# Titan

## Installation and Setup Guide

Version: 4.0

Written by: Documentation Team, R&D
Date: January 2025

**Hyland**™

# Documentation Notice

# Table of Contents

# Overview

This document explains how to download and deploy Titan.

# Prerequisites

The following prerequisites are required prior to installing Titan.

- Perceptive Content Integration Server and Apache Tomcat. For more information, refer to the Perceptive Integration Server on Tomcat Installation Guide.

  **Note**  If using Forms with Titan, Titan must reside on the same server as Integration Server.

- OpenID Connect Compatible Identity Provider - Any OIDC provider including Hyland Identity Provider.

- Hyland Preferences Service

- Perceptive Content Rendering and Conversion Service

- File Conversion Service

# Titan

## Secure Environment Recommendations

The following recommendations are intended to help secure the installation environment and apply to all product installations. These recommendations should be followed as a minimum requirement for all Hyland products. The policies of your organization may have additional or more robust requirements that should also be followed.

Hyland products may have additional recommendations described in the specific documentation for that product. In some cases, the recommendations may change or may only apply when using certain Hyland products together in a solution.

- Use TLS for all HTTP traffic, including private network segments. TLS ciphers have to be maintained to stay current over time.
- Use Secure FTP instead of standard FTP for all FTP traffic.
- End-to-end encryption is recommended for all data in transport, independent of a network segment. Note that some regulatory compliance requirements may require end-to-end encryption.
- Change all default passwords before activation of the production system. This applies to Hyland products as well as third-party products used by Hyland products (such as a database server).
- Authorization rules should be configured and tested before activation of the production system. This applies to Hyland products as well as to file system folders and database user accounts.
- Use database encryption for all sensitive data persisted in the database.
- Use file system encryption for all sensitive data and content persisted on the file system.
- Enable encryption when available as part of a subsystem configuration. For example, since ODBC provides the capability to use strong encryption for data, it is recommended to have that option enabled.

## Side-Channel Risks

Consider the following to mitigate side-channel risks:

- Ensure the latest application and operating system patches are applied.
- Ensure the latest firmware patches are applied for any hardware on-premises.

## Download

To download Perceptive product installation files, complete the following steps.

1. Go to the Hyland Community site.

2. From the menu, click **Support** and then under **Software Downloads** select **Perceptive Downloads**.

3. Find and download the installer file corresponding to the version to be installed.

**Note** New and updated documentation and help topics are regularly published to the documentation website at docs.hyland.com.

## Install

To install Titan, complete the following steps.

1. After downloading the *application* zip file to a temporary directory on your computer, create a subdirectory in the **webapps** directory of the Tomcat server. You may name the subdirectory with any name. We recommend using **titan**, which is used in the examples of this document.

   **Note** The default location for the *webapps* directory is **[drive:]\Program Files\Apache Software Foundation\[Tomcat Installation]\webapps\**.

2. Extract the files to the subdirectory you just created.

3. Navigate to the **Tomcat\conf** directory and open the **server.xml** file in a text editor.

4. In the **Host** section, add the **RewriteValve** following the example below.

   ```
   <Host name="localhost"  appBase="webapps" unpackWARs="true" autoDeploy="true">
       ... insert the following line anywhere in this Host tag ...
       <Valve className="org.apache.catalina.valves.rewrite.RewriteValve" />
   </Host>
   ```

5. Save the **server.xml** file and close the text editor.

6. Navigate to the **Tomcat\conf\Catalina\localhost** directory, create a **rewrite.config** configuration file, and then open it in a text editor.

7. Add the following two lines, where **titan** is the name of the subdirectory you created in step 1.

   ```
   RewriteCond %{REQUEST_PATH} !-f
   RewriteRule ^/titan/(.*) /titan/index.html
   ```

   **Note** If you already have Titan deployed in this Tomcat instance then this file will already have one or more entries. You must re-add both lines for the new subdirectory name.

8. Save the **rewrite.config** file and close the text editor.

9. Restart Tomcat.

10. Navigate to the **Titan \webapps\titan** directory.

11. Open the **index.html** file in a text editor.

12. Update the **<base href="/">** tag in line 3 to include the subdirectory name you used in step 1. You must include the leading and trailing forward slashes.

```
<!DOCTYPE html><html lang="en"><head>
  <meta charset="utf-8">
  <base href="/titan/">
```

13. Save the **index.html** file and close the text editor.

## Configure

This section details the configuration process for Titan.

### Configure app.config.json

To configure the app.config.json configuration file, complete the following steps.

1. Navigate to the assets directory and open the **app.config.json** configuration file in a text editor.

   **Note** The default location for the configuration file is **[drive:]\ Program Files\Apache Software Foundation\Tomcat 9.0\webapps\titan\assets\**.

2. In the **authConfig** section, configure the following subsections.

   1. Set the appropriate values, where **<app-host>** specifies the application host you configured in the installation steps.

   ```
   "authConfig": {
     "issuer": "https://<idp-host>/identityprovider",
     "clientId": "titan-app",
     "scope": "openid profile profile.onbase",
     "redirectUri": "https://<app-host>/view/authentication-confirmation",
     "postLogoutRedirectUri": "https://titan/home",
     "requireHttps": true
   },
   ```

   2. Update the **integrationServer.url** and **integrationServer.bearerLoginProfileName** settings to specify the host location for Integration Server and the bearer token login profile name.

   ```
   "integrationServer": {
     "url": "https://<is-host>/integrationserver",
     "applicationId": "titan",
     "bearerLoginProfileName": ""
   },
   ```

   3. Update the **renderingConversionService.url** setting to specify the host location of the Perceptive Content Rendering and Conversion Service.

   ```
   "renderingConversionService": {
     "url": https://<rcs-host>:<port>
   },
   ```

3. Save the **app.config.json** configuration file and close the text editor.

4. Restart Tomcat.

### Configure integrationserver.ini

To configure the integrationserver.ini configuration file, complete the following steps.

1. Navigate to the **[path]/${TOMCAT_HOME}/webapps/integrationserver/WEB-INF** folder.
2. Open the **integrationserver.ini** configuration file in a text editor.
3. In the **[Forms]** section, set **forms.allowed.origins** to the appropriate application origin URL.
4. Save the **integrationserver.ini** configuration file and close the text editor.

## Configure barcode processing

During the capture process, indexing values can be read from barcodes. To assign indexing values from barcodes, complete the following steps.

1. In **Management Console**, in the **External User Application Plan**, click **Modify Row** or double-click the appropriate key. The system opens the **<Key> Attributes** dialog box.
2. Under **General**, complete the following substeps.
    1. From the **Source** list, select **Barcode**.
    2. In the **Value** or **Drawer** field, enter a JavaScript regular expression pattern to locate a barcode match.

        **Note**

        The pattern should be enclosed with slashes, followed by an optional case insensitive flag, **i**. The first barcode on the page that matches is assigned to the key. The first regular expression capturing group is used to populate the key value. The capturing group allows sub-patterns of the barcode value to be used or can surround the pattern to use the entire barcode value.

3. Optional. Under **Advanced**, select any advanced option available for the **Barcode** source type.
4. Click **OK**.

## Regular expression examples

The following table displays a list of regular expression examples.

| Pattern | Barcode | Value |
|---|---|---|
| /^MRN(.+)$/ | MRN123456 | 123456 |
| /^(0+[1-9]+)/ | 001234 | 001234 |
| | 00005678 | 00005678 |
| | 102346 | No match. Must start with one or more leading zeros. |
| /^DT(.+)$/i | DTABC | ABC |
| | dtIgnoreCase | IgnoreCase |
| /^([A-Z0-9]{5})$/ | A1234 | A1234 |
| | 12345 | 12345 |
| | 12p34 | No match. All letters must be uppercase. |
| | A12345 | No match. Must be exactly 5 uppercase letters or numbers |

| /^(?:DT|DocType)(.+)$/ | DTHR | HR |
|---|---|---|
| | DocType1099 | 1099 |

For more information on developing and testing regular expressions, see the websites, https://regexr.com/ or https://regex101.com/.  Ensure that the site is using ECMAScript or JavaScript for the regular expressions.

# Appendix A: Token authentication

A token signing key is required to configure Titan for agent token authentication or Integration Server bearer token authentication. Refer to the Token authentication chapter in the Manage Content help for more information. To configure a token signing key, complete the following steps.

1. In a text editor, open the **inow.ini** configuration file.

2. Under the **[Logon Control]** section, set `token.signing.key.path` to the file path of the private key used to sign tokens. If the private key is encrypted, set `token.signing.key.password` to the private key password. The token signing key must be in Base64 encoded DER (PEM) format.

3. In the same section, set the `token.signing.algorithm` to the algorithm used when signing tokens. Refer to inow.ini configuration file documentation for a list of supported values.

4. Import the certificate or public key associated with the configured private key using either the `import-cert` or the `import-public-key` **intool** commands, specifying `token-auth` for the SSO type. Refer to the **intool** documentation for more information.

5. Save and close the text editor.

# Appendix B: app.config.json file

You can configure Titan by updating the app.config.json file. The following is a list of the sections available in the app.config.json file.

## appInfo

The appInfo section specifies the application information for Titan.

**name**

Specifies the name of the product.

**logoUrl**

Specifies the path to the logo that displays when logging into Titan.

**licenseUrl**

Specifies the path to the license text file for Titan.

```
"appInfo": {
  "name": "Titan",
  "logoUrl": "assets/logo.svg",
  "licensesUrl": "assets/licenses.txt"
},
```

## inactivityTimeoutInMinutes

The inactivityTimeoutInMinutes section specifies the amount of inactivity time, in minutes, that you want the system to wait before logging the user out of Titan.

```
"inactivityTimeoutInMinutes": 30,
```

## authConfig

The authConfig section specifies the configuration options for validating Titan authentication.

**issuer**

Specifies the full URL to the Hyland IDP instance.

**clientID**

Specifies the clientId defined when configuring IDP.

**scope**

Specifies the list of scopes to request when connecting to the Identity Provider.

**redirectUri**

Specifies the URI to redirect to after authenticating with IDP.

**postLogoutRedirectUri**

Specifies the logout URI to redirect users to after logging out of Titan.

**requireHttps**

Specifies whether you want Titan to require https level authentication.

```
" authConfig": {
  "issuer": "https://<idp-host>/identityprovider",
  "clientId": "titan-app",
  "scope": "openid profile",
  "redirectUri": "https://<app-host>/<subdirectory>",
  "postLogoutRedirectUri": "https://<app-host>/<subdirectory>",
  "requireHttps": true
},
```

## loginSubtitle

The loginSubtitle section specifies the message that displays in the login dialog box.

```
"loginSubtitle": "Click login to begin the login process through the identity
provider",
```

## loginLogoUrl

The loginLogoUrl section specifies the URL location for the logo that displays on the login dialog box.

```
"loginLogoUrl": "assets/logo.svg",
```

## displayModules

The displayModules section specifies which Titan modules to display on the application's home page.

```
"displayModules": {
  "capture": true,
```

```
      "documents": true,
      "workflow": true,
      "tasks": true,
      "folders": true
    },
```

## enableDelegate

The enableDelegate section specifies whether to display the delegate user list in the Out of Office pane to the logged-on user when an out of the office event is being enabled.

```
 "outOfOffice": {
      "enableDelegate": true
 }
```

## appSettings

The appSettings section specifies the configuration options for the Titan application. The subsections are described below.

### general

The general section specifies the general configuration settings for Titan.

**pathDelimiter**

Specifies the character used to separate values in the path that displays the folder location.

```
    "general": {
      "pathDelimiter": "\\"
    },
```

### integrationServer

The integrationServer section specifies the configuration options for Integration Server.

**url**

Specifies the URL for Integration Server.

**applicationId**

Specifies the application ID that Titan uses for the Hyland Preferences Service.

**bearerLoginProfileName**

Specifies the bearer token login profile name.

```
    "integrationServer": {
      "url": "https://<is-host>/integrationserver",
      "applicationId": "titan",
      "bearerLoginProfileName": ""
    },
```

### documentViewer

The documentViewer section specifies the configuration options for the Document Viewer in Titan.

**removeFormsPane**

Removes the Forms pane from the viewer.

**removePropertiesPane**

Removes the Properties pane from the viewer.

**useNativePdfViewer**

Specifies whether to use your systems native PDF viewer to view PDF documents.

```
"documentViewer": {
  "removeFormsPane": false,
  "removePropertiesPane": false,
  "useNativePdfViewer": false
},
```

## folderViewer

The folderViewer section specifies the configuration options for the Folder Viewer in Titan.

**removeFormsPane**

Removes the Forms pane from the viewer.

**removePropertiesPane**

Removes the Properties pane from the viewer.

**removeRelatedFoldersPane**

Removes the Related Folders pane from the viewer.

**removeRelatedTasksPane**

Removes the Related Tasks pane from the viewer.

**canEditPageLabel**

Specifies whether the user can edit a page label.

```
"folderViewer": {
  "removeFormsPane": false,
  "removePropertiesPane": false,
  "removeRelatedFoldersPane": false,
  "removeRelatedTasksPane": false,
  "canEditPageLabel": true
},
```

## formViewer

The formViewer section specifies the configuration options for the Forms Viewer in Titan.

**autoloadFirstForm**

Specifies whether the first form in the list displays by default when viewing the Forms pane.

```
"formViewer": {
  "autoloadFirstForm": false
},
```

## propertiesPane

The propertiesPane section specifies the configuration options for the Properties pane in Titan.

**name**

Specifies the display of the Name field in the Properties panes.

**location**

Specifies the display of the Drawer or Path field in the Properties panes.

**keys**

Specifies the display of the Field1 through Field5 fields in the Properties pane.

**type**

Specifies the display of the Type field in the Properties panes.

**notes**

Specifies the display of the Notes pane in the user interface.

**customProperties**

Specifies the display of the Custom Properties pane in the user interface.

**pageLabel**

Specifies the display of the page label in the user interface.

**additionalPageMetadata**

Specifies the display of additional page metadata under Page Properties on the Properties pane in the user interface.

```
    "propertiesPane": {
      "displaySettings": {
        "document": {
          "name": "enabled",
          "location": "enabled",
          "keys": "enabled",
          "type": "enabled",
          "notes": "enabled",
          "customProperties": "enabled",
          "pageProperties": "disabled",
          "pageLabel": "disabled",
          "additionalPageMetadata": "disabled"
        }
      }
    },
```

## renderingConversionService

The renderingConversionService section specifies the configuration options for the Perceptive Content Rendering and Conversion Service.

**url**

Specifies the host location of the Perceptive Content Rendering and Conversion Service.

**scopes**

Specifies the list of scopes to request when obtaining a bearer token from Perceptive Content. This token is used to authenticate with the Perceptive Content Rendering and Conversion Service.

```
    "renderingConversionService": {
      "url": https://<rcs-host>
      "scopes":["rendition-service"]
```

```
      },
```

## search

The search section specifies the configuration options for the search functionality in Titan.

**vslExtendedOperatorsEnabled**

Specifies whether Titan adds the "contains" and "ends with" operators in the list of search operators.

**vslAnyKeyConditionEnabled**

Specifies whether Titan includes the *Any Document Key* condition in the list of search conditions.

**excludeSearchFields**

Specifies what search properties Titan excludes from the list of search conditions. If no values are included in the list, then all search properties are displayed. Only values added to the list are excluded.

**includedCustomPropertySearchFields**

Specifies what custom properties Titan includes in the list of search conditions. Only values added to the list are displayed. If the setting is not in the configuration file, or is set to undefined, then all custom properties are displayed. If the setting is defined with no values included in the list, then no custom properties are displayed.

```
    "search": {
      "vslExtendedOperatorsEnabled": false,
      "vslAnyKeyConditionEnabled": true,
      "document": {
        "excludeSearchFields": ["[drawer]", "[field2]" ],
        "includedCustomPropertySearchFields": ["Prop1", "Prop2"]
      },
      "folder": {
        "excludeSearchFields": ["[drawer]"],
        "includedCustomPropertySearchFields": ["Prop1"]
      },
      "folderContent": {
        "excludeSearchFields": ["[drawer]"],
        "includedCustomPropertySearchFields": ["Prop1"]
      },
      "task": {
        "excludeSearchFields": ["[drawer]"],
        "includedCustomPropertySearchFields": ["Prop1"]
      },
      "workflow": {
        "excludeSearchFields": ["[drawer]"],
        "includedCustomPropertySearchFields": ["Prop1"]
      }
    },
```

## documentPrinter

The documentPrinter section specifies the configuration options for the printer settings in Titan.

**pageSize**

Specifies the paper size printed.

**enableCoverPage**

Specifies if the cover page of a document is to be printed.

**enableAnnotations**

Specifies if the annotations are printed.

```
"documentPrinter": {
  "pageSize": "Letter",
  "enableCoverPage": true,
  "enableAnnotations": true
}
```

## toolbarActionDisplaySettings

The toolbarActionDisplaySettings parameter specifies the conditional display settings for configurable toolbar actions based on the specific module and the screen size of a mobile device such as smart phone. You can apply these settings to various modules, including documents, capture, workflow, tasks, and folders. For instance, when configuring the document module, you can control the visibility of individual toolbar actions for different screen sizes, such as desktop and smallForm (mobile).

Refer to the following sample JSON configuration illustrating this setting:

```
"toolbarActionDisplaySettings": {
  "document": {
    "download-document": {
      "desktop": {
        "isVisible": false
      },
      "smallForm": {
        "isVisible": true
      }
    },
    "create-task": {
      "desktop": {
        "isVisible": true
      },
      "smallForm": {
        "isVisible": false
      }
    }
  },
```

The following table lists actions that you can configure.

| Module / Area / Common Actions | Action Name |
|---|---|
| Document Capture | capture-profile-selector |
| | capture-pages |
| | complete-capture |
| | reset-session |
| Document Viewer | doc-viewer-forms-pane |
| | doc-viewer-properties-pane |
| | doc-viewer-thumbnails-pane |

| Module / Area / Common Actions | Action Name |
|---|---|
| | related-documents-pane |
| Folder Viewer | folder-content-pane |
| | folder-viewer-forms-pane |
| | folder-viewer-properties-pane |
| | related-folders-pane |
| | save-folder |
| | export-grid |
| | move-content |
| Common Actions | annotation-menu |
| | add-page |
| | save-document |
| | download-document |
| | export-pdf-document |
| | print-document |
| | export-email |
| | digital-signature-menu |
| | copy-document |
| | apply-retention-holds |
| | version-control |
| | workflow-actions |
| | create-actions |
| | delete-document |
| | create-task |

## exportEmail

The exportEmail section specifies whether the user can email links to a document and what values are included in the email template.

### to

Specifies the email address of the recipient.

### cc

Specifies a carbon copy of an email when the user wants to include someone other than the direct recipient.

### bcc

Specifies a blind carbon copy of an email when the user does not want the other recipients to see who else is included in the email.

### subject

Specifies a subject of an email where the user can add a short description of the main content.

### preBody

Specifies the section where the user can add any content before a document or a workflow item link.

### postBody

Specifies the section on complimentary closure, such as "Thanks & regards".

```
"exportEmail": {
    "document": {
      "to": "to_recipient@domain.com",
      "cc": "cc_recipient@domain.com",
      "bcc": "bcc_recipient@domain.com",
      "subject": "Perceptive Content Document",
      "preBody": "Below are links to the documents:",
      "postBody": "Regards,"
    },
    "workflow": {
      "to": "to_recipient@domain.com",
      "cc": "cc_recipient@domain.com",
      "bcc": "bcc_recipient@domain.com",
      "subject": "Perceptive Content Workflow Item",
      "preBody": "Below are links to the workflow items:",
      "postBody": "Regards,"
    }
```

## capture

The capture section specifies the settings for the Titan capture module.

### fileMaxFileSize

Specifies the maximum file size, in MB, that you can upload when capturing a document. If the value is left blank then there is no maximum size limit.

## toolkitDowloadLocation

Specifies the URL of where to download the Perceptive Web Scan Toolkit installer if the toolkit is not detected.

## postSaveScriptName

Specifies the post-save iScript file.

## profileNamesToSourceMapping

Specifies mapping between a capture profile name and capture source identifier.

## barcodeSeparatorMapping

Specifies one or more regular expression patterns for identifying separator pages. If any barcode on the page matches one of the patterns, the page is flagged as a separator sheet and not saved. A new group is created and any application plan document keys (field1-field5) set to Barcode are cleared. The groupSeparation setting must be configured.

## barcodeDocSplitMode

Creates new group and clears any application plan document keys (field1-field5) that are set to Barcode when the page has no barcodes, and a previous page had a barcode. The groupSeparation setting must be configured.

## groupSeparation

Specifies group separation options.

**blankPage** - A new group is created for each blank image scanned. Source profile filters and blank page detection must be enabled in the source profile for blank pages to be detected.

**eachScan** - A new group is created on each capture or scan.

**eachPage** - A new group is created for each file captured or image scanned.

```
capture: {
 "fileMaxFileSize: "5",
 "toolkitDownloadLocation: http://127.0,0,1:8090/cloud/PerceptiveWebCapture.zip,
 "postSaveScriptName: "post-save-actions.js"
 "profileNamesToSourceMapping": [
   "/scan/i:scanner-source",
   "/file/i:file-source"
 ],
 "barcodeSeparatorMapping": [
   "^T$|^3$|^HEADER.*"
 ],
 "barcodeDocSplitMode": false,
 "groupSeparation": {
   "blankPage": true,
   "eachScan": true,
   "eachPage": false
 },

};
```

# Appendix C: Perceptive Content Rendering Service authentication

Perceptive Content Rendering Service should be configured to use bearer token introspection as its method of authentication. Titan will send a bearer token obtained from Perceptive Content, containing a set of configured scopes, with each request made to Perceptive Content Rendering Service. To configure Titan to create the bearer token, complete the following steps.

1. Configure a token signing key. Refer to the Appendix A: Token Authentication section for the appropriate steps.

2. In a text editor, open the **app.config.json** configuration file.

3. Under the **renderingConversionService** section, update `scopes`.

   ```
   "renderingConversionService": {
       "scopes": [ "rendition-service" ]
   },
   ```

4. Save the **app.config.json** configuration file.

5. In a text editor, open the Perceptive Content Server's **inserver.ini** configuration file.

6. Under the **[Perceptive Token Management]** section, set `psw.bearer.token.allowed.scopes` to include in this list the of scopes configured above.

7. Configure Perceptive Content Rendering Service to use the bearer token introspection. Refer to the Configure section in the Perceptive Content Rendering and Conversion Service Installation Guide on how to configure the required `Identity:Introspection:ClientToken` and `Identity:Introspection:Endpoint` settings.

8. Save the **inserver.ini** configuration file and close the text editor.