Perceptive Integration Server

Getting Started Guide

ImageNow Version: 6.7.x

perceptivesoftware

Written by: Product Documentation, R&D
Date: September 2016

Table of Contents

Getting Started with Integration Server

This document introduces Integration Server and provides an overview of its requirements and web services. It also includes a high-level summary of the steps to write your own application using Integration Server. This document is written for developers. For an overview of ImageNow that is intended for developers, refer to the *ImageNow Overview for Developers* document. This document, as well as other developer content, is available on the Customer Portal of the Perceptive Software website at www.perceptivesoftware.com.

What is Integration Server?

ImageNow Integration Server is a service that provides a window into ImageNow functionality. Integration Server exposes a rich set of functions as web services enabling you to embed ImageNow functionality, such as document management and workflow, directly into your application. This document provides you with an overview of these services. For additional information about Integration Server, refer to *Integration Server Help*, which is available as part of the Integration Server product and is also on the Customer Portal of the Perceptive Software website at www.perceptivesoftware.com.

Integration Server is Perceptive Software's new web services framework that supports calls that are lighter weight, easier to build, and return human readable results. Integration Server makes ImageNow content and functionality available to third-party applications. Integration Server enables external customer applications that are coded in standard development languages, such as Java, C++, or C#, and that are also compatible with HTTP Web services, to send and receive data from ImageNow Server.

Integration Server is a middle-tier web service that provides communication over a network between ImageNow software and third-party applications. The architecture supports asynchronous and synchronous communications using standard XML and JSON Representational state transfer (REST) message formats. Integration Server is multi-threaded, which allows for concurrent execution of multiple client requests. For secure client-to-server and server-to-client communication, Integration Server supports SSL.

Third-party applications can use Integration Server to incorporate ImageNow functionality directly into products and enterprise systems by providing the required classes, functions, methods, subroutines, or other routines. Integration Server interacts seamlessly with ImageNow Server to increase efficiency with the use of your existing development tools.

Web services

Integration Server exposes ImageNow functionality through a multiplex of resources, such as Document resources, Folder resources, and Drawer resources. Each web resource includes a family of functions that Integration Server makes available to third-party applications, such as starting a session, copying a document, or routing a document.

Standards

The Integration Server approach to working with client applications is based on widely accepted standards. Integration Server uses a REST approach for Web services and HTTP/HTTPS transport for structured data exchange.

REST is an architectural style for message exchange that addresses the web as remote resources. In a REST application such as Integration Server, each URL points to a resource. This approach differs from SOAP in that SOAP exposes functionality as URL resources that contain functions that can be called. Unlike SOAP applications that are restricted to using GET or POST operations, REST applications include a greater range of operations: GET, POST, PUT, and DELETE.

REST applications are stateless, and no session state is stored on Integration Server. The information required for a request is included in the request message itself. The client application can cache a resource representation, potentially improving application performance.

Development requirements

Integration Server requires a third-party software that is capable of sending and receiving HTTP requests with custom headers, regardless of operating system and development platform. When developing applications for Integration Server, make sure that the software you use is capable of interfacing with Integration Server.

Documentation

The documentation provided with Integration Server includes this guide, the *Integration Server Installation Guide*, and online help. You can find the Integration Server help files in the \integrationserver\help\en directory. To view help, navigate to \integrationserver\help\en located in the web applications folder of your Tomcat web server directory and then double-click Start_IntegrationServer_Help.htm. You can also use the Perceptive Software website at www.perceptivesoftware.com. This website always contains the most current version of the documentation.

About Integration Server licensing

The transaction package licensing model is available for developing applications with Integration Server. For general information about license types, refer to "About licensing in ImageNow" in the Managing ImageNow Licensing section of *Administrator Help*.

Transaction Pack licensing

A transaction pack license counts the number of transactions against Integration Server. ImageNow Server tracks each call made via Integration Server, which is charged against a transaction package license. ImageNow Server controls transaction package licensing. The system counts transactions

licenses based on the number of transactions within ImageNow and decrements the count over a period of time. After users perform the maximum number of transactions allowed in a specified period, users cannot perform any more transactions until the period expires. You can purchase overdraft licensing protection for Integration Server transaction licenses.

Integration Server Transaction Licenses Overdraft Licensing Report

If you purchase overdraft licensing protection for Integration Server transaction licenses, you must generate a quarterly report to send to Perceptive Software. You may also use this report or the ImageNow Server Administrator utility to check the usage status of your overdrafts.

The following Integration Server calls are not charged against a transaction package licensing model.

Call	Function
GET: /connection	Verifies the connection status of Integration Server to ImageNow Server and makes a connection to ImageNow Server if one does not already exist.
DELETE: /connection	Disconnects the connection between Integration Server and ImageNow Server for the given user.
POST: /document	Creates a document.
POST: /document/{id}/page	Adds a page to a document.
POST: /captureGroup	Starts a capture upload.
POST: /captureGroup/{Id}/page	Captures pages to ImageNow documents.
PUT: /captureGroup/{Id}	Ends a capture upload.
PUT: /form/{id}/document/{docId}	Stores form data to a specific version of a document.
PUT: /form/{id}/folder/{folderId}	Stores form data to a folder whose folder type is associated with a given form.

About X-IntegrationServer-Encode-Headers

The values for this custom header are UTF-8 and ISO-8859-1. Use the X-IntegrationServer-Encode-Headers header in calls if the request/response header values can contain Unicode characters. It indicates whether the request/response headers should be base-64 decoded/encoded.

The value of this header indicates the charset to be used when encoding/decoding custom header values. If this header is omitted, the request and response headers will not be encoded/decoded. For more information, refer to “Request Headers” in the Integration Server help.

User session management

This section discusses the methods for establishing a user session. It also includes how Integration Server uses the session hash to monitor licensing.

Establish a user session

You can begin a session with any Integration Server call, but the user credentials are required for every call. You do not need to use the `GET: /connection` call to establish a user session. You can provide the user credentials to Integration Server using one of two methods.

1. Pass the `Authorization` part in the header. This method provides user credentials represented by base-64 encoding that includes the user name and password separated by a colon (for example, `base64("user1:pass1")`).
2. Pass the ImageNow user name with the `X-IntegrationServer-Username` part and the password in the `X-IntegrationServer-Password` part in the header.

If ImageNow Server accepts the user credentials, the response includes the session hash in the header. The session hash is a unique identifier for the session. The subsequent section discusses licensing and how Integration Server uses the session hash to maintain licensing and user sessions.

If ImageNow Server does not accept the user credentials, the header of the response indicates this with the `Status Code`, `X-IntegrationServer-Error-Code`, and `X-IntegrationServer-Error-Message`.

```
Status Code: 401
X-IntegrationServer-Error-Code: LOGIN_FAILED_ERROR
X-IntegrationServer-Error-Message: The system cannot establish a connection. Re-enter
User name and/or Password.
```

Perceptive Software recommends disconnecting the client from a session when finished. If you leave a session open, the client must wait for the server to disconnect, which by default occurs after an hour of inactivity.

How Integration Server uses the session hash to maintain licenses and user sessions

Integration Server uses the session hash to monitor licenses. Once authentication is established, Integration Server provides a session hash in the `X-IntegrationServer-Session-Hash` part of the response header. Integration Server allocates a license when it establishes a new user session. There are a few things to keep in mind when developing regarding the use of the session hash.

- Because the system ties the session hash to the license, it is important to pass the session hash in the header of every request to ensure that Integration Server maintains the user session.
- If you do not pass the session hash, Integration Server assumes that you are establishing a new session and allocates another license.
- For every response to Integration Server, ensure that the session hash you provide in the header matches the session hash that Integration Server provided on the previous call. If you pass in an invalid session hash, Integration Server attempts to establish a new session and provides you with a new session hash.

  The following error code and message typically means that all Integration Server licenses are currently in use and that a new license cannot be provided for the user session.

  ```
  Status Code: 401
  X-IntegrationServer-Error-Code: LOGIN_FAILED_ERROR
  ```

```
X-IntegrationServer-Error-Message: The system cannot establish a connection. A
session management operation failed.
```

- If your application is using transaction pack licenses, the session hash is still required for all transactions, because Integration Server uses the session hash to track transaction pack licenses and to maintain individual user sessions.

Integration Server logging

This section provides a brief overview of Integration Server logging. For a detailed explanation of each logging setting, refer to the integrationserver.ini setting table in the "integrationserver.ini file options" section in the *Integration Server Installation Guide*.

The following settings are available to control Integration Server logging functionality. These settings are located in the [Logging] section of the integrationserver.ini file.

Setting	Description
log.directory	Specifies the path to the directory that contains the log files. The default is C:/inserver6/log.
log.line.pattern	Formats the output of a logging event as a string of literal text. This pattern adheres to the Log4j PatternLayout standard.
log.rolling.interval	Specifies the amount of time data is stored in a log file before the data is archived to another file. This keeps log files from becoming too large. The default is DAY.
log.max.history	Specifies the threshold for the maximum number of log files that are retained per day. One file is created each day, hour, or month depending on the log.rolling.interval setting. The default is 28.
log.level	Specifies the verboseness of the logging that Integration Server uses to log errors for troubleshooting. The default is 2 for error-level logging.
log.level.interceptor	Specifies the level Integration Server uses to log messages from Apache CXF. The default is 2 for error-level logging.
log.level.thirdparties	Specifies the level Integration Server uses to logs messages from third-party libraries, such as the Spring Framework. The default is 2 for error-level logging.

Note The logging level captures and writes all information for that level and the lower levels. When troubleshooting, we recommend turning up the log.level setting gradually until you reach the level you need to resolve the issue.

The following is an excerpt of an Integration Server log file with the log.level setting at 6 for trace-level logging. By default, the log file indicates the log level for each entry. You can change the log.line.pattern value to not include the log level.

```
09:44:39.254 INFO  - Logging configured.
09:44:39.255 DEBUG - Finished creating instance of bean 'logger'
```

```
09:44:39.255 DEBUG - Creating shared instance of singleton bean
'connectionFactoryConfig'
09:44:39.255 DEBUG - Creating instance of bean 'connectionFactoryConfig'
09:44:39.256 DEBUG - Eagerly caching bean 'connectionFactoryConfig' to allow for
resolving potential circular references
09:44:39.256 TRACE - Getting BeanInfo for class
[com.imagenow.apiserver.connection.ConnectionFactoryConfig]
09:44:39.262 TRACE - Caching PropertyDescriptors for class
[com.imagenow.apiserver.connection.ConnectionFactoryConfig]
09:44:39.262 TRACE - Found bean property 'class' of type [java.lang.Class]
09:44:39.262 TRACE - Found bean property 'domain' of type [java.lang.String]
```

Integration Server resources

Integration Server provides the resources listed in the following table. Select calls for some of these resources appear in subsequent sections to familiarize you with Integration Server methods.

For detailed information about the Integration Server calls for each resource, refer to the Start_IntegrationServer_Help.htm file. To locate the Start_IntegrationServer_Help.htm file, navigate to the \integrationserver\help\en folder located in the web applications folder of your web server directory.

Resource	Description
applicationPlan	**Retrieve and request application plans and details. Create folder hierarchy or identify existing folders or drawers based on External application plans.**
GET: /applicationPlan	Requests the application plans stored in ImageNow Server that match the query parameter.
GET: /applicationPlan/{id}	Requests the application plan details that match the given ID.
PUT: /applicationPlan/{id}/location	Uses an External application plan to create a new folder or folder hierarchy, or to identify an existing folder or drawer.
captureGroup	**Capture pages to ImageNow.**
POST: /captureGroup	Creates a capture group, which is the first step in capturing a group of pages into one or more ImageNow documents.
POST: /captureGroup/{id}/page	Captures a page to an existing capture group.
PUT: /captureGroup/{id}	Indicates that the capture process for a capture group is finished.

Resource	Description
captureProfile	**Get capture profile information from your ImageNow system.**
GET: /captureProfile	Requests the list of active capture profiles stored in ImageNow Server.
GET: /captureProfile/{id}	Requests the capture profile stored in ImageNow Server that matches the given ID.
GET: /captureProfile/applicationPlan/itemList/{id}	Requests the context map item list stored in ImageNow Server that matches a given list source type.
captureSourceProfile	**Get capture source profile information from your ImageNow system.**
GET: /captureSourceProfile/{id}	Requests the capture source profile from ImageNow Server that matches the given ID.
connection	**Verify, establish, or delete a connection between Integration Server and ImageNow Server.**
GET: /connection	Verifies the connection status of Integration Server to ImageNow Server and makes a connection to ImageNow Server if one does not already exist.
DELETE: /connection	Disconnects the connection between Integration Server and ImageNow Server for a given user.
digitalId	**Check the validity of a user's digital ID and create a digital ID for a user.**
GET: /digitalId/validity	Check whether the user's digital ID is valid.
POST: /digitalId	Creates a digital ID for the user.
document	**Work with documents in your ImageNow system.**
DELETE: /document/{id}	Requests for ImageNow Server to delete a document that matches the given ID.
DELETE: /document/{id}/page/{pageId}	Deletes a page from the document.
GET: /document/{id}	Retrieves document information for the document that matches the given ID.
GET: /document/{id}/digitalSignature	Requests digital signature information from ImageNow Server for the document that matches the given ID.

Resource	Description
GET: /document/{id}/page	Requests page information from ImageNow Server for the document that matches the given ID.
GET: /document/{id}/page/{pageId}/file	Requests the file contents from ImageNow Server for the page that matches the given ID.
GET: /document/{id}/page/{pageId}/preview	Gets the preview file associated with a page stored in ImageNow Server.
GET: /document/{id}/page/{pageId}/thumbnail	Returns a thumbnail image for a specific page stored in ImageNow Server.
GET: /document/{id}/thumbnail	Returns a thumbnail image for a specific document stored in ImageNow Server.
HEAD: /document/{id}/page/{pageId}/file	Requests the file information associated with a page stored in ImageNow Server, including the ID and the content length.
HEAD: /document/{id}/ page/{pageId}/preview	Requests file information for a preview PNG file of a page stored in ImageNow Server.
HEAD: /document/{id}/ page/{pageId}/thumbnail	Requests file information for a thumbnail PNG file of a page stored in ImageNow Server.
HEAD: /document/{id}/thumbnail	Requests file information for a thumbnail PNG file of a document stored in ImageNow Server.
POST: /document	Creates an empty ImageNow document and indexes the document using the document keys.
POST: /document/{id}/copy	Copies a document stored in ImageNow Server.
POST: /document/{id}/page	Adds a page to an ImageNow document.
POST: /document/{id}/page/{pageId}	Updates the given page in a document.
POST: /document/{id}/page{pageId}/file	Replaces a page in a document.
PUT: /document/{id}	Updates the document name, location, document keys, and properties of a document stored in ImageNow Server.
PUT: /document/{id}/name	Updates the name of a document in ImageNow Server.
documentType	**Get information about the document types in your ImageNow system.**

Resource	Description
GET: /documentType	Requests a list of all document types stored in ImageNow Server.
GET: /documenType/{id}	Requests the associated attributes and properties for the document type stored in ImageNow Server that matches the given ID.
documentTypeList	**Get information about the document type lists in your ImageNow system.**
GET: /documentTypeList	Requests the complete set of document type lists stored in ImageNow Server.
GET: /documentTypeList/{id}	Requests the document type list stored in ImageNow Server that matches the given ID.
drawer	**Get information about the drawers in your ImageNow system as well as move existing items to a drawer.**
GET: /drawer	Gets the list of ImageNow drawers.
GET: /drawer/{id}	Returns detailed information about the drawer stored in ImageNow Server that matches the given ID.
POST: /drawer/{id}/content	Moves a set of existing ImageNow documents, folders, or shortcuts to a drawer.
folder	**Work with folders in your ImageNow system.**
DELETE: /folder/{id}	Deletes a folder and all of its content from ImageNow Server.
GET: /folder/{id}	Requests the associated attributes and metadata for the folder stored in ImageNow Server that matches the given ID.
GET: /folder/{id}/path	Requests the path for a folder.
POST: /folder	Creates a folder in your ImageNow system that is available to store any combination of ImageNow documents, subfolders, and shortcuts.
POST: /folder/{id}/content	Moves a set of existing ImageNow documents, folders, and shortcuts to a folder.

Resource	Description
POST: /folder/searchResult	Requests a list of folders from ImageNow Server that match some given search criteria.
PUT: /folder/{id}	Provides the ability to update the name, folder type, and properties associated with a given ImageNow folder after the folder has been identified through its unique ID.
PUT: /folder/{id}/name	Updates the name of a folder stored in ImageNow Server.
folderType	**Get information about the folder types in your ImageNow system.**
GET: /folderType	Requests the list of folder types stored in ImageNow Server.
GET: /folderType/{id}	Requests the associated capture profile and property definitions stored in ImageNow Server for the folder type that matches the given ID.
form	**Work with forms in your ImageNow system.**
GET: /form	Requests a list of the forms stored in ImageNow Server.
GET: /form/{id}	Requests the form in ImageNow Server that matches the given ID.
GET: /form/{id}/document/{docId}	Requests the form data associated with the given form ID.
GET: /form/{id}/folder/{folderId}	Requests the form data associated with the given form ID and folder.
GET: /form/{id}/presentation	Requests all presentations for the form in ImageNow Server that matches the given ID.
GET: /form/{id}/presentation/{presentationId}	Requests information about all of the files for a presentation associated with the form in ImageNow Server that matches the given ID.
GET: /form/{id}/presentation/{presentationId}/file/{field}	Gets a single physical file that matches the given ID for the presentation associated with the given form.
PUT: /form/{id}/document/{docId}	Requests ImageNow Server to store form data to a specific version of a document for the form that matches the given ID.

Resource	Description
PUT: /form/{id}/folder/{folderId}	Requests ImageNow Server to store form data to the given folder for the form that matches the given ID.
hostedDocument	**Work with hosted documents in your ImageNow system**
GET: /hostedDocument/{clientDocumentId}	Retrieves the ImageNow document ID for the document that matches the document properties within the hosted business application.
GET: /hostedDocument/{clientDocId}/ClientLogob/{clientLogobId}	Retrieves the ImageNow document ID and page ID for the document that matches the document properties within the hosted business application.
licenseGroup	**Get the license group for the ImageNow Server group.**
GET: /licenseGroup	Gets the license group for the ImageNow Server group.
property	**Get information about properties in your ImageNow system.**
GET: /property	Requests all of the property definitions stored in ImageNow Server.
GET: /property/{id}	Requests the property definition stored in ImageNow Server that matches the ID.
reasonList	**Get information about ImageNow reason lists.**
GET: /reasonList/{id}/reason	Requests the reasons in the reason list stored in ImageNow Server that matches the given ID.
serverAction	**Run an iScript.**
POST: /serverAction	Requests ImageNow Server to run the designated iScript.
task	**Work with tasks in your ImageNow system.**
GET: /task/{id}	Requests detailed information about the task in ImageNow Server that matches the given ID.
GET: /task/{id}/comment	Retrieves all comments for the task that matches the given ID.

Resource	Description
POST: /task/{id}/comment	Posts a comment to the task in ImageNow Server that matches the given ID.
POST: /task/{id}/completeAction	Requests the completion of an ImageNow task.
POST: /task/{id}/returnAction	Requests the return of an ImageNow task.
user	**Get lists of ImageNow users and user groups.**
GET: /user	Retrieves the complete list of users in your ImageNow system.
GET: /user/group	Retrieves the list of ImageNow user groups to which a given user belongs.
view	**Work with views in your ImageNow system.**
GET: /view	Gets the list of ImageNow views that match a given category and classifier and that the user has the privileges to access.
GET: /view/{id}	Requests information about the view or view filter in ImageNow Server that matches the given ID.
POST: /view/{id}/result	Runs the view that matches the given ID.
POST: /view/{viewId}/filter/{filterId}/result	Runs a view filter.
POST: /view/{id}/refreshResult	Returns a select set of items when you provide their IDs in the request body.
workflowItem	**Work with workflow items in your ImageNow system.**
DELETE: /workflowItem/{id}	Removes the workflow item that matches the given ID from a workflow.
GET: /workflowItem/{id}	Requests information about the workflow item stored in ImageNow Server that matches the given ID.
POST: /workflowItem	Adds a document or folder to a workflow queue, which creates a workflow item.
POST: /workflowItem/{id}/routingAction	Creates a routing action for a workflow item.
PUT: /workflowItem/{id}/hold	Places the workflow item on hold that matches the given ID.

Resource	Description
PUT: /workflowItem/{id}/state	Updates the state of the workflow item that matches the given ID.
workflowQueue	**Get information about the workflow queues in your ImageNow system.**
GET: /workflowQueue	Requests the list of workflow queues stored in ImageNow Server that the user has the privileges to access.
GET: /workflowQueue/{id}	Gets the workflow queue stored in ImageNow Server that matches the given ID.
GET: /workflowQueue/{id}/form	Requests the list of forms associated with the workflow queue that matches the given ID.
GET: /workflowQueue/{id}/forwardRoute	Requests the list of all possible destinations when forwarding items out of the workflow queue that matches the given ID.

Application Plan resource

The Integration Server Application Plan resource calls allow you to retrieve the application plans stored in ImageNow Server, request specific application plan details, and create a new folder or folder hierarchy, or identify an existing folder or drawer based on an External application plan.

Request application plans

This API call requests the application plans stored in ImageNow Server that match the query parameter. Integration Server structures the response using the `applicationPlan` type, which includes the application plan ID, name, and type, such as DOS.

Request application plan details

This API call requests the application plan details that match the given ID. Integration Server structures the response using the `applicationPlan` type.

Create folder hierarchy

This API call uses an External application plan to create a new folder or folder hierarchy, or identify an existing folder or drawer. The call creates a new folder hierarchy based on the application plan, if all or part of the folder hierarchy does not exist. However, the call does not create a new drawer. The drawer must already exist if the application plan specifies placing new folders or documents below that drawer. The call creates some or all of the hierarchy, as needed, and returns the path information. The call also returns the ID and type of the lowest level of the folder structure. Integration Server requires dictionary

values to determine the location that the application plan uses. The call does not return the dictionary values.

Capture Group resource

You can use the Capture Group resource to capture pages into one or more ImageNow documents. Capturing pages to ImageNow consists of three primary steps, detailed in the following subsections.

Create a capture group

In capturing pages, your first step is to create a capture group by running `POST: /captureGroup`. A capture group is a temporary container for the pages you are adding to ImageNow that only exists during the capture process. Creating a capture group allows you to apply the rules of a capture profile and to distribute the pages to one or more locations within ImageNow. The rules of the capture profile apply to all of the pages you are capturing, such as whether to submit the pages to Content Server or whether to add the pages to a workflow queue.

When you send the `POST: /captureGroup` request, you include the ID of the capture profile that you want to apply to the captured items along with any associated capture source metadata required to run the application plan. Integration Server responds by supplying the suggested location, document keys, and document shortcuts based on the application plan associated with the capture profile. Note that you are not required to use these values when you add the pages to ImageNow.

Capture the pages

After you create the capture group, the next step is to capture the pages into one or more documents by running `POST: /captureGroup/{id}/ page`. While the response of `POST: /captureGroup` provides suggested values for how to store the pages in ImageNow, this call provides ImageNow Server with the actual values for storing the pages. You define the location, document keys, and additional metadata values. These capture values can either be based on the given capture profile or on different values the user provides. This call also includes the binary data of the files for each page.

End the capture

As the final step in the process, you indicate the end of the capture process by running `PUT: /captureGroup/{id}`. At this point, ImageNow Server applies the rules of the capture profile to the captured pages, such as submitting the document to Content Server or adding the document to workflow.

The Connection resource

The Connection resource allows you to verify, establish, or delete a connection between Integration Server and ImageNow Server. Note that it is not necessary to use the `GET: /connection` call to initiate a session with ImageNow Server. You can use any Integration Server call to begin a session, because user

authentication is achieved by the exchange of information in the headers. Perceptive Software recommends ending a session with the `DELETE: /connection` call. By default, Integration Server automatically disconnects a user session after an hour of inactivity.

Verify the connection status

Use the `GET: /connection` call to verify the connection status of Integration Server to ImageNow Server and make a connection to ImageNow Server if one does not already exist. If the connection is active, ImageNow Server returns the Integration Server version number. If there is not an active connection, Integration Server attempts to make a connection to ImageNow Server. If connecting fails, ImageNow Server returns an HTTP error status code, and the return message body is the error message.

Disconnect from ImageNow Server

Use the `DELETE: /connection` call to disconnect the connection between Integration Server and ImageNow Server for the given user session. If the disconnect is successful, ImageNow Server returns an HTTP Response Status Code of 200. If Integration Server cannot contact ImageNow Server to disconnect, ImageNow Server returns a `LOGIN_FAILED_ERROR` code and corresponding error message.

Document resource

The Integration Server Document resource provides operational calls to use as you develop your business application for ImageNow document management functionality. You can use GET calls to retrieve information about ImageNow documents, POST calls to create or copy documents or document pages, PUT calls to update document information, HEAD calls to request document file information, and the DELETE call to delete an ImageNow document.

Get information about a document

Use the `GET: /document/{id}` call to get information for an ImageNow document. The request must include the unique ID of the document. Integration Server returns the detailed information pertaining to the document: the document ID, document name, document keys, version number, and the properties associated with the document. Integration Server also returns data pertaining to each page of the document: the page ID, page name, file name, and file extension.

Get a document page

Use the `GET: /document/{id}/page/{pageId}/file` call to request the file contents from ImageNow Server for the page that matches the given ID. The request must include the ID for the document and the ID for the page within the document that you want to retrieve. ImageNow Server returns the page as a MIME attachment with the content type of application and a subtype of octet-stream (a binary file that opens with an application designated by the host system). Note that in ImageNow, a document might be comprised of multiple files, where each file is a page of the document.

Create a document

Use the `POST: /document` call to create an empty ImageNow document and index the document using the document keys. You can also include the property data to associate with the new document. Integration Server returns the location of the file as a URL if the document creation is successful; otherwise, it returns a naming conflict or other exception.

Add pages to a document

Use the `POST: /document/{id}/page` call to add content to a document. Each page is a MIME attachment with the content type of application and a subtype of octet-stream (a binary file that opens with an application designated by the host system). Perform this call using the unique ID of the ImageNow document for each page you want to add.

The Workflow Item resource

The Integration Server Workflow Item resource allows you to work with workflow items in your ImageNow system. The Workflow Service resource allows you to create workflow items, get workflow items, and delete workflow items. You can also use operations to route workflow items, put workflow items on hold, and update the state of workflow items.

Get information about a workflow item

Use the `GET: /workflowItem/{id}`call requests information about a workflow item. The request must include the unique ID of the workflow item. Integration Server structures the response using the `workflowItem` type, which provides detailed information about the workflow item. The `workflowItem` type includes the `stateInfo` subtype, which provides information about the state of the workflow item.

Add an item to a workflow queue

Use the `POST: /workflowItem` call to add a document or folder to a workflow queue, which creates a workflow item. Integration Server structures the request using the `workflowItem` type, which provides the ID and type of the item, the ID of the workflow queue to place the item in, and the priority level for the item.

Update the state of a workflow item

Use the `PUT: /workflowItem/{id}/state` call to update the state of the workflow item that matches the given ID. The valid states are `IDLE`, `PENDING`, and `WORKING`. Integration Server structures the request using the `stateInfo` type, which provides the state to place the workflow item in, along with comments about the state change.

Error Messages

As you develop your business application using Integration Server, you may encounter situations when an operation fails, causing an error condition. To help you manage these situations, this section provides an overview of the error message format along with descriptions for each of the errors categorized by resource.

All operations can return an error code and error message in the HTTP header instead of, or along with, the expected output when an operation fails. You can use this output to troubleshoot the failure.

```
HTTP/1.1 200 OK
Date: Mon, 10 Apr 2012 14:56:54 GMT
Content-Length: 265
Server: Jetty(7.3.1.v20110307)
Content-Type: application/json
X-IntegrationServer-Session-Hash: 0475b3f6be3065afdd69cfc2d4cb19c878c41e1d9
X-IntegrationServer-Error-Code: INSUFFICIENT_PRIVILEGES_ERROR
X-IntegrationServer-Error-Message: The user has insufficient privileges to return the
specified task.
```

The header includes the following parts related to the error:

Header Part	Description
X-IntegrationServer-Error-Code	A short description of the error.
X-IntegrationServer-Error-Message	A detailed description of the reason for the process error or failure. This is is associated with the accompanying `X-IntegrationServer-Error-Code` part.

The tables in the following sections relate to each Integration Server resource and provide the error codes and descriptions for each error.

General errors

The following table provides the general error codes that Integration Server can return as exceptions for operations across multiple resources along with their descriptions.

Name	Description
CLIENT_ENUM_TRANSFORM_ERROR	This generic error indicates the user provided an invalid enumeration constant to Integration Server.
DATABASE_ERROR	This is a generic error that Integration Server returns when an error occurs in the database.
INSUFFICIENT_PRIVILEGES_ERROR	This generic error indicates that the user does not have the privileges to perform the requested action.

Name	Description
LICENSE_ERROR	This can occur for the following reasons. • The request is attempting to use a license that is not supported by Integration Server. This occurs when an invalid value is passed in the `X-IntegrationServer-Extender-Type` header. • The request is attempting to use a restricted call that is not allowed with the specified license type. • There is an issue with the license.
LOGIN_FAILED_ERROR	An error has occurred while attempting to log on.
MESSAGE_CALL_FAILED_ERROR	This generic error indicates that a failure occurred. The error message text further describes the nature of the error.
MISSING_OR_INVALID_INSTANCE_TYPE_ERROR	An instance type (document, folder, or shortcut) required by the call is missing or invalid. Integration Server can return this error for calls that move items.
UNKNOWN_ERROR	An unknown error has occurred.

Capture Group resource errors

The following table provides the error codes that Integration Server can return as exceptions for Capture Group resource operations along with their descriptions.

Name	Description
INVALID_EMAIL_METADATA_ID_ERROR	The email metadata ID for the capture group page is invalid.

Capture Profile resource errors

The following table provides the error codes that Integration Server can return as exceptions for Capture Profile resource operations along with their descriptions.

Name	Description
APPLICATION_PLAN_NOT_FOUND_ERROR	An application plan with the specified ID value cannot be found.
CAPTURE_PROFILE_NOT_FOUND_ERROR	A capture profile with the specified ID value cannot be found.

Capture Source Profile resource errors

The following table provides the error codes that Integration Server can return as exceptions for Capture Source Profile resource operations along with their descriptions.

Name	Description
CAPTURE_SOURCE_PROFILE_NOT_FOUND_ERROR	A capture source profile with the specified ID cannot be found.
INVALID_CAPTURE_SOURCE_TYPE_ERROR	The specified capture source type does not exist.

Document resource errors

The following table provides the error codes that Integration Server can return as exceptions for Document resource operations along with their descriptions.

Name	Description
CONFLICTING_ITEM_ERROR	This error occurs when you attempt a document or folder operation, such as create, move, copy, or rename, but there is a conflict due to name uniqueness. This can occur, for example, if you try to move a document from one location to another, but a document by the same name already exists in that location. If you specify RETURN_CONFLICTS as the value of the query parameter, Integration Server returns CONFLICTING_ITEM_ERROR. This error occurs for conflicts that the caller can resolve.
DOCUMENT_ALREADY_CHECKED_OUT_ERROR	A user is attempting an operation when the user already has the document checked out.
DOCUMENT_CHECKED_OUT_BY_OTHER_USER_ERROR	A user is attempting an operation when another user already has the document checked out.
DOCUMENT_COPY_TO_SELF_ERROR	An attempt was made to copy a document to the same name and location as itself.
DOCUMENT_HAS_BEEN_DELETED_ERROR	The specified document has been deleted.
DOCUMENT_NOT_FOUND_ERROR	The specified document ID cannot be found.
DOCUMENT_NOT_CHECKED_OUT_ERROR	The requested document operation requires the user to first check out the document.
INVALID_DOCUMENT_VERSION_NUMBER_ERROR	The version number given by the user cannot be parsed. Integration Server returns this error from calls that accept a document version number from the user.
INVALID_MOVE_DESTINATION_ERROR	An invalid destination was supplied when moving items.
INVALID_NAME_ERROR	An invalid name was supplied when renaming a document or folder.

Name	Description
PAGE_NOT_FOUND_ERROR	A failure occurred trying to retrieve a document page.

Drawer resource errors

The following table provides the error codes that Integration Server can return as exceptions for Drawer resource operations along with their descriptions.

Name	Description
DRAWER_NOT_FOUND_ERROR	A drawer with the specified unique ID cannot be found.
MISSING_OR_INVALID_DRAWER_CONTENT_ITEMS_ERROR	When attempting to move items to a drawer, the drawer content items were missing or invalid.

Folder resource errors

The following table provides the error codes that Integration Server can return as exceptions for Folder resource operations along with their descriptions.

Name	Description
ADD_TO_FOLDER_ERROR	An error occurred while attempting to add an item to a folder.
CANNOT_ADD_FOLDER_TO_ITSELF_ERROR	An operation occurred that attempted to add a folder to itself.
CONFLICTING_ITEM_ERROR	This error occurs when you attempt a document or folder operation, such as create, move, copy, or rename, but there is a conflict due to name uniqueness. This can occur, for example, if you try to move a document from one location to another, but a document by the same name already exists in that location. If you specify RETURN_CONFLICTS as the value of the query parameter, Integration Server returns CONFLICTING_ITEM_ERROR. This error occurs for conflicts that the caller can resolve.
FOLDER_NOT_FOUND_ERROR	A folder with the specified ID value cannot be found.
INVALID_FOLDER_SEARCH_CRITERIA_ERROR	The folder search criteria are missing or invalid.
INVALID_NAME_ERROR	An invalid name was supplied when renaming a document or folder.
MISSING_OR_INVALID_FOLDER_CONTENT_ITEMS_ERROR	When moving to a folder, the folder items were missing or invalid.
INVALID_MOVE_DESTINATION_ERROR	An invalid destination was supplied when moving items.

Name	Description
REQUIRED_DATA_INVALID_ERROR	This generic error indicates that at least one of required data for the call is invalid.
REQUIRED_DATA_NOT_SET_ERROR	This generic error indicates that at least one of required data for the call is missing.
RESOURCE_NOT_FOUND_ERROR	This generic error can occur in a number of situations when the resource specified either directly or indirectly by the call does not exist in ImageNow Server.
SERVER_ENUM_TRANSFORM_ERROR	This is a generic, internal error that indicates that an invalid value for an enumeration constant was returned by ImageNow Server to Integration Server.

Property resource errors

The following table provides the error codes that Integration Server can return as exceptions for Property resource operations along with their descriptions.

Name	Description
PROPERTY_NOT_FOUND_ERROR	A property with the specified unique ID cannot be found.

Task resource errors

The following table provides the error codes that Integration Server can return as exceptions for Task resource operations along with their descriptions.

Name	Description
COMPLETE_TASK_PASSWORD_ERROR	When completing a task, the user provided the incorrect password.
EMPTY_ID_ERROR	An attempt was made to complete an operation when the provided task ID is empty.
INVALID_TASK_STATE_ERROR	The task cannot be completed or returned because it is currently in a state that does not permit these operations.
NOT_LATEST_VERSION_NUMBER_ERROR	The call failed because the version number is not the latest version number in your ImageNow system. In completing a task, this assures ImageNow Server that tasks are not completed with respect to dated documents.
TASK_LOCKED_ERROR	The user is attempting to change the state of a task that is locked.
TASK_NOT_FOUND_ERROR	A task with the given ID cannot be found.

Name	Description
USER_NOT_ASSIGNED_TO_TASK_ERROR	The user attempting to complete a task is not the user to whom the task is assigned.

View resource errors

The following table provides the error codes that Integration Server can return as exceptions for View resource operations along with their descriptions.

Name	Description
INVALID_VIEW_CATEGORY_ERROR	The specified view category is invalid for the view call.
VIEW_NOT_FOUND_ERROR	The specified view cannot be found.

Workflow Item resource errors

The following table provides the error codes that Integration Server can return as exceptions for Workflow Item resource operations along with their descriptions.

Name	Description
INVALID_HOLD_REASON_ERROR	The specified hold reason is invalid.
INVALID_WORKFLOW_QUEUE_ITEM_STATE_ERROR	When setting a workflow item state, the specified state was either missing or cannot be parsed.
ITEM_NOT_AVAILABLE_IN_WORKFLOW_QUEUE_ERROR	Returned when the workflow item cannot be found in the specified workflow queue.
MISSING_HOLD_REASON_LIST_ERROR	The user tries to set an item on hold with a reason ID, but the workflow queue hold reason list was not specified.
SET_WORKFLOW_ITEM_STATE_ERROR	The state of a workflow item cannot be set.
WORKFLOW_ITEM_NOT_FOUND_ERROR	The specified workflow item cannot be found.
WORKFLOW_ITEM_STATE_ALREADY_WORKING_ERROR	The user tried to set a workflow item state to `WORKING` when the state of the item is already `WORKING`.
WORKFLOW_QUEUE_NOT_FOUND_ERROR	The specified workflow queue cannot be found.

Workflow Queue resource errors

The following table provides the error codes that Integration Server can return as exceptions for Workflow Queue resource operations along with their descriptions.

Name	Description
INVALID_HOLD_REASON_ERROR	The specified hold reason is invalid.

Index