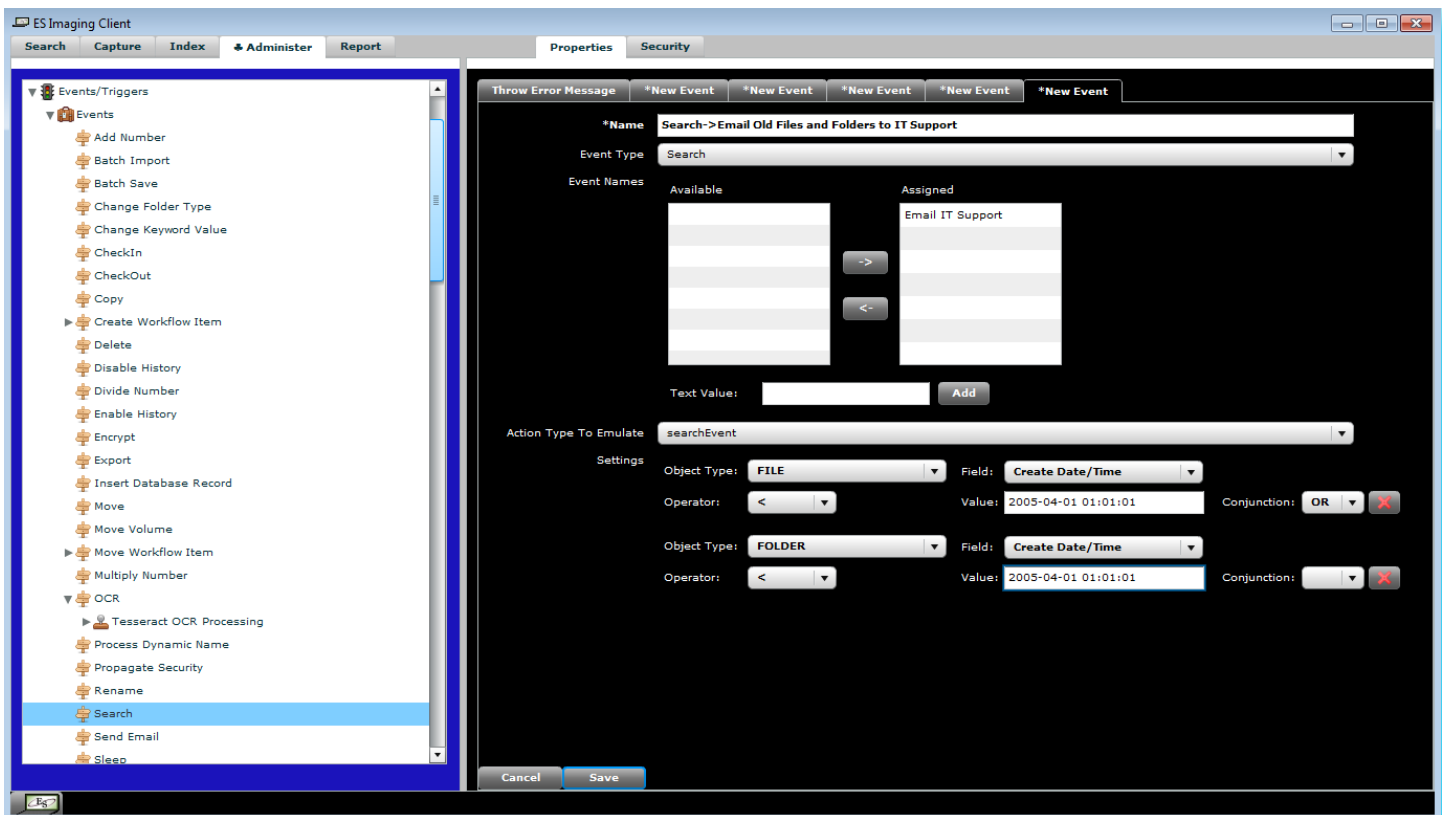


- Name – Again, the name is VERY important when setting up workflow related events. The reason is it can get confusing very quickly because there are so many paths workflow items may take during a workflow process. Typically, you will want to use some sort of naming convention, like “Move Workflow Item->XXX” where XXX is the workflow queue or step in the process.
- Process Dynamic Name – If checked, the server will utilize the dynamic name on the workflow item when the event fires. Any “Change Keyword Values” specified will also process correctly, if they are in the dynamic name.
- Apply Save Permission – If checked, any users able to view the parent workflow queue will automatically be able to “Save” the item as well. This is a convenient way to have security be dynamic for the items so people can change status or cause other triggers to fire in order to move the workflow item along in the workflow process.
- Apply View Permission – If checked, any users able to view the parent workflow queue will automatically be able to “View” the workflow item as well.
- Apply View to References – If checked, any users able to view the parent workflow queue will automatically be able to “View” the original documents/files underneath the workflow item. It is important to note that this basically gives the user rights to view

anything underneath this workflow item, regardless of where it is saved in the system. These rights will continue to stick even after the workflow item is removed or moved to a different workflow queue. It is up to the administrator to setup a new trigger or process to strip this security from the user when appropriate. Basically, this is an easy way to give view rights to the original documents/files for any user able to view the workflow item.

- f. Workflow Queues – If only 1 “Assigned” workflow queue is specified, the workflow item will be moved. Any queues beyond the 1st “Assigned” workflow queues will get a copy of workflow item. Typically you will only want to move a workflow item to 1 queue at a time, but ES Imaging does allow for multiples if desired. Again, it’s important to note the 1st queue gets the original workflow item moved to it and any additionally assigned queues get COPIES of the same workflow item.
- g. Checkout Login Ids – Any “Assigned” checkout login ids will automatically have the workflow item checked out to them. This is optional and only needed if you want the workflow item to automatically be assigned to a specific person. This is particularly useful if you are using personal workflow queues (only 1 person monitoring the queue) as it will save a step for the user having to check out the item. This also guarantees that others can’t check out the item in a group workflow queue situation. If you are using multiple “Workflow Queues”, you must either have only 1 person specified here, or a 1 to 1 ratio for each queue. In other words, if you had 3 workflow queues, you could either specify 1 person (John Doe for example), or you could specify 3 people (John Doe, John Doe, and Mary Sue). The order of the “Workflow Queues” must match the order of the “Checkout Login Ids” if specifying multiple people to check out to.
- h. Change Keyword Values – This allows you to set specific keyword values on the workflow item when the item is moved. For example, you may want to set the “Approval Status” to “High School Approved”. To do this, you would select “Approval Status” from the “Field” drop down list and then type in a value of “High School Approved” in the “Value” text box. You can use any dynamic tag from the “Dynamic Tags” section of the manual as well.

- 39) Search – Performs a search for objects in the system, and then turns around and feeds the results into 1 or more other events. The primary purpose for this is for scheduled events (discussed later). This is one of the most powerful event types because it allows you to grab specific objects and feed them in to any event that can be fired by an Action Trigger. See the figure below for an example.
- Event Names – The specific event names to feed the search results into.
 - Action Type to Emulate – You will typically leave this as “searchEvent”. However, if you have already setup other Action Triggers for other events and want to reuse them, you can emulate that Action Trigger firing by changing this parameter.
 - Settings – The search criteria to use.
 - Object Type – The type of object to search on.
 - Field – The field to search on the object.
 - Operator – The type of comparison to make.
 - Value – The value to compare the field to. This can be any Dynamic Tag for even more power.
 - Conjunction – If you want to do more than one comparison, you can specify “Or” or “And”. You can even mix object types as the figure below indicates (files and folders).



- 40) Get Search Data – Similar to the ‘Search’ event type. The main difference is specific values can be obtained from the search results in order to populate drop down lists (used in conjunction with the ‘getPossibleKeywordValues’ action trigger). The results are returned in a delimited String, which in turn is processed automatically to fill drop down lists. This is also useful for obtaining a single String of various keyword values when transaction history is enabled on the action trigger. This can then be imported/parsed from another program. Note, it is possible to use this event type and perform no search whatsoever. You can simply provide ‘Values to Include’ as hardcoded values and specify nothing for the ‘Fields to Select’. Useful in case you want to force drop down lists to specific values (such as Male, Female).
- a. Fields to Select – The specific values/fields to retrieve from the search results.
 - b. Keep Results In Keyword With Name – The String result can be stored inside a keyword value if desired. Specify the name of the keyword in which to store the result. Example: ‘My Results’.
 - c. Values to Exclude – These are hardcoded or expandable tag values that you wish to exclude from the results.
 - d. Values to Include – These are hardcoded or expandable tag values that you wish to include in the results.
 - e. Include Values Before Search – If checked, the specified ‘Values to Include’ will appear toward the top of the drop down list. Otherwise, they will appear sorted.
 - f. Other Excluded Search Values – It’s possible to exclude values from the search results by specifying another search. Here you specify the name of the Get Search Data event. Any results from that search will be excluded from this event’s results.
 - g. Other Included Search Values – It’s also possible to include additional values from other Get Search Data event results. Simply specify the name of the event here.
 - h. Keep Only Mutually Included Search Values – If checked, any values from this event will only be included if they are also included in the ‘Other Included Search Values’.
 - i. Value Delimiter – The delimiter to use for the results. Note: The drop down list population will only work properly when a ‘~’ character is used.
 - j. Settings – The search criteria to use.
 - i. Object Type – The type of object to search on.
 - ii. Field – The field to search on the object.
 - iii. Operator – The type of comparison to make.
 - iv. Value – The value to compare the field to. This can be any Dynamic Tag for even more power.
 - v. Conjunction – If you want to do more than one comparison, you can specify “Or” or “And”. You can even mix object types as the figure below indicates (files and folders).

- 41) Batch Save – Takes the supplied XML and saves the objects in the system. This is extremely powerful but very complicated as well, since it allows any type of object to be created or modified. Only the System Administrator can save this event type.
- a. Settings – This is the XML structure to save. Everything must be precisely specified or the save(s) will fail.
- 42) Random Keyword Value – Selects 0 to many random keyword values based on a percent chance and adds them to the supplied folder. Mainly used in conjunction with 'Panel' or folder types that have multiple keyword values. Useful for choosing who to assign workflow items to at random, or perhaps specifying random values for a game or calendar.
- a. Source Folder – The Id or unique name of a folder to use as the randomizer and keyword source. Typically it's a folder that is used as a template.
 - b. Source Keyword Value Name – The name of the keyword value that will be used to pull the random value. This can be a keyword type that is a Panel in order to pull all values. If this is the case, it will just be that panel's name. Example: 'Address'. If a specific sub-panel value is desired, it can be qualified with a '.'. For example: 'Address.Street Name' where 'Address' is the Panel.
 - c. Source Keyword Percent Name – The name of the keyword value that will be used to test the percent chance. The folder must have a value between 0 and 100 in this keyword. The same rules apply for Panels as in 'Source Keyword Value Name' above.
 - d. Destination Keyword Value Name – The keyword value name on the destination folder to set the value for. Again, this can be a Panel if the 'Source Keyword Value Name' is also a Panel.
 - e. Allow Repeats – If checked, the randomizer may select the same keyword value multiple times based on the 'Minimum Random Finds' and 'Maximum Random Finds'.
 - f. Minimum Random Finds – If 0, it's possible no random keyword value may be selected. Otherwise, this number is used to force the number of random finds to this minimum number. The percent keyword will be tested over and over in order of the keyword values on the Source Folder until the minimum is satisfied.
 - g. Maximum Random Finds – Similar to the 'Minimum Random Finds' except this is the most that will be randomly selected before the random processing stops.
 - h. Process Dynamic Name - If checked, causes the system to recalculate the name on the supplied object after the keyword value(s) have been added.

- 43) Synchronize Records – Causes a Remote Storage Engine to synchronize its database records with this Storage Engine. Only the System Administrator can save this event type.
- a. Remote Storage Engine – The name of the Remote Storage Engine to utilize.
 - b. Push Data – If checked, the Remote Storage Engine will pull data from this Storage Engine. If unchecked, this Storage Engine will pull updates from the Remote Storage Engine.
 - c. Use Object Context – If checked, only the tables involved with the supplied objects will be updated. For example, if a folder was saved and fired this event, only folder related tables will be checked. This is to drastically speed up the synchronization process. It allows for near real-time synchronization when records are deleted or saved. If unchecked, all tables will be synchronized.
 - d. Last Synchronization – This field is read-only but shows the last time the synchronization was successfully completed. All synchronizations performed will be incremental from this date/time on.
- 44) Replicate Volume Engine – Causes a Volume Engine to replicate its files with any configured replicated Volume Engines. Only the System Administrator can save this event type.
- a. Volume Engine – The name of the Volume Engine to replicate.
- 45) Create Text Overlay Markup – Create a text overlay markup on the supplied file/image.
- a. Text – The text to overlay.
 - b. X – The X (horizontal) position to start the text overlay. This can be either pixels (a number) or a percent. For example: '150' would start 150 pixels from the very left side of the image. '50%' would start in the middle of the image.
 - c. Y – Same as 'X', except it is the Y (vertical) position of the image.
 - d. Color – The color of the text. Examples: Red, Black, White, Blue, etc.
 - e. Font Size – The font size of the text.
 - f. Line Height – This is only used when there are multiple lines in the text (by hitting the enter/return key). It is the spacing (in pixels) between lines).
 - g. Line Text Gap Height – This should normally be left blank. However, in rare cases where the markups are being generated from some foreign user interface, this may be utilized. It specifies (in pixels) the transparent gap between the very top of the text to the highest letter in the word of the first line.

- 46) Create Image Overlay Markup – Create an image overlay markup on the supplied file/image.
- a. File – This can be one of three options:
 - i. A unique file name
 - ii. A file id
 - iii. The bytes of a base 64 encoded image (must be one of the supported image formats)
 - b. X – The X (horizontal) position to start the image overlay. This can be either pixels (a number) or a percent. For example: '150' would start 150 pixels from the very left side of the image. '50%' would start in the middle of the image. The position is determined by the upper left corner of the overlay image.
 - c. Y – Same as 'X', except it is the Y (vertical) position of the image.
 - d. Scale Width – If a number below 1 is used, the width will be shrunk for the overlay image. For example, 0.5 is 50% of the original overlay image size. 0.3 is 30%.
 - e. Scale Height – Same as 'Scale Width' except applies to the height.
 - f. Translucency – A number that indicates how translucent (transparent) the overlay image will be. 100 means completely opaque (no translucency). 50 means 50% translucent. This is useful for watermarks.
- 47) Compress – Attempts data compression on the supplied file. This is useful to decrease the amount of space a file takes up on the server. Of course, no one will be able to view the file without first decompressing it.
- a. Compression Level – This determines the strength of the compression (at the cost of more memory and processing time). Default is "maximum".
- 48) Decompress – Decompresses the supplied file if it has been compressed at an earlier time.
- a. Ignore Errors – If checked, no error will be thrown if the file isn't compressed. This is useful if you aren't sure if a file is compressed, but wish to try decompression anyway. The original bytes will be left the way they are.
 - b. Save – If checked, the file bytes will be saved after being decompressed. Otherwise the decompression will simply occur in memory. It may be useful to leave unchecked if you want to decompress and stream down to a client for viewing purposes (with a 'getFile' action trigger), but leave compressed on the storage device of the server.

- 49) Execute Multiple Events – Executes a series of specified events. They execute in order in which they are selected.
- a. Abort on Execution Error – If checked, the execution will stop when the first error is reached. If unchecked, the next event will be executed as though nothing wrong happened prior.
 - b. Validate Events Before Execution – If checked, all the selected Events are checked to make sure they exist before even trying to execute any of them. Otherwise, execution begins and each event is checked right before its turn.
 - c. Process Conditions – If checked, any conditions associated with the action triggers of the selected events will be tested to see if the Event should execute. If unchecked, the selected events are automatically executed and any conditions set on the action triggers is ignored.
 - d. Events – The selected events to execute. The order in which they are selected determines the order in which they execute. By default, the “Available” events are any that have a ‘noActionEvent’ trigger specified. Otherwise, they will not appear in the list. You can manually type in an Event ID or Name however, if you want to execute one that doesn’t have a ‘noActionEvent’ trigger.
- 50) Merge PDF – Merges/parses one or more PDFs into single or multiple output PDFs. The parameters for a merge is quite complicated, so the various sections (mentioned below) will have ‘Add’ or ‘Delete’ buttons to add/remove multiples.
- a. Destination Folder – This can be one of two options:
 - i. A unique folder name
 - ii. A folder id
 - b. Destination File Name – The output (merged) PDF file name. If ‘Group By’ keywords are used, multiple files will be produced with this name. You can utilize dynamic tags to produce unique file names.
 - c. Extract Merged Text – If checked, the text inside the output PDF will be immediately extracted and saved for freeform text searching. The text will be available from a %text% dynamic tag in any ‘mergePDFEvent’ triggered, or any other following events.
 - d. Create Sub-Folder – If checked, the output file(s) will be created in a sub-folder of the destination folder. It will create one sub-folder each time the event is fired.
 - e. Create Sub-Folder Per Output File – If checked, and ‘Create Sub-Folder’ is also checked, a sub-folder will be created for each output file (applies to ‘Group By’ keyword conditions that cause multiple output files).
 - f. Sub-Folder Type – If ‘Create Sub-Folder’ is checked, this will be the type of folder created. Any dynamic name associated to this folder type will be processed when the folder is created.
 - g. Error Handling – This contains comma delimited error handling strings to control how errors behave at the highest level.
 - i. 1. Zero Page Output Options (default is to throw an error and abort):

1. ignore on zero page output
 2. log on zero page output
 - ii. Output Text Extraction Failure (default is to ignore)
 1. throw error on output text extraction failure
 2. log on output text extraction failure
- h. Source PDF(s) – This section allows an administrator to define where the source PDFs are located in order to perform the merge.
 - i. Friendly Name – This is a friendly name assigned to a source so that it can be referred to in the 'Page(s)' section.
 - ii. Source Type – One of the following values:
 1. File Id – a file id
 2. Folder Id – a folder id
 3. Unique File Name – a file with a unique name
 4. Unique Folder Name – a folder with a unique name
 5. OS File – a file located on an accessible drive (local or network) of the server.
 6. OS Folder – a folder located on an accessible drive (local or network) of the server.
 - iii. Source Value – This will either be the id, unique name, or full path to the file/folder depending on the 'Source Type'. Examples:
 1. Unique File Name – Bob's 401k.pdf
 2. Unique Folder Name – Mike's Templates
 3. OS File – c:\reports\header.pdf
 4. OS Folder - [\\reportsrv\personnel\reports\%date%](#)
 - iv. PDF Password – If the PDF is password protected (inside the PDF itself – this is not ES Imaging encryption), an administrator may provide it here.
 - v. Error Handling – This contains comma delimited error handling strings to control how errors behave at the source level. Here are the possible options:
 1. Read Failure Options (default is to throw an error and abort):
 - a. ignore on unreadable file
 - b. log on unreadable file
 2. Non-PDF Source Options (default is to ignore)
 - a. throw error on non pdfs
 - b. log on non pdfs
 3. PDF Decryption Failure (default is to ignore)
 - a. throw error on decryption failure
- i. Keyword(s) Parsed From Source Name – This section allows an administrator to parse a source (input) file name to extract keywords. This is useful when you are processing batches of similar PDFs, but each for different people, ids, etc. It is also useful to dynamically name the 'Destination File' using dynamic tags.

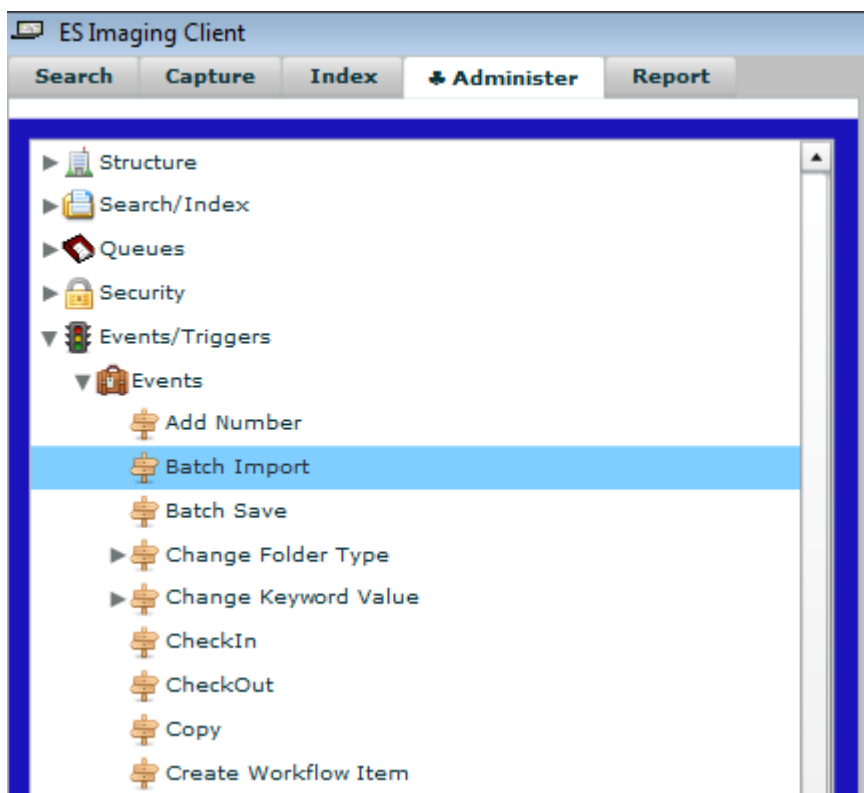
- i. **Keyword Name** – The keyword name to fill. If the keyword doesn't exist, it will still work in order to temporarily evaluate tags for dynamic names or conditions. For example, you can specify 'Lab Id' and then refer to this in the 'Destination File' as '%Lab Id.value%'.
- ii. **Parse Start Position** – A number starting at 0. This is the position to start parsing the keyword value.
- iii. **Parse Stop Position** – If specified, this is the last position to read for the keyword. For example, Start of 0 and Stop of 4 on a file name of 'testing' will result in 'test'. If not specified, it will assume either a delimiter is specified or go to the end of the file name.
- iv. **Parse Delimiter** – If specified, the parsing will continue to read from 'Parse Start Position' until encountering the first 'Parse Delimiter'. If this occurs before 'Parse Stop Position', it will stop early.
- v. **Save Keyword To Folder** – If checked, and 'Create Sub-Folder' is not checked, the 'Destination Folder' will have the parse keywords saved to it. Sub-folders will always have the parse keywords saved to them (if the keywords are tied to that folder type of course).
- vi. **Group By** – If checked, any source PDFs discovered that have the same "parsed" keyword(s) will be grouped into the same output PDF. This will only happen if the keyword parsing is setup correctly. For example, let's assume the 'Destination File' name is '%Employee Id.value%.pdf'. Let's also assume the 'Keyword Name' is 'Employee Id' and 'Parse Start Position' is '0' and 'Parse Stop Position' is '3'. This will result in:
 - 1. 123 header.pdf, 456 header.pdf, 123 bob.pdf, 456 jane.pdf will result in the following merged together:
 - a. Output file: 123.pdf
 - i. Sources: 123 header.pdf and 123 bob.pdf.
 - b. Output file: 456.pdf
 - i. Sources: 456 header.pdf and 456 jane.pdf.
- j. **Page(s)** – This section determines the order in which pages are merged from the sources into the output(s). The order in which the 'page criteria' is added is the sequence in which conditions are "tested". Because of this, this section has 'Up' and 'Down' buttons to change the order of the conditions (when multiples exist).
 - i. **PDF Friendly Name** - The merging happens for one document at a time, specified by the 'PDF Friendly Name'. You can bounce back and forth among PDF sources at any time. For example, the first source may be 'Header', second 'Body', and third 'Header' again. Perhaps you want to include page 1 from the header, then all from the body, and finally page 2 from the header. This is allowed by specifying multiple 'Page(s)'.

- ii. Error Handling – This contains comma delimited error handling strings to control how errors behave at the page level. Here are the possible options:
 - 1. No PDF To Process (default is to ignore)
 - a. throw error on no documents
- iii. Start Page Merge When – This section specifies when to start pulling in pages from the source. You can have any number of conditions and chain them together with 'and/or' conditions.
 - 1. Field – The type of field/analysis to perform on the page.
 - a. File Name – The source's file name.
 - b. Is Even Page – True if the current page is even (odd = false).
 - c. Is First Page – True if the current page is the first page.
 - d. Is Last Page – True if the current page is the last page.
 - e. Last Page Included – The number of the previous page that had been analyzed. For example, if page 6 is currently being analyzed, this field would have the value of 5. Resets when a new document begins analysis.
 - f. Lines of Text – The number (count) of lines of text on the current page.
 - g. Output Page Count – The number (count) of pages that have been output to the destination pdf at this time. The number will reset when a new output file is being generated from within the same event (because of 'Group By' on keywords).
 - h. Page – The number of the current page. Examples: 1, 5, 7, etc.
 - i. Page Included – A list of pages that were included in the output file. For example, let's say pages 1, 2, 4, and 6 were included from the source and merged into the output file. Now let's assume page 7 is currently being analyzed. The condition could be checked with 'Page Included' = '1' and the result would be true. If it was checked with 'Page Included' = '3' the result would be false. Resets when a new document begins analysis.
 - j. Text – Used to compare the text contents of the current page. For example, if the page contained the words 'this is the axis report', the condition could be checked with 'Text' like 'axis report'. Most of the time either 'like' or 'not like' will be used since '=' would mean every single word on the page must match the 'Value'.
 - k. Document Text – Same as 'Text', except the value can occur on any page of the source document. In other words, the condition will be true as long as any page gets the match, regardless of the text on the current page being analyzed.

2. Operator – How the comparison will be performed. Options for most 'Fields': =, like, not =, not like, <, >.
 3. Value – The value to compare against. This can also be dynamic tags.
 4. Conjunction – How to compare the conditions. Options:
 - a. And – This condition, and the next condition, must be true or the entire evaluation is false.
 - b. Or – This condition, or the next condition, may be true for the entire evaluation to be true.
- iv. Stop Page Merge When – This optional section is the same as the 'Start Page Merge When', except it determines when to stop including pages. If this section has no values, the stop occurs immediately and only the conditions specified in the 'Start Page Merge When' section will result in merged pages.
1. Include Stop Page – If checked, the page that results in the 'Stop' of the merge will be included in the output. If false, it is excluded. For example, let's say merging begins at page 2, and continues until page 10 (because of the conditions specified in this section), here is an example of behavior:
 - a. Output file pages:
 - i. With 'Include Stop Page' checked – 2,3,4,5,6,7,8,9,10
 - ii. With 'Include Stop Page' unchecked – 2,3,4,5,6,7,8,9

Create an Event

- Click on the “Administer” tab.
- Expand the “Events/Triggers” tree node.
- Expand the “Events” tree node (see the figure below as a demonstration).
- Right click on the desired Event Type (such as “Create Workflow Item”).
- Left click the “New Event” context menu item.
- Proceed with filling in the desired parameters as described in the “Event Types” section of the manual.
- Click the “Save” button when finished.



Triggers

There are 3 ways to trigger events: Action, Schedule, or On Demand. An “Action” trigger is one that occurs after a given action (save, delete, after indexing, etc). A “Schedule” trigger is one that is fired based on the current date and time (think batch processing). An “On Demand” trigger happens when you right click on a “Schedule Event” and select “Execute”.

Action Types

The following is a list of the types of action triggers in ES Imaging (XXX means there are multiple object types that support this function):

- 1) checkIn – Triggered when an object is checked in.
- 2) checkOut - Triggered when an object is checked out.
- 3) createFolderEvent – Triggered when a folder is created from the “Create Folder” event.
- 4) createWorkflowItemEvent - Triggered when a workflow item is created from the “Create Workflow Item” event.
- 5) decryptFile - Triggered when a file is decrypted.
- 6) deleteXXX - Triggered when an object is deleted.
- 7) encryptFile - Triggered when a file is encrypted.
- 8) getFile - Triggered when a file’s properties are retrieved from the database.
- 9) getFileBytes – Triggered when a file’s bytes are retrieved (when a person is viewing the actual contents of the file).
- 10) getFolder - Triggered when a folder’s properties are retrieved from the database.
- 11) getFolderTypeKeywordValues – Triggered any time a user views the properties of a folder, or changes the folder type of a folder. The main purpose for this is so default values can be set with the “Change Keyword Value” event type.
- 12) getPossibleKeywordValues – Triggered any time a folder or file is edited and the possible drop down list values are retrieved. Useful when used in conjunction with a GetSearchDataEvent in order to control drop down list values. Also see the ‘Current Keyword Value.value’ tag for additional information.
- 13) processKeywordValueChange – Triggered any time a user is editing keyword values on a folder. Each time a character is typed, or a drop down list value selected, the folder data is transferred to the server’s memory. Any changes made to the folder are reflected on the client’s screen in memory (not saved to the database). The main purpose for this action trigger is to automatically change folder types or keyword values any time a given keyword value is updated.
- 14) indexTransaction - Triggered when folder(s) or file(s) are indexed from the “Index” tab. The main purpose for this is so workflow items can be created via the “Create Workflow Item” event type.
- 15) moveWorkflowItemEvent - Triggered when a workflow item is moved from a “Move Workflow Item” event.
- 16) newXXX - Triggered when a user is creating a new object. The main purpose for this is to set default keyword values, folder type, name, etc. on the newly created object.
- 17) saveXXX - Triggered when an object is saved.

- 18) searchEvent – Triggered when a “Search” event is performed. The main purpose for this is to allow events to be fired from a schedule and to help administrators segregate action triggers from schedule triggers.
- 19) mergePDFEvent – Triggered every time an output PDF is created/saved from a Merge PDF Event. This allows performing other actions, such as copying other files into the file’s parent folder. The source provided to the triggered event is the file just saved.
- 20) noActionEvent – This is a special action trigger that is never fired, unless explicitly fired from another event (such as an Execute Multiple Events event).
- 21) eventFired – Triggered any time an event is fired. It is important to narrow the conditions down in this action trigger or you can end up with many events firing unintentionally. Use the ‘Previous Event Name’ condition to see the event that fired before this action trigger.
- 22) login – Triggered any time a user logs in to ES Imaging through the ES Imaging Client. Technically, it is any time the ‘loginTest’ method is called on the webservice.
- 23) eventException – Triggered any time an event fails. Used mainly for error handling.

Schedule Types

As mentioned earlier, schedule triggers happen based on the date and time. They are “scheduled” to happen once or recurring. If recurring, they can happen on a specified interval. The interval relates to a type of schedule:

- 1) Minutely
- 2) Hourly
- 3) Daily
- 4) Weekly
- 5) Monthly
- 6) Yearly
- 7) On Demand (this is a special type and cannot be ‘scheduled’)

An administrator can specify a start date/time, as well as an end date and time.

If a schedule has not executed yet, and the start date and time is in the past, it will run immediately. Otherwise, the schedule will wait until the date and time passes. The interval specifies how often with the given schedule type. As an example, an interval of 3, with a schedule type of weekly, means the schedule will run every 3 weeks.

If an end date/time is specified, the schedule will not kick off any triggers past that value. Ones that are already running will continue to run until completion.

There are only a few types of events that are allowed to be scheduled. They are types that are independent of object context. The allowed event types are:

- 1) Search
- 2) Send Email
- 3) Batch Email Import
- 4) Batch Import
- 5) Batch Save
- 6) Sleep
- 7) Insert Database Record
- 8) Throw Error Message

It's important to note that the “Search” event type can be used to feed results in to any other action trigger. An example may be that you want to delete all folders greater than 5 years old. You could perform a “Search” event looking for folders with “Create Time” > 5 years old and then feed those results into a “Delete” event.

Transaction History

All action and schedule triggers have the ability to log transaction history. This allows an administrator to see when an event was fired as well as the results (or error). Schedule triggers automatically log their history since there is no other way to tell the results. Action triggers have it enabled as an option since any errors that would occur would most likely result in an error message making its way back to the client.

The transaction history appears in the tree under each appropriate trigger. You can view the details about the transaction by clicking on a specific instance in the tree.

Trigger Conditions

Events are only triggered if their conditions are satisfied (during a test). By default, a trigger has no conditions. This means the event will fire no matter what, when it is triggered. If there are conditions specified, all or some of them must match in order to fire the event. You can think of the matching as similar to the advanced search. You can chain conditions together with “and” or “or” conjunctions.

All triggers can be disabled by checking the “Disabled” field.

Some action types have a “Before” and “After” action. If you’re working with an action trigger of that type, a “Before Action” checkbox can be checked. If it is checked, the trigger is tested before the given action takes place. For example, if you want to throw an error message if certain conditions aren’t met on a given folder, you can do so BEFORE the save actually takes place. This will cause the save to abort and the error message to be sent back to the client. If the throw error message event happens AFTER the save (the checkbox unchecked), the save will STILL take place, but the client will receive an error message after the fact.

Trigger Sequence

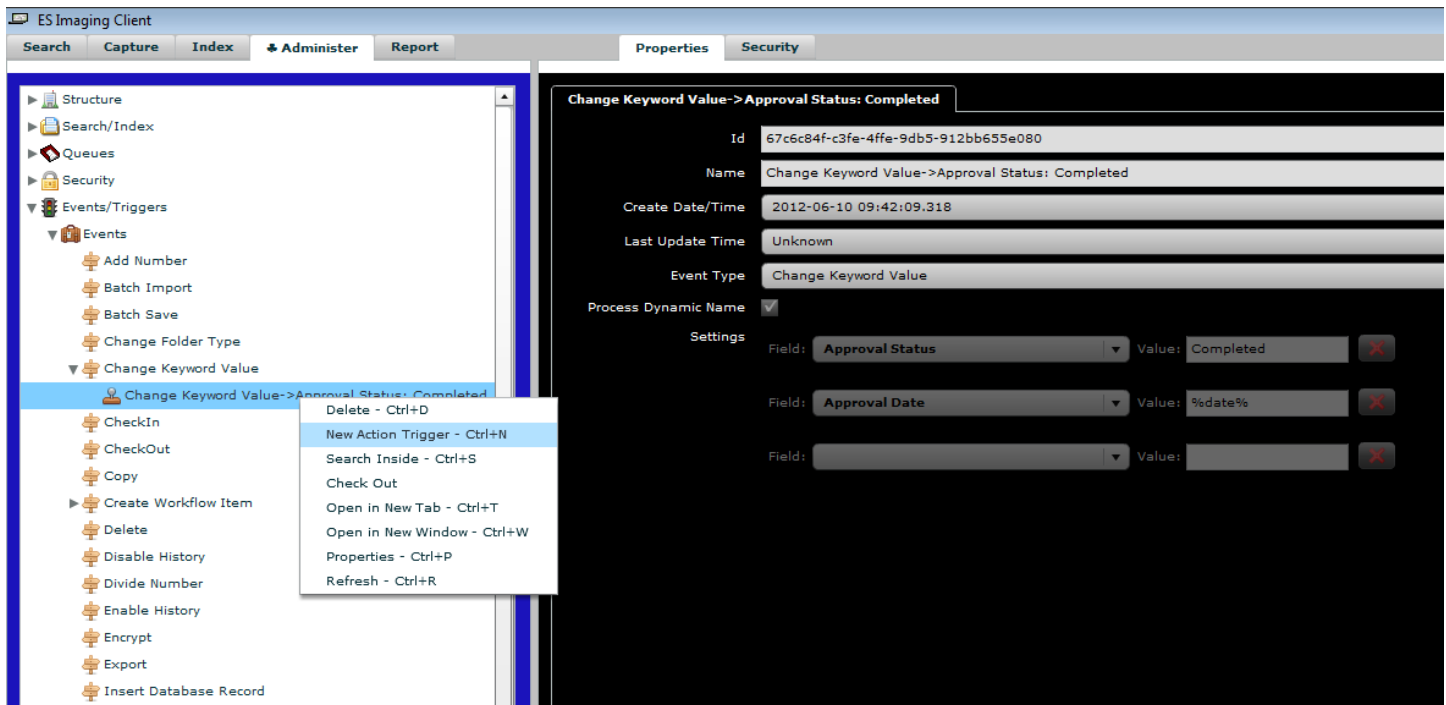
Triggers can be “chained” together for a given action type/schedule. They are tested sequentially in order by their specified sequence number. A lower number is higher priority. If 2 or more triggers are found with the same sequence number, the one created first is tested first.

The reason sequence is important is because of dependencies. For example, if you want to set a keyword called “completion date” to today’s date, but the supplied folder isn’t a folder type that has that keyword, you will need to do a “Change Folder Type” event first. You would then follow up with a “Change Keyword Value” event. The sequence numbers do not matter except that one is greater than another. It could be 0 and 1. Or 0 and 5. Or 1 and 10. It’s usually good practice to leave a gap in between the numbers just in case you need to add a step in the process. If you don’t, you may have to renumber all the sequences once you insert the extra step.

Creating an Action Trigger (Events Perspective)

Action Triggers can be created by following the steps described below:

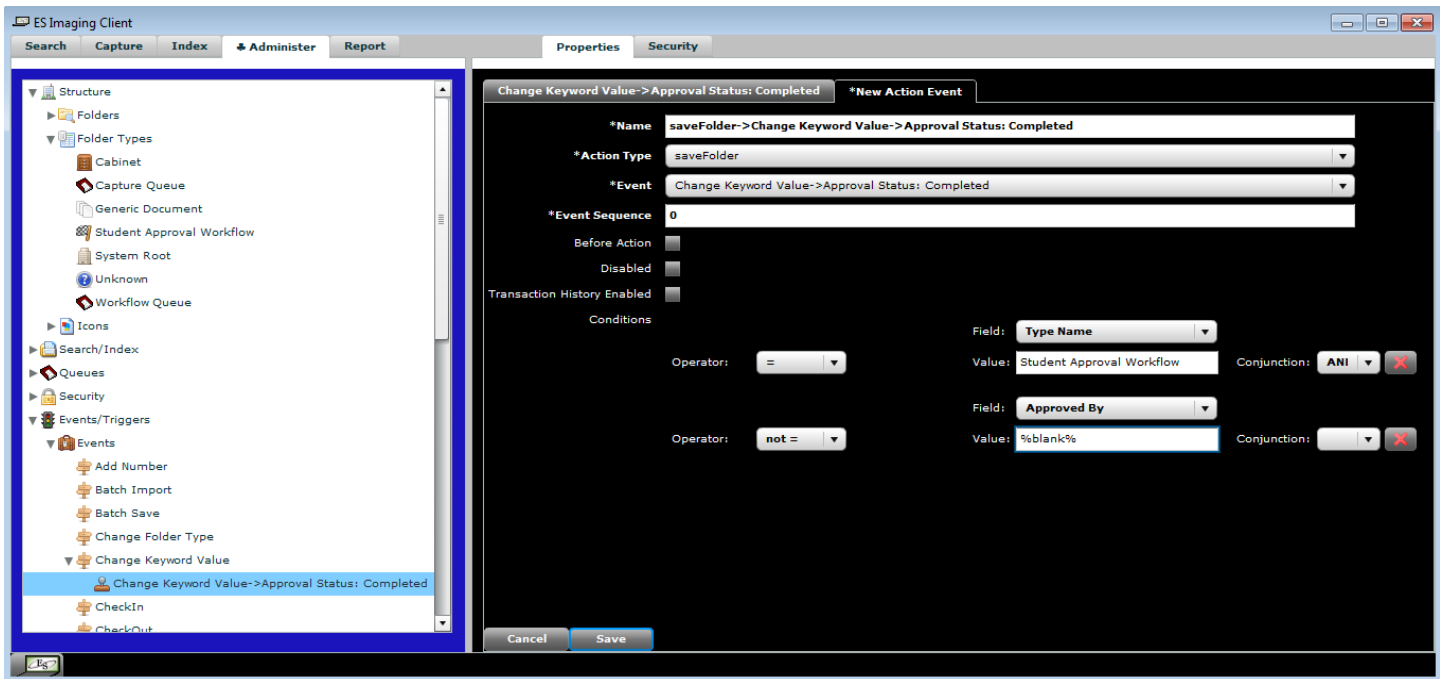
- Click on the “Administer” tab.
- Expand the “Events/Triggers” tree node.
- Expand the “Events” tree node.
- Expand the Event Type that represents the specific Event you wish to create a trigger for.
- Right click on the desired Event (see the figure below for an example).



- Left click the “New Action Trigger” context menu item.

Administration Guide

- Give the Action Trigger a very descriptive “Name” that describes the type of event and when the trigger will fire (see the figure below for an example). It is a good idea to stick to a consistent naming convention so that sorting and searching of Action Triggers will be manageable.

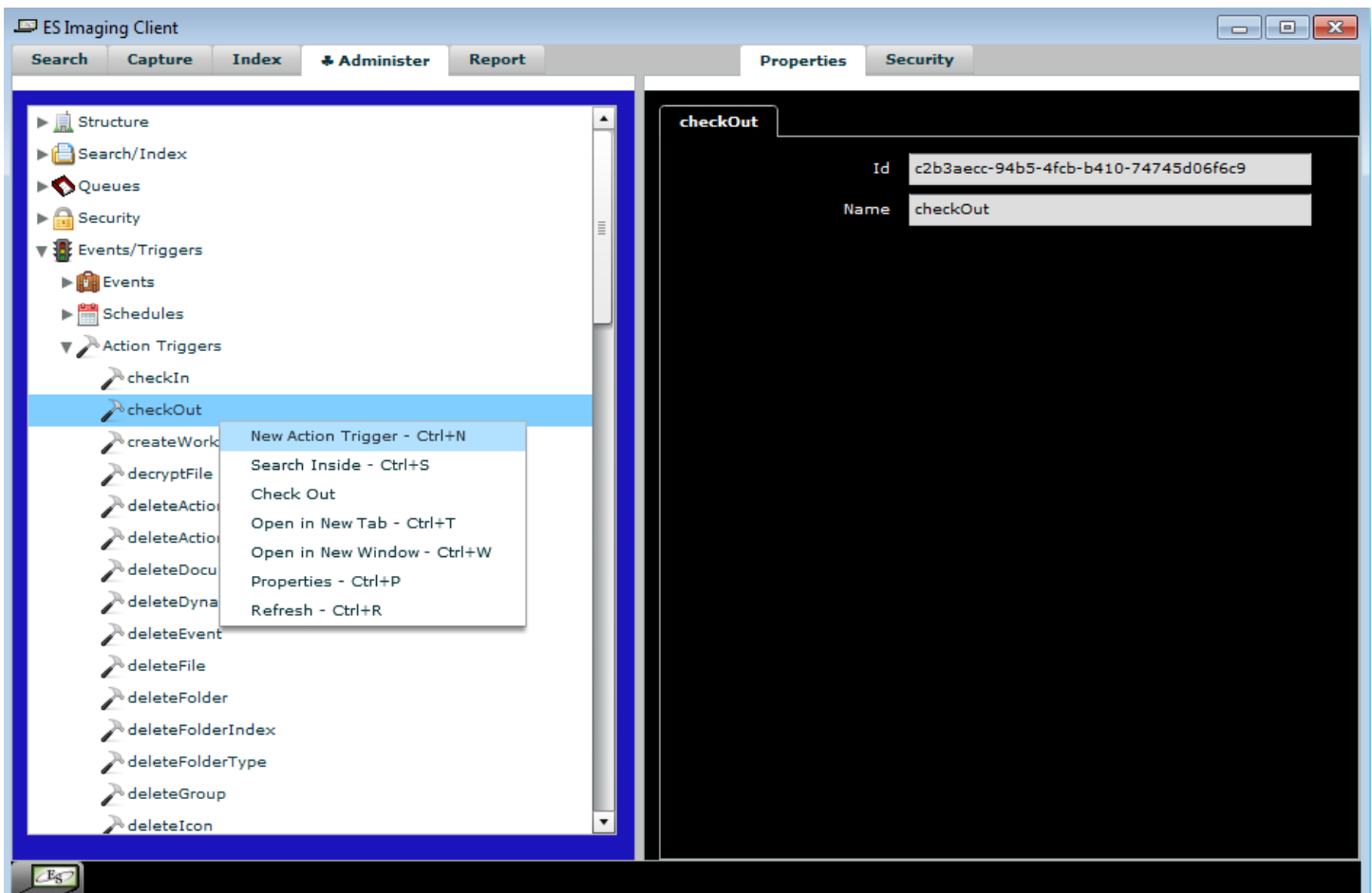


- Choose the desired “Action Type”. In the example above, we chose “saveFolder”. This means the trigger will be tested after a folder is saved.
- Specify the “Event Sequence” to determine the order in which this trigger will be tested. A higher number means the trigger will be tested after other Action Triggers with lower numbers.
- If you want the trigger to be tested before the given action completes (again, saving of a folder in this case), check the box. This only applies to Action Types that have “before” and “after” behavior.
- If you don’t want the trigger to be live (it will never be tested so the event will never fire from this trigger), check the “Disabled” box. Otherwise leave it unchecked and it will be tested the next time the given action is performed.
- If you want to be able to see history on when the trigger fires, and what the results are from the event, check the “Transaction History Enabled” box. The history will appear as tree nodes under this specific trigger after it is fired. If you do not want to see history, leave the box unchecked.
- The “Conditions” of an Action Trigger dictate when a trigger will fire. If no conditions are provided, the trigger will always fire (unconditionally) when the given action occurs in the system. In the example above, there are 2 conditions. The 1st is when the “Type Name” (or folder type) is equal to “Student Approval Workflow” AND the “Approved By” keyword value is NOT EQUAL to “%blank%”. Essentially this means the saved folder must be a “Student Approval Workflow” document and it must have a value set in the “Approved By” keyword. The “%blank%” is a special dynamic tag that evaluates to an empty string (or blank value) at run time. Refer to the Dynamic Tags section for more information.
- Click the “Save” button to complete the save.

Creating an Action Trigger (Action Triggers Perspective)

You can also create an Action Trigger by navigating to the “Action Triggers” tree node under “Events/Triggers” in the Administer tree. This view is geared more toward the sequence of when action triggers are tested. Any existing action triggers will be sorted underneath each type (checkIn, createWorkflowItem, etc) by sequence number.

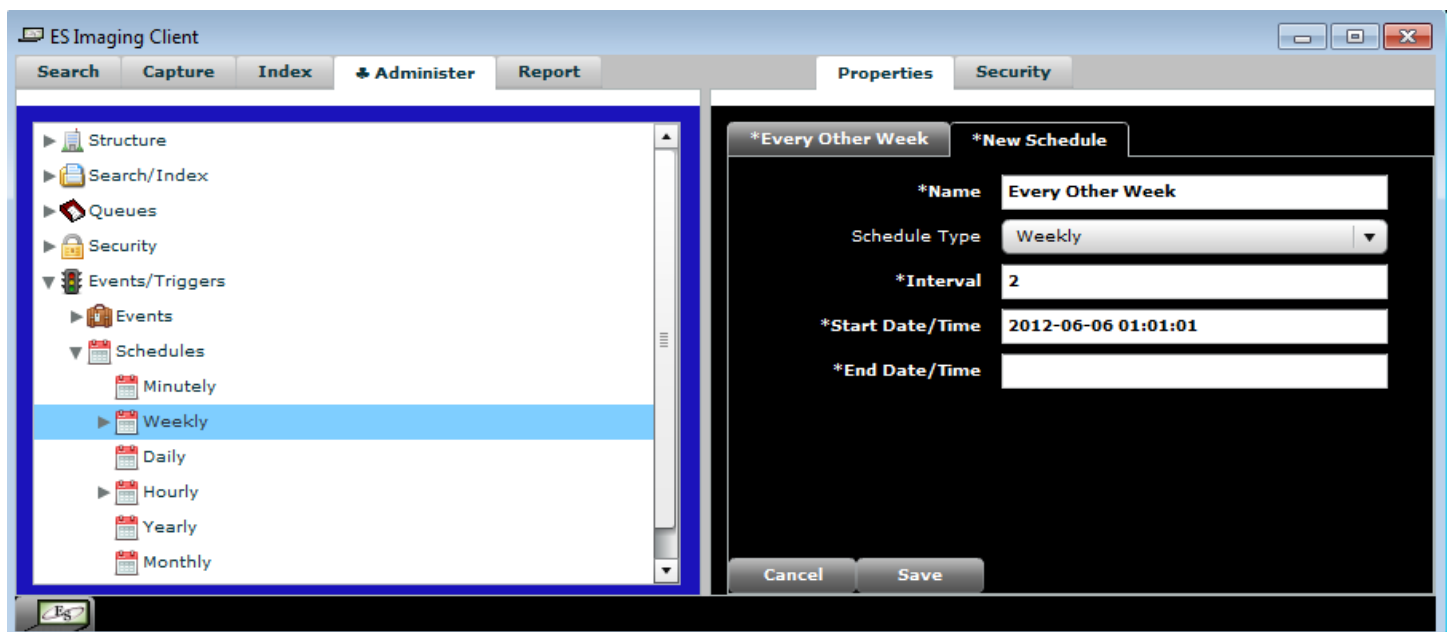
The figure below shows this particular perspective as opposed to the “Events” perspective as described above.



Creating a Schedule

Schedule Triggers can be created by following the steps described below:

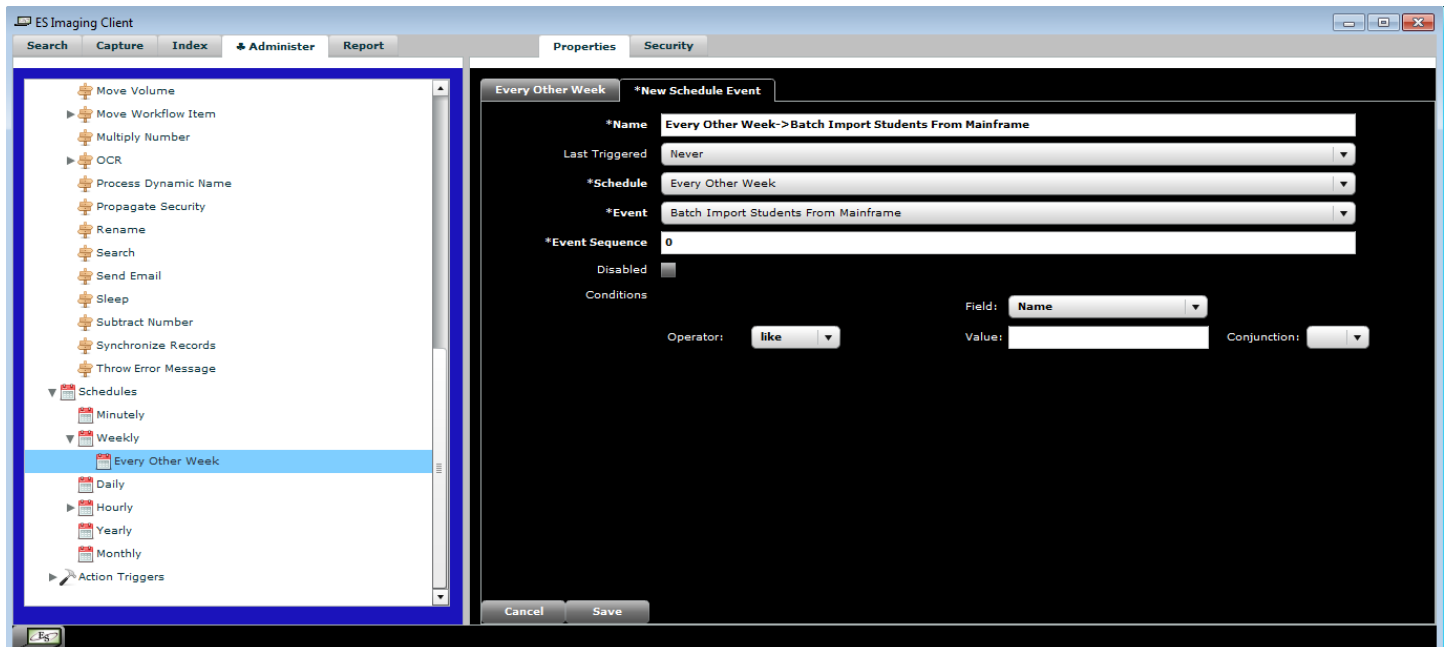
- Click on the “Administer” tab.
- Expand the “Events/Triggers” tree node.
- Expand the “Schedules” tree node.
- Right click on the desired Schedule Type.
- Left click on “New Schedule”.
- Give the schedule a meaningful “Name”. See the example provided in the figure below.
- Specify how often the schedule should fire by providing a number for the “Interval”. In the example below, 2 is used. Since the “Schedule Type” is Weekly, the schedule will execute every 2 weeks from the Start Date/Time.
- Provide a “Start Date/Time” in ‘yyyy-mm-dd hh:MM:ss’ format. This specifies when the schedule should first execute (if the date/time is in the future). If the date/time is in the past, the schedule will execute immediately once a Schedule Trigger is created.
- You can optionally provide an “End Date/Time” in order to say when the schedule should no longer execute. If the date/time is in the past, the schedule will never execute again.



Creating a Schedule Trigger (Schedules Perspective)

Schedule Triggers can be created by following the steps described below:

- Click on the “Administer” tab.
- Expand the “Events/Triggers” tree node.
- Expand the “Schedules” tree node.
- Expand the Schedule Type that represents the specific Schedule you wish to create a trigger for.
- Right click on the desired Schedule.
- Left click the “New Schedule Trigger” context menu item. The figure below demonstrates what one may decide to enter for a Schedule Trigger.



- Give the Schedule Trigger a very descriptive “Name” that describes the type of event and when the trigger will fire (see the figure above for an example). It is a good idea to stick to a consistent naming convention so that sorting and searching of Schedule Triggers will be manageable.
- Select the “Event” you wish to schedule.
- If you are going to fire more than one event from this schedule, specify the “Event Sequence” as a number. The higher the number, the later the event will fire.
- If you don’t want the trigger to be live (it will never be tested so the event will never fire from this trigger), check the “Disabled” box. Otherwise leave it unchecked for it to be live.
- If the “Asynchronous” box is checked, the event will fire simultaneously and run in the background.
- The “Conditions” of a Schedule Trigger dictate when a trigger will fire. If no conditions are provided (which is normally the case), the trigger will always fire (unconditionally) when the schedule is past due. Conditions are normally used on Action Triggers, but ES Imaging also allows for them on Schedule Triggers should the rare occasion occur.
- Click the “Save” button to complete the save.

Creating a Schedule Trigger (Events Perspective)

Schedule Triggers can also be created the same way Action Triggers are created as explained in the “Creating an Action Trigger (Events Perspective)” section of the manual. The only difference is “New Schedule Trigger” should be selected from the context menu.

Dynamic Tags

ES Imaging utilizes the concept of dynamic tags, or formulas. These allow an administrator to dynamically calculate values at run time instead of a given value having to be static. Tags are utilized by placing them within '%' characters. For example, %date% will dynamically calculate today's date and fill in the value in 'yyyy-mm-dd' format.

Some tags are universal (work any time) and others only work when there is an object context (like when testing a condition on a trigger).

Here are the primary times dynamic tags can be used (there may be more):

- 1) The quick search field.
- 2) The "Value" field in advanced searching.
- 3) The "Value" field of a "Condition" on a trigger.
- 4) The "Value" field of a "Change Keyword Value" event.
- 5) The "Folder Type Name" field of a "Change Keyword Type" event.
- 6) The "Login Id" field of a "CheckOut" event.
- 7) The "Destination Folder" field of a "Copy" event.
- 8) The "Workflow Queue(s)" text field of "Create Workflow Item" and "Move Workflow Item" events.
- 9) The "Checkout Login Id(s)" text field of "Create Workflow Item" and "Move Workflow Item" events.
- 10) The "Value" field of "Initial Keyword Values" in a "Create Workflow Item" event.
- 11) The "Value" field of "Change Keyword Values" in a "Move Workflow Item" event.
- 12) The "Destination Path" field of an "Export" event.
- 13) The "Columns" text field of an "Insert Database Record" event.
- 14) The "Values" text field of an "Insert Database Record" event.
- 15) The "Destination Folder" field of a "Move" event.
- 16) The "New Name" field of a "Rename" event.
- 17) The "Value" field of a "Search" event.
- 18) All the fields in the "Send Email" event.
- 19) The "Message" field of a "Throw Error Message" event.
- 20) The value of a String Keyword Value on a folder.

Universal Tags

The following is a list of the dynamic tags that can be used in conditions, as well as other places in ES Imaging:

- guid - Generates a random GUID (Globally Unique Identifier)
- year - Four digit year
- month - Two digit month
- day - Two digit day
- hour - Two digit hour
- minute - Two digit minute
- second - Two digit second
- millisecond - Four digit millisecond
- longtime - Single integer timestamp value to the millisecond
- dayofweek - 1 digit day of the week
- dayofmonth - 2 digit day of the month
- weekofmonth - 1 digit week of the month
- weekofyear - 2 digit week of the year
- admonths – The number of months that have passed since 0 AD.
- adweeks - The number of weeks that have passed since 0 AD.
- addays - The number of days that have passed since 0 AD.
- adhours - The number of hours that have passed since 0 AD.
- adminutes - The number of minutes that have passed since 0 AD.
- adseconds - The number of seconds that have passed since 0 AD.
- date - yyyy-mm-dd (or the server's date format)
- datetime (or timestamp) - yyyy-mm-dd hh:mm:ss (or the server's date/time format)
- user.name - User name of the current user (on the server)
- host.name - The host name of the machine (on the server)
- os.name – The name and version number of the operating system (on the server)
- java.version – The version of Java running (on the server)
- host.address - The IP address of the host machine (on the sever)
- user.home - The home directory for the current user (on the server)
- user.dir - The current user's directory (on the server)
- java.io.tmpdir - The temp directory (on the server)
- launch.text - The text value selected by the user in the launcher
- launch.engine - The engine name selected by the user in the launcher
- xml.dir - The directory where the engine XML files are being utilized
- log.dir - The directory where the main log XML files are being written

- `newline` - An newline output character
- `tab` – A tab output character
- `empty` (or blank) – Generates nothing. The purpose for this is to allow saving of blank values.
- `file.separator` – The file separator character for the specific operating system on the server
- `line.separator` – The line separator character for the specific operating system on the server (similar to `newline`)
- `literal('AAA')` – Output the exact phrase inside the single quote. In order to output a single quote, it must be escaped with another `'` character. Example: `%literal('My name is O'Charley')%`. This will result in: My name is O'Charley. This allows special Java character commands to be utilized as well (such as `\n`).

Object Context Tags

- id - The id of the source object
- name - The name of the source object
- parentName - The parent name of the source object (folders and files)
- parentId - The id of the parent of the source object (folders and files)
- text - The freeform text value (if applicable) of the source object
- lastUpdateTime - The last date and time the source object was saved
- lastUserActivity – The last date and time an activity was performed by the user currently logged in ES Imaging
- createTime - The date and time the source object was originally created
- typeId - For files, the id of the mime-type. For folders, the id of the folder type.
- typeName - For files, the name of the mime-type. For folders, the name of the folder type.
- hierarchyParentName - The parent name of any parent in the hierarchy for the source object (folders and files)
- versionName - The version name for the source object (files only)
- originalFileName - The original file name (if imported) for the source object (files only)
- fileName – The name of the file. This only works with certain events, such as the 'Merge PDF' event. In most cases it is the name of the file currently being processed (source file, not the output file).
- checkedOutLoginId - The login id of the user that has the source object checked out
- imageMarkupText - The text value of any text type markups for the source object (files only)
- userId - The user id for the user currently logged in ES Imaging
- loginId - The login id for the user currently logged in ES Imaging
- actionType - Specifies if the current action trigger is before or after and the name of the action type
- eventName – Specifies the current event name being triggered.
- sourceFiles.value – A special tag that can be utilized when a 'Merge PDF' event is executing. It contains a comma delimited list of all the source files (and full paths if using 'OS File' or 'OS Folder'). This is accessible from any event fired by a 'mergePDFEvent' action trigger as well.
- XXX.value - The keyword value for a keyword type named 'XXX' (folders and files). An example may be: %Organization.value%.
- XXX.id - The keyword value id for a keyword type named 'XXX' (folders and files). An example may be: %Last Name.id%.
- objectType - The type of object the source is (file, folder, keyword type, etc)
- className - The name of the Java class the source object is
- childCount - The number of children underneath the source object (folder)

- `parent.XXX` – Looks at the parent to determine the tag value (folders and files). Parents can be chained together to go up the hierarchy like this: `parent.parent.name` for example. An example may be: `%parent.parent.name%`.
- `httpGet('AAA')` – Calls the URL 'AAA' and the results are used as the value. This works with HTTP and HTTPS. For example:
`%httpget('https://www.esimaging.com/ESImagingClient.version')%`
- `httpPost('AAA')` – Similar to `httpGet`, except a POST command is utilized. For example:
`%httppost('http://www.esimaging.com:9443/ESImagingClient.version')%`
- `random('X','Y')` – Generates a random number between X and Y. For example: `%random('-2','5')%` will result in any one of the following values: -2, -1, 0, 1, 2, 3, 4, or 5.
- `randomValue('XXX','CHANCES','DELIMITER')` – Obtains a single random value from a string 'XXX' with delimited values.
 - a. 'CHANCES'
 - i. If omitted or blank, one of the values in 'XXX' will be automatically chosen randomly (equal chance).
 - ii. If a single value, a random number will be generated between 1 and 100. If this number is greater than the randomly generated number, the current value in 'XXX' will be selected. The values are done in order of the values in the list.
 - iii. If multiple values, the same rule applies as above, but the series of numbers in 'CHANCES' are cycled through.
 - b. 'DELIMITER' – The delimiter used for the values in the string. Should default to '~' since that is what Get Search Data events normally use.
 - c. Examples:
 - i. `%randomValue('Bob~Mike~Tony')%` - A 33.33% chance for any of the following: 'Bob', 'Mike', or 'Tony'.
 - ii. `%randomValue('Bob~Mike~Tony','25')%` – First, 'Bob' has a 25% chance of being selected. If not, 'Mike' then has a 25% chance. If not, 'Tony' has a 25% chance. Note: It is possible for no value to be selected in this case! If that happens, an empty string is used.
 - iii. `%randomValue('Bob, Mike, Tony','25',',')%` – Same as the example above, but a comma is being used as a delimiter.
- `getChild('AAA').XXX` – Looks for a child with the name of 'AAA' to determine the tag value (folders). Child names can be chained together to go down the hierarchy like this:
`%getChild('Clients','John Doe','Portfolio').Status.value%`.
- `replace('AAA','BBB').XXX` – Replaces all text occurrences of 'AAA' with 'BBB' for 'XXX'. If you want to replace multiples, they can be chained like this:
`%replace('Smithe','Smith').replace('Dow','Doe').LastName.value%`.
- `conditionalReplace('AAA','TRUE_REPLACE','FALSE_REPLACE').XXX` – Replaces the tag value with 'TRUE_REPLACE' if the phrase of 'AAA' is found anywhere in the string of XXX. If the phrase of 'AAA' is not found, 'FALSE_REPLACE' is returned. For example:

`%conditionalReplace('true','Y','N').Parent.Status Complete.value%`. In this example, if the keyword value of 'Status Complete' (on the parent of the supplied object) is 'true', a 'Y' will be returned. Otherwise an 'N' is returned. Empty strings can also be returned by specifying ''.

- `upperCase('aaa')` – Converts the supplied value to uppercase.
- `lowerCase('AAA')` – Converts the supplied value to lowercase.
- `length('XXX')` – Returns the length of the supplied value.
- `isEmpty('XXX')` – Returns true if the supplied value is an empty string. Otherwise, false.
- `isNull('XXX')` – Return true if the supplied value is 'null', otherwise false.
- `isNotEmpty('XXX')` – Return true if the supplied value is not an empty string. Otherwise, false.
- `isNotNull('XXX')` – Returns true if the supplied value is not 'null', otherwise false.
- `isEmptyOrNull('XXX')` – Returns true if the supplied value is either an empty string or 'null'.
- `isNotEmptyOrNull('XXX')` – Returns true if the supplied value is not an empty string and not 'null'.
- `isNumber('XXX')` – Returns true if the supplied value is a number (integer or decimal).
- `isInteger('XXX')` – Returns true if the supplied value is an integer (whole number).
- `isBoolean('XXX')` – Returns true if the supplied value is true or false.
- `isBase64Encoded('XXX')` – Returns true if the supplied value is Base 64 encoded.
- `isGreaterThan('XXX','YYY')` – Returns true if the supplied value 'XXX' is greater than 'YYY'. Otherwise, false is returned. Must be numbers or an error will be generated.
- `isGreaterThanOrEqualTo('XXX','YYY')` – 'XXX' must be > or = to 'YYY'.
- `isLessThan('XXX','YYY')` – 'XXX' must be < 'YYY'.
- `isLessThanOrEqualTo('XXX','YYY')` – 'XXX' must be < or = to 'YYY'.
- `valueIfEqual('AAA','CONDITION','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is equal to 'CONDITION'. If the phrase of 'AAA' is not found, the original phrase 'AAA' is returned. Examples:
 - a. `%valueIfEqual('computer','computer','laptop')%` would result in 'laptop'.
 - b. `%valueIfEqual('mycomputer','computer','laptop')%` would result in 'mycomputer'.
- `valueIfNotEqual('AAA','CONDITION','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is not equal to 'CONDITION'. If the phrase of 'AAA' is not found, the original phrase 'AAA' is returned.
- `valueIfInSet('AAA','CONDITIONS','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is somewhere in the set of ~ delimited values in 'CONDITIONS'. If the phrase of 'AAA' is not found, the original phrase 'AAA' is returned. Examples:
 - a. `%valueIfInSet('test','ok~no~yes~test','foundit')%` would result in 'foundit'.
 - b. `%valueIfInSet('test','ok~no~yes','foundit')%` would result in 'test'.
- `valueIfNotInSet('AAA','CONDITIONS','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is not found anywhere in the set of ~ delimited values in 'CONDITIONS'. If the phrase of 'AAA' is found, the original phrase 'AAA' is returned.

- `valueIfNotEmptyOrNull('AAA','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is not an empty string and is not null. Otherwise, the original phrase 'AAA' is returned.
- `valueIfEmptyOrNull('AAA','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is an empty string or null. Otherwise, the original phrase 'AAA' is returned.
- `valueIfGreaterThan('AAA','CONDITION','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is greater than 'CONDITION'. If the phrase of 'AAA' is not found, the original phrase 'AAA' is returned. Only works with numbers other an error is returned.
- `valueIfGreaterThanOrEqualTo('AAA','CONDITION','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is greater than or equal to 'CONDITION'. If the phrase of 'AAA' is not found, the original phrase 'AAA' is returned. Only works with numbers other an error is returned.
- `valueIfLessThan('AAA','CONDITION','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is less than 'CONDITION'. If the phrase of 'AAA' is not found, the original phrase 'AAA' is returned. Only works with numbers other an error is returned.
- `valueIfLessThanOrEqualTo('AAA','CONDITION','REPLACEMENT')` – Replaces the tag value with 'REPLACEMENT' if the phrase of 'AAA' is less than or equal to 'CONDITION'. If the phrase of 'AAA' is not found, the original phrase 'AAA' is returned. Only works with numbers other an error is returned.
- `valueCount('AAA','CONDITION')` – Returns the number of times the value of 'CONDITION' is found inside 'AAA'. Uses the '~' character as the delimiter. This is often used in conjunction with the 'getSearchData' tag.
- `opposite('XXX')` – If the supplied value is a number, the sign is reversed. Other value changes are as follows: true->false, false->true, yes->no, no->yes. If none of the previous cases are true, an error is generated.
- `left('XXX','YYY')` – Returns 'YYY' number of characters from the left of the string 'XXX'. Example: `%left('testing123','5')%` will result in 'testi'.
- `right('XXX','YYY')` – Returns 'YYY' number of characters from the right of the string 'XXX'. Example: `%right('testing123','5')%` will result in 'ng123'.
- `substringBefore('AAA').XXX` – Truncates the value after finding the first occurrence of 'AAA'. For example: `"%substringBefore('of').this is a test of substring%"` will result in the final value of "this is a test ". Any additional tag can be provided after the '.' character. Example: `"%substringBefore('-').company name%"` where the "company name" keyword value is "ACME-2010", would result in the final value of "ACME".
- `substringAfter('AAA').XXX` – This only keeps the part of the value after the first occurrence of 'AAA'. For example: `"%substringAfter('of').this is a test of substring%"` will result in the final value of " substringing ". Any additional tag can be provided after the '.' character. Example:

“%substringBefore(‘-’).company name%” where the “company name” keyword value is “ACME-2010”, would result in the final value of “2010”. You can also chain together like this:

“%substringBefore(‘-’).company name%%substringAfter(‘-’).company name%”. Using the previous example, the final result would be “ACME2010”.

- `addToDate(‘XXX’,‘NUMBER’,‘UNIT’)` – Replaces the date value of ‘XXX’ with a new value by adding ‘NUMBER’ of the ‘UNIT’ type. The allowed ‘UNIT’ types are ‘millisecond’, ‘second’, ‘minute’, ‘hour’, ‘day’, ‘week’, ‘month’ and ‘year’. Example: “%addToDate(‘06/10/2013’,‘5’,‘day’)%” will result in a final value of “2015-06-15”. If ‘NUMBER’ is negative, the value will be subtracted. The addition/subtraction is smart, and calculates the new value based on the number of days in a month, number of months in a year, etc.
- `addToDateTime(‘XXX’,‘NUMBER’,‘UNIT’)` – Replaces the date value of ‘XXX’ with a new value by adding ‘NUMBER’ of the ‘UNIT’ type. The allowed ‘UNIT’ types are ‘millisecond’, ‘second’, ‘minute’, ‘hour’, ‘day’, ‘week’, ‘month’ and ‘year’. Example: “%addToDate(‘06/10/2013 03:09:35’,‘6’,‘seconds’)%” will result in a final value of “2015-06-10 03:09:41”. If ‘NUMBER’ is negative, the value will be subtracted. The addition/subtraction is smart, and calculates the new value based on the number of seconds in a minute, number of days in a month, number of months in a year, etc.
- `dateFormat(‘XXX’,‘FORMAT’)` – Replaces the date value of ‘XXX’ with a format specified by ‘FORMAT’. Example: “%dateFormat(‘2011-02-05’,‘mm/dd/yyyy’)%” will result in a final value of “02/05/2011”. Other tags can also be embedded for the value, such as: “%dateFormat(‘%Document Date.value%’,‘dd~mm~yyyy’)%”. The number of weeks, days, etc. that have passed since 0 AD can also be used: `admonths`, `adweeks`, `addays`, `adhours`, `adminutes`, and `adseconds`.
- `decimalFormat(‘XXX’,‘DIGITS’)` – Specifies the number of digits to display after the decimal point using the ‘DIGITS’ value. Example: “%decimalFormat(‘52.489812345’,‘2’)%” will result in a final value of “52.49” (rounded).
- `exists` – Returns a value of true unless used in combination of a ‘parent’ or ‘getChild’ tag that results in it not being found. In other words, if you try to go too high in the parent chain or the specified children don’t exist, it will return false.
- `hasImageMarkups` – Returns true if the file has image markups.
- `hasRedactionImageMarkups` – Returns true if the file has redaction markups.
- `hasTextImageMarkups` – Returns true if the file has text markups.
- `Current Keyword Value.value` – Returns the name of the current keyword value triggering any ‘getPossibleKeywordValues’ triggers. This is useful for populating drop down lists with specific values based on the keyword type. Note: There are spaces in this tag.
- `fileSize` – Returns the number of bytes for the file.
- `volumeSpaceUsed(‘XXX’)` – Returns the space currently used for the specified volume.
- `volumeSpaceFree(‘XXX’)` – Returns the space currently free for the specified volume.
- `volumeSpaceMax(‘XXX’)` – Returns the maximum space quota for the specified volume.

- `previousResult` – The results value of the previous event that fired before this one. This can be very useful to see the results of what happened before, and fire different events based on what happened. For example, a 'Search' event will give results such as '8 objects were found.' or '1500 objects were found.' By utilizing this tag in the 'Field' of the 'Conditions' on an action trigger, one could check to see how many objects were found. A specific example may be: 'Field' is '%previousResults', 'Operator' is 'like', and 'Value' is '100 '. The event would fire if '100 ' objects were found.
- `previousResults` – The same as 'previousResult' except it checks all the previous results for this executing schedule or action trigger. It only provides the values for this thread, not any that occurred from a different request, transaction, or trigger.
- `encrypt('XXX')` – Encrypts the supplied string. Useful for storing credit card numbers, social security numbers, etc. You can embed a tag instead of a constant string. For example: '%encrypt('%SSN.value%')%'. If this example is used in a 'Change Keyword Value' event for a 'saveFolder' trigger, and the folder has a keyword type of 'SSN', the value will be encrypted in the database. Notice the double '%' characters in order to use another tag. This is mandatory if using a tag within a tag.
- `decrypt('XXX')` – Decrypts the supplied string. This works in conjunction with the 'encrypt' tag. Like the 'encrypt' tag, a tag can be embedded for the value. Typically you will use this on a 'getFolder' trigger. An example would be: '%decrypt('%SSN.value%')%'. Any time the trigger occurred, the SSN would be decrypted and the value viewable.
- `isEncrypted` – Returns true if the file is encrypted.
- `hasUnicodeCharacters` - Returns true if the value has at least one character that is Unicode (numeric character value greater than 255).
- `hasNonASCIICharacters` - Returns true if the value has at least one character that is not ASCII (numeric character value greater than 127).
- `hasAsciiCharacters` - Returns true if the value has at least one character that is ASCII (numeric character value between 0 and 127).
- `hasExtendedAsciiCharacters` - Returns true if the value has at least one character that is ASCII (numeric character value between 0 and 255).
- `hasOnlyAsciiCharacters` - Returns true if the all the characters in the supplied value is ASCII (numeric character value between 0 and 127).
- `hasOnlyExtendedAsciiCharacters` - Returns true if the all the characters in the supplied value is ASCII (numeric character value between 0 and 255).
- `math` – Takes the supplied values, and performs a Math evaluation. Parenthesis are accepted as well. Examples:
 - a. `%math('1+5*(3-1)/2')` results in '16.0'%.
 - b. `%math('%payment amount.value%+10')`).
- `evaluate` – This tag takes the supplied parameters and executes the desired Evaluate Event using those values. See the 'Evaluate Event' for more details. Examples:

- a. This example uses an event that reorders the position of the name:
 - i. `%evaluate('Reorder Name','john','doe')%`
- b. This example multiplies the value of the 'Amount' keyword by 2 and returns whatever text the 'My Multiply' event specifies for `%parameter[1]%`.
 - i. `%evaluate('My Multiply','%math('%amount.value%*2')%')%`
- `getSearchData('EVENTNAME','ACTIONTYPE','DELIMITER')` – Tests/fires all the triggers of the Get Search Data event named 'EVENTNAME' and of action type 'ACTIONTYPE'. The resulting 'DELIMITER' is used if there was more than one value returned. The default delimiter is a comma. Example: `'%getSearchData('Get Occupations','noActionEvent','~')%` may result in: Developer~Professor~Mechanic. If you want endline characters between each value, include a `'%endline%'` tag as the delimiter.

System Deployment

Executable and AIR Application Runtime Parameters

It is possible for parameters to be passed to the ES Imaging Client (AIR or EXE) at runtime. In order to do so, simply provide the parameters when executing the file. The following are a list of parameters the system administrator may utilize for end users:

- 1) loginId – The fully qualified login id for the user (including domain). For example: 'acme\john.doe'
- 2) server – The server name and port to connect to. For example: 'www.esimaging.com:9443'.
- 3) environment – The environment to connect to. Typically, each ES Imaging server will allow you to choose one of the following: 'Production' or 'Test'.

Here is an example of a full command line path to the ES Imaging Client executable. Notice that parameter names and values are separated by a ':' character, and parameters that have spaces should be enclosed in double quotes.

```
"C:\Program Files (x86)\ESImagingClient\ESImagingClient.exe" -loginId:acme\john.doe "-server:www.esimaging.com:9443" -environment:Production
```