Real-Time Transformation Interface

Administrator's Guide

Version: 2.1.0

Date: 2018-02-23

KOFAX

Table of Contents

Preface

This guide provides information about installing and using the Real-Time Transformation Interface. This product allows access to Kofax Transformation Modules projects through a Web service or interface. Kofax Transformation Modules perform optical character reading, classify documents, and extract critical data on images that a user transmits through the Real-Time Transformation Interface.

Related documentation

- Documentation for this product is available online.
- *Real-Time Transformation Interface Release Notes*Contains information about resolved and known issues in Real-Time Transformation Interface.
- *Kofax Mobile SDK Developer's Guide* Contains essential information about installing and configuring the Kofax Mobile SDK.
- The Kofax Transformation Modules set contains information about installing, configuring, and using this product.

Training

Kofax offers both classroom and computer-based training that will help you make the most of your Kofax Capture solution. Visit the Kofax website at www.kofax.com for complete details about the available training options and schedules.

Getting Help for Kofax Products

Kofax regularly updates the Kofax Support site with the latest information about Kofax products.

Documentation for this product is available online.

To access some resources, you must have a valid Support Agreement with an authorized Kofax Reseller/Partner or with Kofax directly.

Use the tools that Kofax provides for researching and identifying issues. For example, use the Kofax Support site to search for answers about messages, keywords, and product issues. To access the Kofax Support page, go to www.kofax.com/support.

The Kofax Support page provides:

- Product information and release news

  Click a product family, select a product, and select a version number.
- Downloadable product documentation

Click a product family, select a product, and click **Documentation**.

- Access to product knowledge bases

  Click **Knowledge Base**.
- Access to the Kofax Customer Portal (for eligible customers)

  Click **Account Management** and log in.

  To optimize your use of the portal, go to the Kofax Customer Portal login page and click the link to open the *Guide to the Kofax Support Portal*. This guide describes how to access the support site, what to do before contacting the support team, how to open a new case or view an open case, and what information to collect before opening a case.
- Access to support tools

  Click **Tools** and select the tool to use.
- Information about the support commitment for Kofax products

  Click **Support Details** and select **Kofax Support Commitment**.

Use these tools to find answers to questions that you have, to learn about new functionality, and to research possible solutions to current issues.

Chapter 1

Overview

The Real-Time Transformation Interface adds a Web service to your Kofax Transformation Modules solution. You can configure the Web service to support different Kofax Transformation Modules projects, which accelerates the processing time and allows Kofax Transformation Modules to quickly perform optical character reading, classification, and extraction on submitted documents.

You can install the Real-Time Transformation Interface on multiple servers. The interface works with multiple Kofax Transformation Modules projects.

The Real-Time Transformation Interface supports:

- Installation using an interactive wizard or quiet mode installation. See Installing the Real-Time Transformation Interface on page 9.
- Adding performance counters. See Adding performance counters on page 19.
- Access to the Kofax Mobile SDK, which guides users through image capture.
- Integration with Smart Mobile Components like Kofax Mobile Bill Pay and Kofax Mobile Deposit Capture (for more information, see the related component guide).
- Multiple image files as input, including multi-page TIFF images, and single-page TIFF, JPG, PNG, and PDF images.
- AppStats Reporting and Events. The AppStats reporting service can be accessed at: `http://localhost/mobilesdk/api/AppStats`

Kofax Transformation Modules provide limited processing for projects submitted through the Real-Time Transformation Interface and do not support the following:

- Online learning, even when configured for the project.
- Batch-driven events other than `Batch_Close`.
- Foldering.
- Table fields.
- Interactive module configurations, like thin or thick clients on mobile devices.

Note Configuration examples in this document were specifically formatted for readability in the document format and may not work "as is" in actual configuration files. Therefore, do not copy and paste the configuration samples. Doing so can have an impact on processing.

Supported file formats

File Format	RTTI	Server Side IP[1]	Kofax Transformation Modules[2]
PDF	Yes	No	Read only

File Format	RTTI	Server Side IP[1]	Kofax Transformation Modules[2]
JPEG	Yes	Input & output	Yes
GIF	No	No	No
TIFF	Yes	Input & output	Yes
BMP	No	No	No
PNG	Yes	Input & output	Yes
Text	No	No	Yes

[1] use the OutputFormat request parameter to specify the file type of the processed image (TIFF is default).

[2] see the Kofax Transformation Modules help (Use Project Builder > Supported File Types) for details.

Setting the content type

The Real-Time Transformation Interface detects the content type when you upload a file through a Web service request. The Kofax Mobile SDK includes an Image object property called `ImageMimeType`, which identifies the content type of the image.

Use the following values when setting the content type:

- For TIFF images, use *image/tiff*.
- For JPG images, use *image/jpeg*.
- For PNG images, use *image/png*.
- For PDF files, use *application/pdf*

Chapter 2

Installing the Real-Time Transformation Interface

This chapter describes installation requirements and how to use the interactive wizard or quiet mode to install the Real-Time Transformation Interface. If you are upgrading your version of the Real-Time Transformation Interface, use the same procedure.

System requirements

Server hardware requirements

Hardware requirements for installing the Real-Time Transformation Interface vary based on the following conditions:

- The project being processed.
- The number of concurrent requests that the hardware needs to support.

OCR, classification, and extraction are CPU-intensive activities to consider during planning.

Software requirements

The Real-Time Transformation Interface requires that the following software is installed on the Web server:

- .NET 4.6.2.
- Internet Information Services (IIS) 7.5 or later with ASP.NET.
- Kofax Capture 9.0, 10.0, 10.1, or 10.2; customers with Kofax Capture 9.0 must update the system to the latest service pack and fix pack.
- Kofax Transformation Modules 5.5.2 with Fix Pack 17 or above, or Kofax Transformation Modules 6.0 or above.
  - Install Kofax Capture and Kofax Transformation Modules on each Web server onto which you will install the Real-Time Transformation Interface.
- Microsoft Visual C++ Redistributable Packages for Visual Studio 2013 version 12.0.30501.0, both x64 and x86. Note that these packages are optionally installed by the installation wizard.
- Microsoft Visual C++ Redistributable Packages for Visual Studio 2015 version 14.0.23026.0, both x64 and x86. Note that these packages are optionally installed by the installation wizard.

Use the Kofax Mobile SDK to create an app for users to download onto each mobile device that will use the Real-Time Transformation Interface.

Note By default, Kofax Transformation Modules restrict a user from submitting an image that has a dpi less than 96. Attempting to do so returns an exception error. To keep this from happening, add the following to your mobile app **Image.setDPI()**. Between the parentheses, enter a number greater than 96. The mobile app must call this method before sending the image to the Real-Time Transformation Interface.

User access requirements for installation and execution

- To install the Real-Time Transformation Interface, you must have full Windows administrative privileges.
- To install the Real-Time Transformation Interface, you must have read access to the Kofax Capture shared license configuration file.
- To configure the Real-Time Transformation Interface, you must have access rights to the storage location of the Kofax Transformation Modules projects being set up.
- To use the Real-Time Transformation Interface, you must have access rights to the storage location of the data logging and debug folders configured in the `Web.config` file.

Using multiple servers and load balancers

- To use multiple Web servers, install Kofax Transformation Modules on each server.
- To use the Real-Time Transformation Interface in conjunction with a load balancer such as Windows NLB, configure each server to process the same projects.

Windows roles

The Real-Time Transformation Interface requires the following Windows server roles:

- .NET Extensibility 3.5
- .NET Extensibility 4.5
- ASP
- ASP .NET 4.5
- ISAPI Extensions
- ISAPI Filters

Using the installation wizard

1. On the Web server where Kofax Transformation Modules are installed, shut down any running applications, including the Control Panel, virus detection software, and toolbars.
2. Start the installation by going to the folder where you extracted the installation deliverables and executing the **KofaxRTTI.exe** file.
3. Select **Real-Time Transformation Interface** and click **Next**.
4. Follow the instructions in the wizard to accept the license and select a destination location.
5. Click **Install**.
6. When the installation is complete, click **Finish** to exit the wizard.
7. Repeat this procedure on each Web server on which to run the Real-Time Transformation Interface.

Note If installing the Real-Time Transformation Interface on multiple servers, use the quiet mode installation procedure or the `KofaxRTTI.msi` file to automate the installation.

Using the quiet-mode installation procedure

When you use quiet-mode installation, the installer does not display windows or prompt you for input during the installation process. However, the installer does display error messages. Because you can use the same installation command on multiple servers, use the quiet installation procedure to automate identical installations on multiple servers.

1. On the Web server where Kofax Transformation Modules are installed, shut down any running applications, including Control Panel, virus detection software and toolbars.
2. Open a command-line window and navigate to the directory where you extracted the installation software.
3. Enter a command similar to the following:
   KofaxRTTI.exe /s /v"/q"

Uninstalling the Real-Time Transformation Interface

1. From the Control Panel, select **Add or Remove Programs**.
2. Select the **Show Updates** check box.
3. Select **Real-Time Transformation Interface** and click **Remove**.
4. In IIS Manager, locate and remove the application.

Installing and uninstalling with only the MSI file

To install this product using only the included MSI file, use the following procedures.

Note If this method is selected, the installation will not check for prerequisites, and the necessary prerequisite files will not be automatically installed. It is up to the user to ensure all the prerequisites are in place before attempting to install using this method.

Install on a system without Real-Time Transformation Interface

Follow these instructions to install on a system without Real-Time Transformation Interface installed:

1. Open a Command Prompt window as an Administrator.
2. Navigate to the location of the `KofaxRTTI.msi` file.
   - To install normally: `msiexec.exe /i KofaxRTTI.msi`
   - To install silently: `msiexec.exe /i KofaxRTTI.msi /qn`

Install on a system with a previous version of Real-Time Transformation Interface

Follow these instructions to install this fix on a system with a previous version of Real-Time Transformation Interface installed:

1. Open Internet Information Services (IIS) Manager.
2. Navigate to the Web site containing the `mobilesdk` application.
3. Stop the Web site.
4. Open a Command Prompt window as an Administrator.
5. Navigate to the location of the `KofaxRTTI.msi` file.
   - To install normally: `msiexec.exe /i KofaxRTTI.msi`.
   - To install silently: `msiexec.exe /i KofaxRTTI.msi/qn /norestart`.

   Note The norestart switch is optional; it prevents the server from automatically rebooting after the silent upgrade.
6. Return to IIS Manager.
7. Start the Web site stopped in Step 3.

Uninstallation instructions

Follow these instructions to uninstall this version of Real-Time Transformation Interface:

1. Open Internet Information Services (IIS) Manager.
2. Navigate to the Web site containing the `mobilesdk` application.
3. Stop the Web site.
4. Open a Command Prompt window as an Administrator.
5. Navigate to the location of the `KofaxRTTI.msi` file.
   - To uninstall normally: `msiexec.exe /x KofaxRTTI.msi`
   - To uninstall silently: `msiexec.exe /x KofaxRTTI.msi /qn`
6. Return to IIS Manager.
7. Start the Web site stopped in Step 3.

Chapter 3

Configuring the Real-Time Transformation Interface

After installation is completed, run the Internet Information Services (IIS) Manager as the administrator to configure the Real-Time Transformation Interface Web service.

Note Using an unregistered version of .NET generates an error. To register your version, see the Microsoft article http://msdn.microsoft.com/en-us/library/k6h9cz8h(VS.100).aspx.

Application Pool

The Real-Time Transformation Interface application pool is an entity within Internet Information Services (IIS) that routes Web requests via one or more IIS worker processes (windows process `w3wp.exe`) to the Real-Time Transformation Interface web application. After installing Real-Time Transformation Interface for the first time, a Real-Time Transformation Interface application pool must be created in IIS.

Basic Settings

When creating the Real-Time Transformation Interface application pool, set the following values in the "Add Application Pool" dialog:

- Name - any user-defined name.
- .NET Framework version - .NET Framework v4.0.
- Managed pipeline mode - Integrated.
- Set "Start application pool immediately."

Advanced Settings

Once the Real-Time Transformation Interface application pool is created, open its "Advanced Settings" dialog and set the following values:

- Enable 32-Bit Applications - True.
- Start Mode - AlwaysRunning.
- Identity - set to a user with rights to access the locations where your store project files.
- Idle Time-out - 0.
- Load User Profile - False.

Recycling

Recycling the Real-Time Transformation Interface application pool means that the worker process (`w3wp.exe`) is terminated and a new one started. This is a standard practice in IIS, and there are many settings to customize and fine-tune this action. The values listed below are the default values (unless noted), and can be changed to suit the customer environment.

- Disable Overlapped Recycle - True
  - The recycle will happen such that the existing worker process exits before another worker process is created. This setting must be "True".

- Disable Recycling for Configuration Changes - True
  - If true, the Real-Time Transformation Interface application pool will not recycle when its configuration is changed, meaning a manual recycle must be done to activate configuration changes. This setting must be "True".
- Private Memory Limit (KB) - 0
  - Maximum amount of private memory the worker process can consume before causing a recycle. Zero means there is no limit.
- Regular Time Interval (minutes) - 1740
  - Period of time after which the Real-Time Transformation Interface application pool will recycle. Zero means it does not recycle on a regular interval. It is recommended to set this value to zero and configure specific recycle times
- Request Limit - 0
  - Maximum number of requests the Real-Time Transformation Interface application pool can process before it is recycled. Zero means it can process an unlimited number of requests.
- Virtual Memory Limit (KB) - 0
  - Maximum amount of virtual memory the worker process can consume before causing a recycle. Zero means there is no limit.
- Specific Times
  - A set of specific local times (in 24-hour format) when the Real-Time Transformation Interface application pool is recycled. It is recommended to configure specific recycle times.

Event Logging

There are multiple settings that will trigger IIS to generate informational entries in the Windows System event log. The source of these entries is the IIS Windows Process Activation Service (WAS). These entries can be used to diagnose any issues with the Real-Time Transformation Interface application pool, as well as fine-tune the Real-Time Transformation Interface application pool's advanced settings.

- Idle Time-out Reached
  - An event log entry is generated when the Real-Time Transformation Interface application pool is shut down after exceeding its idle time-out limit.
- Application Pool Configuration Changed
  - An event log entry is generated when the Real-Time Transformation Interface application pool recycles due to a change in its configuration.
- ISAPI Reported Unhealthy
  - An event log entry is generated when the Real-Time Transformation Interface application pool recycles because an ISAPI extension has reported itself as unhealthy.
- Manual Recycle
  - An event log entry is generated when the Real-Time Transformation Interface application pool has been manually recycled.
- Private Memory Limit Exceeded
  - An event log entry is generated when the Real-Time Transformation Interface application pool recycles after exceeding its private memory limit.
- Regular Time Interval
  - An event log entry is generated when the Real-Time Transformation Interface application pool recycles on its scheduled interval.

- Request Limit Exceeded
  - An event log entry is generated when the Real-Time Transformation Interface application pool recycles after exceeding its request limit.
- Specific Time
  - An event log entry is generated when the Real-Time Transformation Interface application pool recycles at a scheduled time.
- Virtual Memory Limit Exceeded
  - An event log entry is generated when the Real-Time Transformation Interface application pool recycles after exceeding its virtual memory limits.

IIS Web Application

The Real-Time Transformation Interface web application is an entity within Internet Information Services (IIS) that provides services over HTTP. After installing Real-Time Transformation Interface for the first time, a Real-Time Transformation Interface application must be created in IIS.

Basic Settings

When creating the Real-Time Transformation Interface application, set the following values in the "Add Application" dialog:

- Alias - any user-defined name; this name will be part of the URL.
- Application pool - the Real-Time Transformation Interface application pool created above.
- Physical path - the Real-Time Transformation Interface installation folder (default = `C:\Program Files\Kofax\RTTI`).
- Connect as… - set to a user with rights to access the locations where your store project files.
- Enable Preload - checked

Note The Enable Preload setting works in conjunction with the Application Initialization feature of IIS (built-in for IIS 8.0, module for IIS 7.5), which enables IIS to initialize the Real-Time Transformation Interface application. In IIS 8.0, this feature is installed via the "Server Manager", and can be found under the *Application Development* sub-feature.

Edit Web.config

For each project that you will process, set up a project element as described in the following procedure. The Real-Time Transformation Interface lets you work with multiple projects, however, you must set up a separate project element for each. If you use the `warmupImage` feature, you should also configure Application Initialization for IIS to prevent performance slow-downs.

1. Navigate to the folder where the Real-Time Transformation Interface files are installed.
2. Open the `web.config` file.
3. Scroll to the bottom of the file and locate the `<processingSettings>` section.
4. (Optional) Set a logging level to track project activity and the interval at which the system clears the logging folders.
   - `loggingOption`—indicates which transactions to preserve. Accepted values include: `ErrorsOnly`, `All`, `None`.
   - `cleanupTimeSpan`—indicates how long to keep data logging folders. Folders older than this value are deleted once an hour.

- `folder`—indicates the log location and must be writable by the application pool user. The default value is `[CommonApplicationData]\Kofax\Real-Time Transformation Interface \DataLogging`.

For example:

```
<dataLogging loggingOption="All" cleanupTimeSpan="7.00:00:00"
folder="C:\ProgramData\Kofax\Real-Time Transformation Interface\DataLogging" />
```

The logging option you set applies to all projects that you set up.

5. (Optional) To enable the AppStats feature, in the `<processingSettings>` section, specify the locations of a user-written plugin and/or the file containing the JSON formatted AppStats data:

```
<appStats appStatsPlugin="\\localhost\CaptureSV\AppStats\Plugin.dll"
          appStatsFolder="\\localhost\CaptureSV\AppStats" />
```

- `appStatsPlugin` is the full path to the user-written plug-in for processing device and Real-Time Transformation Interface app stats.
- `appStatsFolder` is the folder path for writing JSON files containing device-side app stats.

The project mapping `AppStats` will be reserved for this feature to allow the Real-Time Transformation Interface server administrator to configure the process pool settings `processPoolInitialSize` and `processPoolThreshold`:

```
<project projectMapping="AppStats"
               projectFile=""
               processPoolInitialSize="2"
               processPoolThreshold="1" />
```

To lift the security restriction when using a UNC path, add the following to `Kofax.MobileTransformation.Worker.exe.config`:

Note Please note that due to .NET default security restrictions, the plugin library must be located in a local folder on the Real-Time Transformation Interface server.

```
<configuration>
  <runtime>
    <loadFromRemoteSources enabled="true" />
  </runtime>
</configuration>
```

6. Create a project element for each project you will process using the following format:

```
<projects>
  <project projectMapping="BillPay"
    projectFile="\\localhost\CaptureSV\Projects\BillPay\Project.fpr"
    ipProfile="\\localhost\CaptureSV\Projects\BillPay\BillsProfile.txt"
    processPoolInitialSize="2"
    processPoolThreshold="1" />

  <project projectMapping="CheckDeposit"
    warmupImage="\\localhost\CaptureSV\Projects\CheckDeposit\
      Sample_Images\Laura_Wilson_Check_Front.tif"
    projectFile="\\localhost\CaptureSV\Projects\CheckDeposit\
      Kofax_Check_Deposit.fpr"
    ipProfile="\\localhost\CaptureSV\Projects\CheckDeposit\Checks.txt"
    processPoolInitialSize="4"
    processPoolThreshold="0" />
```

```
</projects>
```

The example shows two project elements, one to process bill payments (Kofax Mobile Bill Pay) and another to process check deposits (Kofax Mobile Deposit Capture).

- project `projectMapping`—Provides a name for the project and lets you build the URL for the project.
- `ipProfile`—(Optional; add this parameter to use image processing with the project.) Enter a fully qualified path of the file containing the imaging processing profile. For more information, see Sample Real-Time Transformation Interface image processing profile.

  Note If you update the image perfection settings in the `ipProfile` on the server, verify that you use the correct syntax and numbers. Entering an incorrect syntax or number generates an error.

- `warmupImage`—(Optional; use with Application Initialization enabled.) When the Real-Time Transformation Interface Web application starts, it starts the process pool, and loads the specified Kofax Transformation Modules project. If you use `warmupImage`, Real-Time Transformation Interface processes the image after loading the project. This improves the speed of processing the first request that comes in because the initial items are already started. The data results of processing the warm-up image are discarded.
- `projectFile`—Add the full path to the Kofax Transformation Modules project file.
- `processPoolInitialSize`—The initial number of worker processes available for performing transformation on a given project.
- `processPoolThreshold`—The minimum number of idle worker processes available for a given project.

To distribute Real-Time Transformation Interface processing across a number of Web servers, put your projects in a common location, for example, in the Kofax Capture share. Then, map the project elements in the config file to that location, for example, `\\Server\CaptureSV\KtmProjects`.

Example of `processingSettings` with `dataLogging` and projects:

```
<processingSettings>
 <dataLogging loggingOption="All" cleanupTimeSpan="7.00:00:00" />
  <projects>
    <project projectMapping="BillPay"
      projectFile="\\localhost\CaptureSV\Projects\BillPay\Project.fpr"
      processPoolInitialSize="2"
      processPoolThreshold="1" />

    <project projectMapping="CheckDeposit"
      warmupImage="\\localhost\CaptureSV\Projects\CheckDeposit\
        Sample_Images\Laura_Wilson_Check_Front.tif"
      projectFile="\\localhost\CaptureSV\Projects\CheckDeposit\
        Kofax_Check_Deposit.fpr"
      processPoolInitialSize="4"
      processPoolThreshold="0" />
  </projects>
</processingSettings>
```

Configuring for security

The system administrator can configure IIS authentication as needed to add security to mobile processing. The Real-Time Transformation Interface supports SSL and the following authentication types:

- Anonymous Authentication
- Basic Authentication
- Client Certificate Authentication

ASP.NET transport security

ASP.NET provides configuration options, which allow developers to define the transport security level for each communication binding. Messages transmitted without the proper transport security level cannot be guaranteed to be confidential. If a WCF security binding is set to "none" both transport and message security are disabled.

Real-Time Transformation Interface uses the binding `NetNamedPipeBinding`, which only supports two security modes, None or Transport.

In the `Web.config` file verify that the security mode for the binding Binding1 is Transport. For example:

```
  <system.serviceModel>
...
    <bindings>
      <netNamedPipeBinding>
        <binding name="Binding1">
          <security mode = "Transport">
          </security>
        </binding>
      </netNamedPipeBinding>
    </bindings>
...
  </system.serviceModel>
```

In the `Kofax.MobileTransformation.Worker.exe.config` file verify that the security mode for the binding `NetNamedPipeBinding_IWorkerHost` is Transport. For example:

```
    <system.serviceModel>
...
        <bindings>
            <netNamedPipeBinding>
              <binding name="NetNamedPipeBinding_IWorkerHost"
...
                       maxReceivedMessageSize="2147483647">
                <security mode="Transport" />
              </binding>
            </netNamedPipeBinding>
        </bindings>
...
    </system.serviceModel>
```

Denial of service conditions

A potential Denial of Service (DoS) condition vulnerability may exist in cases where proper throttling protections are not put in place within the application configuration file.

In the `Web.config` file verify that `serviceThrottling` is set in the `MobileKtmServiceBehavior`. For example:

```
  <system.serviceModel>
...
    <behaviors>
      <serviceBehaviors>
        <behavior name="MobileKtmServiceBehavior">
...
          <serviceThrottling />
          <!--
            Defaults for service throttling:
              maxConcurrentCalls = 16 * processor count
              maxConcurrentSessions = 100 * processor count
              maxConcurrentInstances = maxConcurrentSessions + MaxConcurrentCalls
          -->
        </behavior>
      </serviceBehaviors>
    </behaviors>
...
  </system.serviceModel>
```

Web server returns version in HTTP header

Disclosure of server version numbers can allow an attacker or automated attack tool to easily identify outdated servers and target specific vulnerabilities associated with those versions. In the event of a zero-day exploit announcement, attackers may be able to use this information to quickly identify vulnerable systems before patches can be deployed. While suppressing banner information should not be relied upon as the sole mechanism of defense, it can be used to complement an otherwise secure environment.

For `web.config`, verify that `enableVersionHeader` is false. For example:

```
  <system.web>
...
    <httpRuntime targetFramework="4.5" maxRequestLength="102400" executionTimeout="300"
 enableVersionHeader="false"/>
...
  </system.web>
```

Adding performance counters

The Real-Time Transformation Interface supports the following Windows Performance Monitor counters:

- Request Count
- Requests Executing
- Failure Count
- Request Execution Time

- Average Request Duration
- Average Request Duration (Last Minute)
- Worker Processes
- Worker Processes Available

Restarting the application resets these counters.

Locating information about the API

For information about the Web service calls and methods available in the Real-Time Transformation Interface API, browse to the URL where you installed the product and access the Help. For example, browse to http://localhost/rtti/help, where "rtti" is the Web application that you created during configuration.

Returning HTTP 200 responses for all requests

The enhanced error reporting feature directs Real-Time Transformation Interface to return HTTP 200 responses for most requests. When enabled, most requests sent to Real-Time Transformation Interface will return an HTTP 200 response. One exception to this (for example), is sending a **PUT** request with an unsupported media type, such as `image/gif`, which will still generate an HTTP 500 error.

A **GET** request in the format `http://localhost/mobilesdk/api` returns version information and the current setting of enhanced error reporting:

```
{
 "productName": "Kofax Real-Time Transformation Interface",
 "copyright": "Copyright © 1994-2015 Kofax, Inc. U.S. Patent No. 6,370,277 All rights
 reserved. Use is subject to license terms.",
 "productVersion": "1.7.0.0.0.57",
 "kofaxTransformationModulesVersion": "6.1.0.0.0.3930",
 "kofaxCaptureVersion": "10.1.0.1.0.800",
 "enhancedErrorReporting": false
 "databaseLoggingEnabled": false
}
```

With each HTTP 200 response, three additional fields are returned. These new fields are `Result`, `ErrorType`, and `ErrorDescription`.

- `Result` can have one of these values: Success, Error, or Exception
- `ErrorType` can have one of these values: None, System, Parameter, Image, Classification, ImageProcessing, Extraction, Project, Validation, or Recognition
- `ErrorDescription` can either be empty or contain a textual description of the error detected

When `Result` is Success, `ErrorType` is None and `ErrorDescription` is empty.

An example of a non-success result:

```
{
 "extractionClass": null,
 "classificationResult": null,
 "fields": null,
 "sessionKey": null,
```

```
 "documentId": "",
 "words": null,
 "result": "Exception",
 "errorType": "System",
 "errorDescription": "No image filter for the specified file format available."
}
```

Note See Error types and descriptions for details about error types and descriptions.

Processing images from multiple sources: one profile

Real-Time Transformation Interface can automatically determine the type of source image (mobile device camera, desktop scanner or MFP device) based on image analysis. Real-Time Transformation Interface supports two approaches for making this determination: the one-profile approach (described in this section), and the two-profile approach (described in the next section).

The single profile approach (preferred) is easier to configure and maintain. In this approach, the administrator configures a single profile to handle both mobile camera type images and scanner or MFP type images.

Enable this by adding the keyword `_DeviceType_-1` to the `imageperfectionsettings` string of the image processing profile file referenced by the `ipProfile` property that you set in the `web.config` file for your Kofax Transformation Modules project. When Real-Time Transformation Interface determines an image is from a scanner or MFP, Real-Time Transformation Interface automatically adjusts image processing parameters internally so that document cropping, if requested in the profile, is performed reliably.

Usage of the `ipProfile` property is described in the section Sample Real-Time Transformation Interface image processing profile.

Processing images from multiple sources: two profiles

Real-Time Transformation Interface can automatically determine the type of source image (mobile device camera, desktop scanner or MFP device) based on image analysis. Real-Time Transformation Interface supports two approaches for making this determination: the two-profile approach (described in this section), and the one-profile approach (described in the previous section).

The one-profile approach is preferred and should work in most situations. However, if you need to customize profiles individually for best results with your image set, then you should use the two-profile approach. In this approach, the administrator configures one profile for mobile camera type images and another profile for scanner or MFP type images.

Enable this by adding `autoProfileDetect="On"` in the `<processingSettings>` section of the `web.config` file as described in the "Multiple Image Processing Profiles" section.

Multiple image processing profiles

The Real-Time Transformation Interface can support multiple image processing profiles (IPP), and allows the user to choose which IPP to use based on either the Web request or the project element in the `web.config` file.

To choose an IPP based on the Web request, add the parameter `ipProfile` to the **PUT** request or `IPProfile` to a **POST** request. Example:

```
http://localhost/mobilesdk/api/billpay?processImage=true&ipProfile=Bills
```

The parameter's value is the `profileName` property of the `ipProfile` element defined in the `Web.config` file.

To choose an IPP based on the properties defined in the "Kofax Mobile ID Capture" project element in the `Web.config` file, the following criteria must be met:

- The project mapping for "Kofax Mobile ID Capture" is selected.
- Image processing is requested.
- An xIDType of ID or 'Passport' is specified.
- Image metadata contains property items for manufacturer and model (if not, the image DPI is used).
- Additional properties are provided in the `Web.config` project element for "Kofax Mobile ID Capture".

Modifying Web.Config

The following changes can be made to the `Web.config` file to enable various features.

Returning HTTP 200 Responses

This feature is enabled by adding a property to the `projects` element in the `Web.config` file. This new property is `enhancedErrorReporting` and can have a value of On or Off, with Off the default.

```
<projects enhancedErrorReporting="On">
```

Multiple Image Processing Profiles

To choose an image processing profile based on a value provided in the Web request, add elements like these to the `Web.config` file in the `<processingSettings>` section:

```
<ipProfiles>
  <ipProfile profileName="MobileID"
             profileFile="C:\ip\mobileIdProfile.txt" />
  <ipProfile profileName="ScannerID"
             profileFile="C:\ip\scannerIdProfile.txt" />
  <ipProfile profileName="MobilePassport"
             profileFile="C:\ip\mobilePassportProfile.txt" />
  <ipProfile profileName="ScannerPassport"
             profileFile="C:\ip\scannerPassportProfile.txt" />
  <ipProfile profileName="Bills"
             profileFile="C:\ip\BillsProfile.txt" />
</ipProfiles>
```

To choose an image processing profile based on the project element for Kofax Mobile ID Capture in the `Web.config` file, use this project element as an example:

```
     <project
         projectMapping="MobileID"
         projectFile="C:\Program Files\Kofax\MobileIDCapture\KofaxIdentity.fpr"
         autoProfileDetect="ON"
         ipProfile="C:\ImageProcessingProfiles\defaultIPProfile.txt"
         ipMobileID="C:\ImageProcessingProfiles\mobileIdProfile.txt"
         ipScannerID="C:\ImageProcessingProfiles\scannerIdProfile.txt"
         ipMobilePassport="C:\ImageProcessingProfiles\mobilePassportProfile.txt"
         ipScannerPassport="C:\ImageProcessingProfiles\scannerPassportProfile.txt"
         ipOutputFormat="JPG"
         processPoolInitialSize="4"
         processPoolThreshold="1"
         separateExtractionProcessPool="true"/>
```

- `autoProfileDetect`=”On” turns the feature on; Off or not specified turns it off.
- i`pProfile` defines the default image processing profile (IPP); this is used if one of the others is not selected.
- `ipMobileID` defines the IPP to use when IDType is ID and the source is a mobile device.
- `ipScannerID` defines the IPP to use when IDType is ID and the source is a scanner.
- `ipMobilePassport` defines the IPP to use when IDType is Passport and the source is a mobile device.
- `ipScannerPassport` defines the IPP to use when IDType is Passport and the source is a scanner.
- `ipOutputFormat` selects the file format produced by image processing; any value other than JPG produces TIF output.
- `separateExtractionProcessPool` The Real-Time Transformation Interface creates a separate worker process pool for Kofax Transformation Modules extraction if present and set to true.

Setting Transaction Logging

To enable transaction logging to a database, add the `databaseLoggingSettings` element to `Web.config` as a child element of the configuration element:

```
<configuration>
<databaseLoggingSettings>
        <databaseLogging enabled="true"
        vendor="oracle"
        host="oracleservername"
        port="1521"
        SID="orcl"
        user="username"
        password="password"/ >
</databaseLoggingSettings>
..
```

The databaseLoggingSettings element is configured using the following child elements:

databaseLogging (required)

The databaseLogging element has the following attributes :

enabled: Specifies whether the feature is enabled; possible values are true or false.

vendor: Specifies the database type used for the logging database; possible values are oracle and sqlserver.

host: The logging database host name or ip address.

port: The network port used to connect to the logging database.

SID: The Oracle System ID for the logging database. This attribute is only required when the vendor attribute is "oracle" and will be ignored otherwise.

user: The username to connect to the logging database.

password: The password to connect to the logging database.

databaseName: The name of the logging database. This attribute is only required when the vendor attribute is sqlserver and will be ignored otherwise.

connectionString: Specifies a specific connection string to supersede the connection information in the above attributes. If this attribute has a non-empty value, all of the above attributes will be ignored and the connection string value will be used instead.

encryptSection: If the value of this attribute is set to “true” Real-Time Transformation Interface will encrypt the `databaseLoggingSettings` section of `Web.config` upon service start-up. Use this attribute if the above attributes contain sensitive information such as passwords that should not be visible in plain text.

Note Oracle and Microsoft SQL Server databases are supported. Oracle: After installation, connect to the logging database and execute the SQL script `oracleschema.sql`, which can be found in the installation directory under `bin\Schema.SQL Server`. After installation, connect to the logging database and execute the SQL script `sqlserverschema.sql`, which can be found in the installation directory under `bin\Schema`. This feature requires an Insight installation and an Insight license. If you are upgrading from 1.5.0.0.FIX4777.193 you need to update the Oracle job logging database. Connect to the job logging database and execute the SQL script oracle-schema-update-FIX4777.sql.

excludedFields (optional)

This element allows you to enumerate fields whose extracted text values should NOT be preserved in the logging database, perhaps because they are sensitive in nature. Text values will appear in the database as an empty string. For each field you wish to exclude, include an `excludedField` child element with the project mapping and name of the field. For example, the following will result in 3 fields being excluded: the `Name` and `AccountNumber` fields of the BillPay project, and the `CVV` field of the CreditCard project

```
<excludedFields>
           <excludedField projectMapping = "BillPay" fieldName = "Name"/>
           <excludedField projectMapping = "BillPay" fieldName = "AccountNumber"/>
           <excludedField projectMapping = "CreditCard" fieldName = "CVV"/>
</excludedFields>
```

excludedImageProjects (optional)

This element allows you to enumerate projects whose original and processed images should NOT be preserved in the logging database, perhaps because they may contain data that is sensitive in nature. For each project you wish to exclude, include an `excludedProject` child element with the project mapping name. For example, the following will result in 2 projects being excluded: MobileID and CreditCard.

```
<excludedImageProjects>
           <project projectMapping = "MobileID"/>
           <project projectMapping = "CreditCard"/>
```

```
 </excludedImageProjects>
```

cleanupSettings (optional)

This element allows you to configure periodic deletion of data from the logging database that matches specified criteria. The `timespan` attribute of the `cleanupSettings` element controls how often the cleanup process is run to delete matching data. For example, the following `timespan` value would result in the cleanup process being run every 4 hours

```
<cleanupSettings  timespan = "00.04:00:00" >
```

This `cleanupSettings` element has the following child elements:

deleteJobs (optional)

This element allows you to configure transactions to be deleted from the logging database based on their status and document type. You can specify multiple status / document type combinations using the `jobInfo` element. You can specify how old the transactions must be using the optional `olderThan` attribute.

For example,

```
<deleteJobs olderThan="7.00:00:00">
    <jobInfos>
    <jobInfo status="0" documentType="Check"/>
    <jobInfo status="0" documentType="Bill"/>
    </jobInfos>
</deleteJobs>
```

will delete all transactions with a status of '0' (Completed) and a document type of "Check" or "Bill" that are at least one week old.

Use the string 'null' to indicate transactions that have a NULL document type.

deleteJobsByStatus (optional)

This element allows you to configure transactions to be deleted from the logging database based on their status only. Use the `option` attribute to specify whether the indicated statuses are included or excluded from those to be deleted. You can specify how old the transactions must be using the optional `olderThan` attribute.

For example,

```
<deleteJobsByStatus option="include" olderThan="3.00:00:00">
    <statuses>
    <status value="0"/>
    <status value="1"/>
    </statuses>
</deleteJobsByStatus>
```

will delete all transactions with a status of '0' (Completed) or '1' (Error) that are at least 3 days old.

Alternatively,

```
<deleteJobsByStatus option="exclude" olderThan="3.00:00:00">
    <statuses>
    <status value="2"/>
    </statuses>
</deleteJobsByStatus>
```

will delete all transactions that DO NOT have a status of '2' (In Progress) that are at least 3 days old.

deleteJobsByDocumentType (optional)

This element allows you to configure transactions to be deleted from the logging database based on their document type only. Use the `option` attribute to specify whether the indicated document types are included or excluded from those to be deleted. You can specify how old the transactions must be using the optional `olderThan` attribute.

For example,

```
<deleteJobsByDocumentType option="include" olderThan="0.12:00:00">
    < documentTypes >
    < documentType value="Bill"/>
    < documentType value="Check"/>
    </documentTypes >
</ deleteJobsByDocumentType >
```

will delete all transactions with a document type of 'Bill' or 'Check' and that are at least 12 hours old.

Alternatively,

```
< deleteJobsByDocumentType option="exclude" olderThan="0.12:00:00">
    < documentTypes >
    < documentType value="ID"/>
    </documentTypes >
</ deleteJobsByDocumentType >
```

will delete all jobs **other than** those with a document type of ID and that are at least 12 hours old.

Use the string "null" to indicate transactions that have a NULL document type.

deleteImagesByDate (optional)

This element allows you to configure images to be deleted from the logging database based on either their age or a date range. There are two attributes: if the `deleteOriginal` attribute is set to true than original images from the client will be included in those deleted. If the `deleteServerProcessed` attribute is set to true than copies of images that are the output of server processing will be included in those deleted.

There are two child elements.

The `dateRange` element is used to indicate a range of time. Images associated with transactions whose request timestamp fall within the specified time will be deleted.

For example,

```
<deleteImagesByDate deleteOriginalImages="false"
 deleteServerProcessedImages="true">
    <dateRange begin="2016-07-01 00:00" end="2016-07-31 00:00" />
</deleteImagesByDate>
```

will delete all server-processed images associated with transactions during July 2016.

The `olderThan` element is used specify how old the transactions must be before the image will be deleted.

```
<deleteImagesByDate deleteOriginalImages="true"
 deleteServerProcessedImages="false">
    <olderThan value="7.00:00:00" />
</deleteImagesByDate>
```

will delete all original images from the client that are associated with transactions at least one week old.

Converting PDF Documents

The Real-Time Transformation Interface converts PDF documents to JPEG when image processing is requested. The PDF documents can be single or multi-page. To improve response time, the project property `ipProcessPDF` has been added to `Web.config` to indicate that PDF processing will be done. When `ipProcessPDF` is set to On, Real-Time Transformation Interface initializes the PDF conversion library during service startup.

This is an example of the `Web.config` project element for Kofax Mobile Bill Pay with the `ipProcessPDF` property set:

```
<project projectMapping="BillPay"
  projectFile="C:\Mobile Frameworks\KofaxMobileBillPay-1.3.0.0.0.78\KTM Project
\Kofax_Bill_Pay.fpr"
  ipProfile="C:\ipp\BillsProfile.txt"
  ipProcessPDF="On"
  processPoolInitialSize="4"
  processPoolThreshold="1" />
```

Return Processed Image

The `maxBufferSize` setting in the `<bindings>` section of the `Web.config` file may need to be updated.

1. Open the `Web.config` file and locate the `<bindings>` section.
2. In the `<binding name="SecurePipeBinding">` setting change the value of `maxBufferSize` to 2147483647.
3. Save and close the `Web.config` file.

Image Resolution Threshold

When using the `autoProfileDetect` feature, the Real-Time Transformation Interface may use an image's resolution to determine the source of the image, which can be a scanner or a mobile device. If an image's resolution is below the threshold (and no image metadata is found), the image source is set to mobile device. The default value of the image resolution threshold is 150, but it can be adjusted using the `scannedImageResolutionThreshold` project property.

This is an example of the `Web.config` project element for Kofax Mobile ID Capture with the `scannedImageResolutionThreshold` property set to 100:

```
<project projectMapping="MobileID"
  projectFile="C:\Program Files\Kofax\MobileIDCapture\KofaxIdentity.fpr"
  autoProfileDetect="On"
  ipOutputFormat="JPG"
  scannedImageResolutionThreshold="100"
  ipProfile="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
  ipMobileID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
  ipScannerID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID_Scanner.txt"
```

```
  ipMobilePassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport.txt"
  ipScannerPassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport_Scanner.txt"
  processPoolInitialSize="4"
  processPoolThreshold="1" />
```

Warmup Image

When using the `warmupImage` feature, a front and back image can be specified by inserting a question mark ('?') between two path names. Please note that it is still valid to provide a single file as the warmup image.

This is an example of the `Web.config` project element for Kofax Mobile ID Capture with the `warmupImage` property set to a front and back image:

```
<project projectMapping="MobileID"
 projectFile="C:\Program Files\Kofax\MobileIDCapture\KofaxIdentity.fpr"
 warmupImage="C:\Images\Mobile_Id_Warmup_Image_Front.jpg?C:\Images
\Mobile_Id_Warmup_Image_Back.jpg"
 autoProfileDetect="On"
 ipOutputFormat="JPG"
 ipProfile="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipMobileID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipScannerID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID_Scanner.txt"
 ipMobilePassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport.txt"
 ipScannerPassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport_Scanner.txt"
 processPoolInitialSize="4"
 processPoolThreshold="1" />
```

Warmup Image Parameters

When using the `warmupImage` feature, the Real-Time Transformation Interface uses default parameters to process that image. The `warmupImageParameters` project property can be set to specify parameters for the warmup image, allowing any processing option, id type, region, and resize value.

This is an example of the `Web.config` project element for Kofax Mobile ID Capture with the `warmupImageParameters` property set:

```
<project projectMapping="MobileID"
  projectFile="C:\Program Files\Kofax\MobileIDCapture\KofaxIdentity.fpr"
  autoProfileDetect="On"
  ipOutputFormat="JPG"
  warmupImage="C:\Images\IDFront.jpg"
  warmupImageParameters="processImage=true?xIDType=ID?xRegion=UNITED STATES?
xImageResize=ID-1"
  ipProfile="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
  ipMobileID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
  ipScannerID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID_Scanner.txt"
  ipMobilePassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport.txt"
  ipScannerPassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport_Scanner.txt"
  processPoolInitialSize="4"
```

```
  processPoolThreshold="1" />
```

Warmup Image and SeparateExtractionProcessPool

When using the `warmupImage` feature, if the `separateExtractionProcessPool` project property is enabled you must provide an additional project property specifying warmup images for the Kofax Transformation Modules extraction worker process pool.

This is done by using the `warmupImageExtraction` and (optionally) `warmupImageExtractionParameters` properties. These properties have the same effect as the `warmupImage` and `warmupImageParameters` properties but are applied only to worker processes in the Kofax Transformation Modules extraction pool.

Because the purpose of the `separateExtractionProcessPool` feature is to separate worker processes that do Kofax Transformation Modules extraction from worker processes that perform image processing, the `processImage` parameter is not supported for use with the `warmupImageExtractionParameters` property. If `processImage` parameter is set to true within `warmupImageExtractionParameters` it will be ignored. For this reason it is recommended to use processed images only for the `warmupImageExtraction` property.

```
<project
 projectMapping = "MobileID"
 projectFile = "C:\Program Files\Kofax\MobileIDCapture\KofaxIdentity.fpr"
 autoProfileDetect = "On"
 ipOutputFormat = "JPG"
 ipProfile = "C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipMobileID = "C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipScannerID = "C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID_Scanner.txt"
 ipMobilePassport = "C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport.txt"
 ipScannerPassport = "C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport_Scanner.txt"
 processPoolInitialSize = "4"
 processPoolThreshold = "1"
 separateExtractionProcessPool = "true"
 warmupImage = "C:\Images\IDFront.jpg"
 warmupImageParameters = "processImage=true?xIDType=ID?xRegion=UNITED STATES?
xImageResize=ID-1"
 warmupImageExtraction = "C:\Images\IDFront.jpg"
 warmupImageExtractionParameters = "xIDType=ID?xRegion=UNITED STATES?
xImageResize=ID-1"/>
```

Image Quality Check

The Real-Time Transformation Interface can perform image analysis if classification or extraction fails. The image analysis errors detected (in order) are:

- Poorly lit (undersaturated)
- Overly lit (oversaturated)
- Shadows
- Glare
- Blurred
- Missing required borders
- Overly skewed

- Uneven or same background

The project property used to enable image analysis is `imageQualityCheck`. Setting this to On causes the Real-Time Transformation Interface to perform image analysis if classification or extraction fails. The default value for this setting is Off.

This is an example of the `Web.config` project element for Kofax Mobile ID Capture that includes this setting:

```
<project projectMapping="MobileID"
 projectFile="C:\Program Files\Kofax\MobileIDCapture\KofaxIdentity.fpr"
 autoProfileDetect="On"
 ipProfile="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipMobileID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipScannerID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID_Scanner.txt"
 ipMobilePassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport.txt"
 ipScannerPassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport_Scanner.txt"
 ipOutputFormat="JPG"
 imageQualityCheck="On"
 processPoolInitialSize="4"
 processPoolThreshold="1" />
```

To take full advantage of this feature, server-side processing must always be used, even when submitting images from a native mobile application. Furthermore, image analysis is only supported with server-side processed images. Any results of image analysis on an image that has been previously processed are undefined.

If image quality analysis detects more than one of the error conditions, the error code corresponding to the first condition found shall be returned, but all error conditions shall be included in the error description. For example, if image quality analysis detects the Overly Lit and Glare error conditions, it will return the error code corresponding to Overly Lit, which is 305 or 505, but include Overly Lit and Glare in the error description.

This feature also supports new image processing profile keywords to adjust image analysis thresholds. These keywords are not required unless a threshold's value needs to be changed.

- `blurthreshold`
  - Type is integer.
  - Range is [0,100] with the default = 10.
  - Sensitivity increases as value increases.
  - This threshold is associated with error codes 306 and 506 (image may be blurred).
- `blurpercentthreshold`
  - Type is integer.
  - Range is [0,100] with the default = 70.
  - Sensitivity decreases as value increases.
  - This threshold is associated with error codes 306 and 506 (image may be blurred).
- `undersaturatedthreshold`
  - Type is integer.
  - Range is [0,256] with the default = 18.

  - Sensitivity decreases as value increases.
  - This threshold is associated with error codes 304 and 504 (image may be poorly lit).
- `undersaturatedpercentthreshold`
  - Type is integer.
  - Range is [0,100] with the default = 49.
  - Sensitivity decreases as value increases.
  - This threshold is associated with error codes 304 and 504 (image may be poorly lit).
- `oversaturatedthreshold`
  - Type is integer.
  - Range is [0,100] with the default = 10.
  - Sensitivity decreases as value increases
  - This threshold is associated with error codes 305 and 505 (image may be overly lit).
- `oversaturatedpercentthreshold`
  - Type is integer.
  - Range is [0,100] with the default = 33.
  - Sensitivity decreases as value increases.
  - This threshold is associated with error codes 305 and 505 (image may be overly lit).

This is an example of how to set thresholds in an image processing profile file:

```
{
 "imageperfectionsettings": <existing image perfection string>,
 "blurthreshold": 10,
 "blurpercentthreshold": 70,
 "undersaturatedthreshold": 18,
 "undersaturatedpercentthreshold": 49,
 "oversaturatedthreshold": 10,
 "oversaturatedpercentthreshold": 33
}
```

Image Quality Check Output

When the Image Quality Check feature is enabled, the Real-Time Transformation Interface can output the images processed during image analysis. The project property used to enable image analysis output is `imageQualityCheckOutput`. Setting this to On causes the Real-Time Transformation Interface to create the folder `ImageQualityCheck` in the `DataLogging` output directory for the request. The default value for this setting is Off.

The files written to this folder are created during image quality analysis. Output images are created from the image analysis checks for the conditions below, with an example file name.

- **Glare**: `glare-image.jpg`
- **Shadows**: `shadows-image.jpg`
- **Uneven or same background**: `background-image.jpg`
- **Undersaturation (poor lighting)**: `poorlight-image.jpg`
- **Skew, blur, and oversaturation (too much light)**: `quickanalysis-image.jpg`

This is an example of the `Web.config` project element for Kofax Mobile ID Capture that includes this setting:

```
<project projectMapping="MobileID"
 projectFile="C:\Program Files\Kofax\MobileIDCapture\KofaxIdentity.fpr"
 autoProfileDetect="On"
 ipOutputFormat="JPG"
 imageQualityCheck="On"
 imageQualityCheckOutput="On"
 ipProfile="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipMobileID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipScannerID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID_Scanner.txt"
 ipMobilePassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport.txt"
 ipScannerPassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport_Scanner.txt"
 processPoolInitialSize="4"
 processPoolThreshold="1" />
```

Diagnostic Data

When the Diagnostic Data feature is enabled, the Real-Time Transformation Interface returns diagnostic information with each request. The project property used to enable returning diagnostic data is `diagnosticData`. Setting this to On causes Real-Time Transformation Interface to populate the diagnostic data field of the request response with image and processing information. The default value for this setting is Off.

If `diagnosticData` is On, the response element `diagnosticResult` is populated with the items below. If diagnosticData is Off, the element is null.

- `imageData` - items from each image's Exif data and image processing metadata (if image processing requested) are encapsulated under this element.
- `deviceManufacturerModel` - from the image's Exif data, the manufacturer and model of the device that captured the image, or null if not found or output image only.
- `imageProcessingMetadata` - the metadata containing the image processing results, or null if input image only.
- `imageResolution` - the horizontal resolution in pixels per inch of the image, or zero if output image only.
- `imageSize` - the width and height in pixels of the image, or null if output image only.
- `imageSource` - either `MobileDevice` or Scanner, determined by `deviceManufacturerModel` and `imageResolution`, or null if output image only.

Note If the output image count equals the input image count (or if there are no output images), the element count equals the image count. If the output image count is greater than the input image count, the element count is input image count plus output image count. In this case, the elements for the output images only contain image processing metadata.

- `imageProcessingProfile` - the file name only of the image processing profile used, or null if image processing not requested.
- `imageProcessingProfileAlternate` - the file name only of the alternate image processing profile used, or null if not specified or image processing not requested.

- `imageProcessingTime` - the elapsed time (in seconds) taken by the Real-Time Transformation Interface to prepare images for image processing and the image processing itself, or null if image processing not requested.
- `requestResponseTime` - the elapsed time (in seconds) taken by the Real-Time Transformation Interface to process the request; this time does not include the time to transport the request from the client to the server.

Additionally, if the project parameter `autoProfileDetect` is On, the artifacts used to determine the profile to use are encapsulated under the element `autoProfileDetectResult`. If `autoProfileDetect` is Off, this element is null.

- `idType` - the xValue provided in the URL, or null if not specified.
- `deviceManufacturerModel` - from the image's Exif data, the manufacturer and model of the device that captured the image, or null if not found or `idType` is null.
- `imageSource` - either `MobileDevice` or Scanner, determined by `deviceManufacturerModel` and `imageResolution`, or null if `idType` is null.
- `imageProcessingProfile` - the image processing profile, determined by `idType` and `imageSource`, or null if `idType` is null or not id or passport
- `imageResolution` - the horizontal resolution in pixels per inch of the image or zero if

  ```
  idType
  ```

  is null.

This is an example of the `Web.config` project element for Kofax Mobile ID Capture that includes the `diagnosticData` project property:

```
<project projectMapping="MobileID"
 projectFile="C:\Program Files\Kofax\MobileIDCapture\KofaxIdentity.fpr"
 autoProfileDetect="On"
 diagnosticData="On"
 ipOutputFormat="JPG"
 ipProfile="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipMobileID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID.txt"
 ipScannerID="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_ID_Scanner.txt"
 ipMobilePassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport.txt"
 ipScannerPassport="C:\Program Files\Kofax\MobileIDCapture
\ImagePerfectionProfile_MobileID_Passport_Scanner.txt"
 processPoolInitialSize="4"
 processPoolThreshold="1" />
```

MRZ-based Cropping for Passports

This feature provides support for the Real-Time Transformation Interface to specify usage of MRZ-based cropping functionality, which allows more precise passport cropping based on MRZ detection. MRZ detection is enabled if the property `usemrzpassportdetection` is set to on in the image processing profile file. The default setting of this property is off.

If successful, MRZ detection returns the four corners of the passport, which are formatted into an addendum to the image perfection operation string. Image processing then continues as normal.

If not successful, the MRZ detection's return structure is examined to determine the failure condition. An error string is constructed and returned, which gets written to the debug log and added to the diagnostic data object if that feature is active.

An example of how to set this property in an image processing profile file:

```
{
 "imageperfectionsettings": <existing image perfection string>,
 "usemrzpassportdetection": "on"
}
```

Raising the Batch_Close KTM script event

When this feature is enabled for a project, the Real-Time Transformation Interface will cause the `Batch_Close` script event for the Kofax Transformation Modules project class to be executed. The script event will be executed once per request after all documents have been extracted. This feature is enabled by default. To disable it set the `batchCloseEvent` property of the project to "off".

```
<project projectMapping="BillPay"
projectFile="C:\Mobile Frameworks\KofaxMobileBillPay-1.3.0.0.0.78\KTM Project
\Kofax_Bill_Pay.fpr"
ipProfile="C:\ipp\BillsProfile.txt"
ipProcessPDF="On"
processPoolInitialSize="4"
processPoolThreshold="1"
batchCloseEvent="off"
 />
```

When invoked by the Real-Time Transformation Interface, the `Batch_Close` event supports reading and writing field values only. Other operations are not supported.

Creating and checking debug logs

Logging is managed through .NET TraceSource, which allows runtime modification of the logging level.

To create a debug log, uncomment the DebugListener section in the web.config file.

To check for errors, open the Windows Application Event Log with the source listed as "Real-Time Transformation Interface." This is where errors are written by default.

Note Real-Time Transformation Interface can also log transactions to a database. Enable this by adding the databaseLogging element to the processingSettings section of the `Web.config` file.

Enhancements to error type reporting

This feature provides support for the Real-Time Transformation Interface to return an error code and more detailed error descriptions in the web request result structure when the enhanced error reporting feature is enabled. The error codes and descriptions are documented in Error types and descriptions.

Four image processing profile features have been added to validate image height, width, and DPI. For height and width validation, both dimensions of the image are checked the same way because the image orientation is not known. Height and width validation is configurable via the following properties in the image processing profile files.

Property	Allowed Range	Default Value
MinSidePixelCount	[72, 9600]	72
MaxSidePixelCount	[72, 9600] or 0 (no limit)	0

Please note that the fact that an image passes this height and width validation does not guarantee it will be free of dimension related issues during subsequent processing.

DPI validation is configurable via the following properties in the image processing profile files.

Property	Allowed Range	Default Value
MinScannedDPI	[72, 9600]	72
MMaxScannedDPI	[72, 9600] or 0 (no limit)	0

Please note that the fact that an image passes this DPI validation does not guarantee it will be free of resolution related issues during subsequent processing.

For these properties, max-type values must be greater than min-type values.

An example of how to set all four new properties in an image processing profile file:

```
{
 "imageperfectionsettings": <existing image perfection string>,
 "minsidepixelcount": 100,
 "maxsidepixelcount": 5000,
 "minscanneddpi": 200,
 "maxscanneddpi": 600
}
```

Activating the Real-Time Transformation Interface license

When you purchase the Real-Time Transformation Interface, your Kofax Capture license is updated to include a system license for the Real-Time Transformation Interface. If you activate the license before installing the Real-Time Transformation Interface, the license appears in the utility as "License 13002." After installing and activating the product, the license appears in the utility as "Real-Time Transformation Interface."

Use the Kofax Capture License Utility to activate the Real-Time Transformation Interface license.

Note To check the licensing status of Kofax Transformation Modules components, open a project in Project Builder and select **File** > **License Utility**.

If you use a Kofax Capture hardware license key, attach the key to the Web server before performing the following procedure.

1. Open the **Kofax Capture License Utility**.

2. Select **File** > **Activate**.
   The Kofax Capture License Utility checks the licensing status and activates the license.

The system decrements the Kofax Transformation Modules license based on the project configuration. See the Kofax Transformation Modules documentation for more information.

Chapter 4

Real-Time Transformation Interface AppStats

If an application also incorporates the Real-Time Transformation Interface server, then the device-side statistics that are collected can be uploaded and combined with the server side metrics, to provide a complete, end-to-end traceability of performance and usage metrics.

Public API

The public API offers the following features.

Invocation URL - AppStats reporting

The App Stats reporting service can be accessed at the following URL.

```
http://localhost/mobilesdk/api/AppStats
```

When submitted using the `GET` verb, a flag is returned indicating if AppStats reporting is enabled on the server.

When submitted using the `PUT` verb, the exported app stats file is part of the request. The format (content type) of this file is `application/json`.

In each of these requests there can be an optional session key in the URL query string that is created on the device. This key is part of any event logged for Real-Time Transformation Interface AppStats. If the session key is not included in the URL, the Real-Time Transformation Interface server will create a GUID as the session key for the Web service call.

Request headers

The caller may specify the format in which the response is sent by utilizing the `Accept` header. Currently supported formats are either `application/json` for JSON formatted output or `application/xml` for XML output. In absence of a request header, the service will default to `application/json`.

Response

The response will vary depending on the `Accept` header. For both formats, the data contains three fields: `appStatsFolder` and `appStatsPlugin` (Boolean), and `sessionKey` (string).The two Boolean flags indicate the following:

- `appStatsFolder` - true = a folder has been configured to receive AppStats data files.
- `appStatsPlugin` - true = a plug-in has been correctly configured to process AppStats data.

The session key returned is either the one parsed from the URL query string or a new one generated on the server if one was not provided by the Web service call; if generated by the Real-Time Transformation Interface server, the session key is a GUID. The following is a sample JSON response:

```
{
   "sessionKey": "ad8e3864-c646-4c5b-b272-2f3f3c6bece3",
   "appStatsFolder": true,
   "appStatsPlugin": true
}
```

Errors

If there is an error, response code 500 is returned along with the error text and a GUID identifier.

Plug-in interface

The application developer can write a plug-in that will write data sent to the reporting service to a data repository of their choosing. The interface members that must be implemented by the plug-in for this feature are defined by `IAppStatsRecorder`:

- `Initialize` - perform one-time operations.
- `Open` - perform open-specific operations.
- `WriteDeviceAppStats` - deliver the device-provided AppStats file to a user-defined destination.
- `Close`- perform close-specific operations
- `RTTIEvent` – an RTTI event passes an event type and an object to encapsulate the event data to be added to the data repository.
- `RTTIEventUpdate` – an RTTI update event passes an event type and an object to encapsulate the event data to be updated in the data repository.

One of the parameters of the `WriteDeviceAppStats` method is an object of type `KofaxJsonObject`. This type defines the JSON data passed by the device to RTTI; this data represents the AppStats database schema. The full schema can be found in the *Kofax Mobile SDK Developer's Guide* by searching for "SQL Database Schema".

Sample plug-in

A sample plug-in is provided with the SDK that writes the AppStats data to an MS SQL Server or Oracle database. This plugin:

- Is written in C#.
- Was developed in Visual Studio 2013.
- Is delivered as both code and a DLL.
- Parses JSON data into data structures resembling tables in the database.
- Inserts the data into a database using ado.net connection objects.
- By default, can be found at `C:\Program Files\Kofax\RTTI\Plugin`.

The classes that implement the sample plugin are:

- `RTTIJsonParser` – implements the `IAppStatsRecorder` interface.

- `DatabaseConnect` – reads the configuration file containing database settings, interfaces with the database object (open, close, insert, update), parses the JSON data from the device, and builds insert and update strings.
- `DatabaseConnection` – creates, opens, and closes database connections and executes query and non-query database statements.
- `DatabaseUtils` – contains string and integer constants and utility properties and methods.

Note When using your own custom AppStats plugin, Real-Time Transformation Interface performance will depend on how the plugin has been implemented. In some cases performance may be negatively impacted. If your plugin performs time intensive operations, consider delegating them to a background thread.

Application statistics

When the focus is on collecting metrics related to Real-Time Transformation Interface (Real-Time Transformation Interface) requests and responses (Transformation and Validation), the `logSessionEvent` method allows the application to define which sequence of local device operations and Real-Time Transformation Interface requests belong to the same session.

The `logSessionEvent` method receives a parameter from an associated AppStats data object `AppStatsSessionEvent` which encapsulates details of each logged event.

Session Key

When any Real-Time Transformation Interface Web service is invoked (except for GET), a Session Key is part of the URL query string; this Session Key is passed to all subsequent methods for Real-Time Transformation Interface event reporting, and is returned to the caller as part of the Web service response. If the Session Key is not present, one is created by the API controller and is used the same way as one provided by the Web service call.

Document ID

When the Real-Time Transformation Interface Transformation or Validation Web service is invoked, a document ID is part of the URL query string; this document ID is passed to all subsequent methods for Real-Time Transformation Interface event reporting, and is returned to the caller as part of the Web service response. If the document ID is not present, one is generated for each document created during the extraction or validation process.

Image ID

When the Real-Time Transformation Interface Transformation Web service is invoked, an image ID can be part of the URL query string; this image ID is passed to all subsequent methods for Real-Time Transformation Interface event reporting (where required). If the image ID is not present, one is created for each image passed in; if image processing is specified, an image ID is also created for each image processed.

Real-Time Transformation Interface Events

The `RTTIEventType` parameter passed into the methods `RTTIEvent` and `RTTIEventUpdate` describes the type of event and classifies the event data passed with it. The Real-Time Transformation Interface event types are:

- Document Create
- Image Create
- ImageProcessor Start / End
- Image Processor Event
- Request Start / End
- Separation Start / End
- OCR Start / End
- Extraction Start / End
- Validation Start / End
- Classification Start / End
- Classification Event
- Classification Event Alternative
- Field Change Event
- Document Start / End
- Environment
- Instance
- Session Start / End
- Session Event

Extraction Accuracy

Additional field properties related to extraction accuracy will be returned by the transformation Web service. These properties, along with the currently-returned field properties, will be logged as a Real-Time Transformation Interface Field Change event.

The added properties are:

- The extraction confidence.
- Whether or not a formatter failed.

Chapter 5

Real-Time Transformation Interface parameters

Field alternatives

Field alternatives are a set of confidence-ordered results for a specific field. These are referred to as "alternatives" even if there is only one result. The first listed alternative has the highest confidence and in most cases is copied into the main field value.

Lower-confidence alternatives do not get considered for the field's root value, but are used in the evaluation process. For example, a project threshold can be set to invalidate the field if the first two alternatives are too close in confidence, to make it clear that an incorrect alternative may have been chosen.

Sometimes field alternatives can be useful for custom evaluation purposes if a lower-confidence result is preferred based on its origin or other properties. For example, the Billers field within Kofax Mobile Bill Pay exposes the alternative addresses extracted with different techniques within the Transformation project, and developers of the Kofax Mobile SDK can use this data to decide what is presented to the device user within their custom application.

Subfields

Subfields are a related set of parallel fields for the field alternative. There are always the same number of subfields for each alternative, and they always have the same names and are arranged in the same order.

If we draw an analogy to the results of a database query, field alternatives are similar to the records / rows returned, and subfields are similar to the columns within each record.

In the following example URL, `projectMapping` is the project element defined in the `web.config` file.

```
http://localhost/mobilesdk/api/{projectMapping}?fieldAlternatives=true
```

For `POST` calls, put `FieldAlternatives True` in the form data.

Process image

The parameter string on the Real-Time Transformation Interface Web service calls supports a flag to indicate that image processing is to be performed on the image prior to sending it to the associated Kofax

Transformation Modules project. This optional parameter is `processImage`, which can be set to `true` or `false`. For example: `processImage=true`.

A value of `true` tells Real-Time Transformation Interface to process the image using the profile specified in `web.config`. If this parameter is omitted, no image processing is done.

For more details on the image processing profile see Sample Real-Time Transformation Interface image processing profile.

In the following example URL, `projectMapping` is the project element defined in the `web.config` file.

```
http://localhost/mobilesdk/api/{projectMapping}?processImage=true
```

For `POST` calls, put `ProcessImage True` in the form data.

Alternate image processing profile

The parameter string on the Real-Time Transformation Interface Web service calls supports flags to indicate that an alternate image processing profile is to be used for all images in a request after the first one.

The optional parameter `ipprofilealt` can be set to the name of the alternate profile to be used. For example: `ipprofilealt=IPProfile1`. The profile named in this parameter must be defined in `web.config`. If `ipprofilealt` is omitted, and `processImage=true` is specified, then all images in the request are processed with the same profile.

For more details on the image processing profile see Sample Real-Time Transformation Interface image processing profile.

In the following example URL, `projectMapping` is the project element defined in the `web.config` file.

```
http://localhost/mobilesdk/api/{projectMapping}?
processImage=true&ipprofilealt=IPProfile1
```

For POST calls, put `IPProfileAlt IPProfile1` in the form data.

Process first image only

The parameter string on the Real-Time Transformation Interface Web service calls supports a flag to indicate that the primary image processing profile is to be applied only to the first image of a request. This optional parameter is `processfirstimage`, which can be set to `true` or `false`. For example: `processfirstimage=true`.

A value of `true` tells the Real-Time Transformation Interface to process the first image of the request using the profile specified in `web.config`, and skip image processing for images other than the first image of the request. If this parameter is omitted, and `processimage=true` is specified, then all images of the request are processed.

Note Image processing must be specified for this parameter to have an effect. If image processing is suppressed by this parameter, an output image is still produced to satisfy output format uniformity.

For more details on the image processing profile see Sample Real-Time Transformation Interface image processing profile.

In the following example URL, `projectMapping` is the project element defined in the `web.config` file.

```
http://localhost/mobilesdk/api/{projectMapping}?
processImage=true&processfirstimage=true
```

For POST calls, put `ProcessFirstImage True` in the form data.

Return processed image

Real-Time Transformation Interface returns processed images when "image processing" and "image result "are requested. In a POST request, the parameter is `ImageResult`. When the `ProcessImage` and `ImageResult` parameters are set to `true`, the processed images will be returned as Base64, along with the image type, in the response field: `ProcessedImages`.

To use this feature in a **PUT** request, use a Web request like this:

```
http://localhost/mobilesdk/api/CheckDeposit?processImage=true&imageResult=true
```

Processing PDF documents

The Real-Time Transformation Interface will accept and upload PDF files via a Web service request. However, the receivingKofax Transformation Modules project must be configured to accept such files, otherwise an error may be generated.

The Web service that allows PDF files to be submitted to Real-Time Transformation Interface can be accessed at the following URL, where *projectMapping* is the project element defined in the `Web.config` file: `http://localhost/mobilesdk/api/{projectMapping}`.

- PUT and POST verbs are supported.
- The media type of the PUT request must be *application/pdf*.

Other parameters and contents of the URL query string are as defined by the transformation Web service.

Process count

Real-Time Transformation Interface supports a setting to indicate the number of images that should be sent to the associated Kofax Transformation Modules project. This optional parameter is `ProcessCount`, which can be set to any value up to the number of images in the request. If the count is higher than the number of images in the request, the value is non-numeric. If the value is less than one, all images are provided to the associated Kofax Transformation Modules project for processing. For example, a value

of 2 tells the Real-Time Transformation Interface to send only the first two images in the request to the associated Kofax Transformation Modules project for processing, but save all provided images to the configured output location. This setting is useful for debugging purposes, and should not be used for normal production activities.

This setting is only supported by the POST verb. For POST calls, put `ProcessCount 2` in the form data.

OCR data

As a developer, you can access raw OCR data generated by Kofax Transformation Modules when performing extraction via the Real-Time Transformation Interface. The Real-Time Transformation Interface extraction Web service has an optional parameter which enables returning OCR results. When enabled the extraction includes OCR data, including the values and locations of individual words extracted by the OCR process. For full documentation of the public API see the Web application help.

The Web service that returns OCR data as part of the response can be accessed at the following URL, where `projectMapping` is the project element defined in the `web.config` file.

```
http://localhost/mobilesdk/api/{projectMapping}?ocrResults=true
```

For `POST` calls, put `OCRResults True` in the form data.

Response

The response format will vary depending on the "Accept" header. For both formats, the data contains an additional document property called `words`. This property is an array of words from the default representation. Each element of the `words` array is a `CscXDocWord` object; properties returned are:

- Height
- Left
- PageIndex
- Text
- Top
- Width

The following is a sample JSON response:


```
{
   "extractionClass": "Bill",
   "classificationResult": [],
   "fields": [
      {
         "name": "Name",
         "text": "AMERICAN EXPRESS",
         "valid": true,
         "errorDescription": "",
         "left": 1189,
         "top": 844,
         "height": 39,
         "width": 448,
         "pageIndex": 0,
         "confidence": 0.28975698,
```

```
            "formattingFailed": false,
            "fieldAlternatives": null
         }
      ],
      "sessionKey": "1a63868d-af89-4d2e-9424-43d11a11a123",
      "documentId": "f2f9a064-ed55-41a6-b86d-0feff7e8f99f",
      "words": [
         {
            "text": "AMERICAN",
            "left": 1189,
            "top": 844,
            "height": 38,
            "width": 239,
            "pageIndex": 0
         },
         {
            "text": "EXPRESS",
            "left": 1446,
            "top": 844,
            "height": 39,
            "width": 191,
            "pageIndex": 0
         }
      ]
}
```


Output format

Real-Time Transformation Interface supports a value to indicate the format of an image processed by RTTI image processing. This optional parameter is `outputFormat` , which can be set to TIF, JPG, or PNG .

For example: `outputFormat=JPG`.

If this parameter is omitted, the output format defaults to TIF.

In the following example URL, `projectMapping` is the project element defined in the

web.config file.

```
http://localhost/mobilesdk/api/{projectMapping}?
processImage=true&outputFormat=JPG
```

For POST calls, put `OutputFormat JPG` in the form data.

Project parameters

Real-Time Transformation Interface supports an interface to provide parameters to the configured Kofax Transformation Modules project. These parameters are referred to as 'x-values' as they are preceded by an 'x' to distinguish them from parameters passed to the Real-Time Transformation Interface.

For example: `xRegion=Asia`.

In the following example URL, `projectMapping` is the project element defined in the `Web.config` file.

```
http://localhost/mobilesdk/api/{projectMapping}?xRegion=Asia&xIDType=ID
```

For POST calls, put `xRegion Asia` in the form data.

Chapter 6

Sample Real-Time Transformation Interface image processing profile

The `ipProfile` property that you set in the `web.config` file specifies the location of the image processing profile to use for images submitted to the associated Kofax Transformation Modules project. The image is processed as a TIFF image, which is sent to Kofax Transformation Modules. Both the original and processed images are stored in the `DataLogging` folder specified in the `web.config` file.

The following sample shows a Real-Time Transformation Interfacee image processing profile that illustrates the JSON layout:

```
{
   "basicsettings": {
      "outputbitdepth": "bitonal",
      "outputdpi": 0,
      "crop": "edge",
      "deskew": "edge",
      "rotate": "auto",
      "inputdoclongedge": 3.375,
      "inputdocshortedge": 2.125
    },
   "imageperfectionsettings":
   "_DeviceType_2_DoCropCorrection__DoSkewCorrectionPage__DoEnhancedBinarization_",
   "jpegquality": 90
}
```

Item	Type	Value	Description
basicsettings	object	See individual constituent objects	Specifies basic image processing settings. • If you provide both a basicsettings object and an `imageperfectionsettings` string, the basicsettings object is not used. • If you provide neither a basicsettings object nor an `imageperfectionsettings` string, an error is returned.
`outputbitdepth`	string	"bitonal" "grayscale" "color" ("Color")	Specifies the bit depth of the output image. If the `outputDPI` is greater-than or equal to 300, using "bitonal" also activates enhanced binarization. Using "bitonal" with an `outputDPI` that is Null or less than 300 activates non-enhanced binarization.

Item	Type	Value	Description
`outputdpi`	integer	0 - Max Int (0)	Specifies the required resolution of the output image in pixels per inch. The image processor scales as necessary. Using 0 specifies that the image processor automatically sets the output resolution and performs no scaling.
`crop`	string	"none" "edge" ("none")	• "none"—prevents image cropping. • "edge"—crops the image to detected page edges.
`deskew`	string	"none" "edge" ("none")	• "none"—prevents deskewing. • "edge"—deskews based on the detected page edges.
`rotate`	string	"none" "90" "180" "270" "auto" ("none")	Controls clockwise image rotation in increments of 90 degrees. If you enable "crop" and "deskew," the image processor rotates the image after cropping and deskewing is completed. If you select "auto," the image processor rotates the image so that it is right-side up based on the image content. (This option is normally used with "crop" and "deskew" enabled.)
`inputdoclongedge`	float	0.0 and up (0.0)	Specifies the long edge of the original physical document in inches; 0 means unspecified. You can use this parameter independently of `inputDocShortEdge`.
`inputdocshortedge`	float	0.0 and up (0.0)	Specifies the short edge of original document in inches; 0 means unspecified. You can use this parameter independently of `inputDocLongEdge`.
`imageperfectionsettings`	string	Empty string	Specifies advanced image processing settings. To use a basicsettings object, make `Imageperfectionsettings` an empty string or omit it altogether. See also the description for `basicsettings` for interaction notes.
`jpegquality`	integer	1-100 (90)	Specifies the quality setting used for JPEG compression in the output gray/color image. This applies to either `basicsettings` or `imageperfectionsettings`.

The following sample shows a Real-Time Transformation Interface image processing profile that illustrates some additional items the JSON layout:

```
{
  "imageperfectionsettings": "_DeviceType_2_DoSkewCorrectionPage__DoCropCorrection_

 _Do90DegreeRotation_200__DoFindTextHP__ProcessCheckFront__DoBinarization__LoadInlineSetting
```

```
_[CSkewDetect.convert_to_gray.Bool=1]_LoadInlineSetting_[CSkewDetect.scale_image_down.Bool=1]
  _LoadInlineSetting_[CSkewDetect.scale_down_factor.Int=80]_LoadInlineSetting
  _[CSkewDetect.correct_illumination.Bool=0]",
 "FrontLengthAssistsAllPageLengths": "true",
 "TokenReplaceList": [
   {
     "SearchFor": "_ProcessCheckFront_",
     "ReplaceWith": "_ProcessCheckBack_"
   }
 ]
}
```

Item	Type	Value	Description
FrontLengthAssistsAllPageLengths	Boolean	"true" "false" ("false")	Forces all output images processed on the same request to match in terms of the long edge pixel count.
TokenReplaceList	Object	List of text strings to search and replace.	Specifies text changes to be made in the imageperfectionsettings string used to process images other than the first image in the request. For example, the TokenReplaceList above has the effect of replacing the _ProcessCheckFront_ keyword with _ProcessCheckBack_ in the imageperfectionsettings used for processing the second and any subsequent images posted in the request.

Chapter 7

Real-Time Transformation Interface on-device extraction licensing

The Mobile SDK uses this Web service for on-device extraction licensing. The Mobile SDK periodically updates the RTTI server with the number of extracted documents in order to consume license volume appropriately. The Web service decrements the KTM Unlimited Fields Extraction volume license for every document that is extracted on-device.

Public API

The public API offers the following features.

Invocation URL - On Device Licensing

The Licensing service can be accessed at the following URL.

`http://localhost/mobilesdk/api/License`

Request Headers

Optionally, the caller may specify the format in which the response is sent by utilizing the `Accept` header. Currently supported formats are either `application/json` for JSON formatted output or `application/xml` for XML output. In absence of a request header, the service will default to `application/json`.

Request Body

The body is a json object containing a list of license IDs and corresponding counts of volume to consume. For example, to consume 50 units of license id 100, the app would specify Id as 100 and Count as 20, as well as the required Key1 and Key2 parameters.

```
{
"Key1":"81A189BE-2040-426A-82DF-2EF46C3F015B",
"Key2":"e1e50bbad9670bfde25e7d86948f3fce09afc5f0",
"Licenses": [{"Id":"100","Count":"20"}]
}
```

Additional pairs of license IDs and counts can be provided in the Licenses array. For example, to consume 50 units of license id 100, and 48 units of license id 200 the caller would specify something such as the following:

```
{
"Key1":"81A189BE-2040-426A-82DF-2EF46C3F015B",
"Key2":"e1e50bbad9670bfde25e7d86948f3fce09afc5f0",
```

```
"Licenses": [{"Id":"100","Count":"50"}, {"Id":"200","Count":"48"}]
}
```

The licensing service is accessed via the HTTP PUT verb. All other verbs are not supported.

The Key1 and Key2 parameters are required by the protocol for verifying license service client and server integrity. The Mobile SDK adds the Key1 and Key2 parameters to the request.

Response

The response will vary depending on the Accept header. For both formats, the data contains the following fields:

- licenseResultArray - this is an array of items. Each item in the array contains a license ID and the number of available licensing units associated with that ID. The number of available units is compared with the number requested. If the request exceeds the number of available units, there is not enough remaining license volume to process the request.

  If the corresponding license ID does not exist, a value of -1 is returned to the request.
- licenseToken - used by the Mobile SDK to validate the response integrity.

The following is a sample JSON response:

```
{
 "licenseResultArray": [
  {
    "licenseID": 100,
    "unitsRequested": 50,
    "unitsConsumed": 50
  }
 ],
 "licenseToken" : "03a9c25307dc9ddd2f87755d78ed33e18d276a7f"
}
```

Errors

If there is an error, HTTP response code 500 is returned.

Chapter 8

Error types and descriptions

The enhanced error reporting feature directs the Real-Time Transformation Interface to return HTTP 200 responses for most requests. When an error or exception is detected, an error type and description are returned. The error type can have one of the values listed below. The error descriptions associated with each type are also listed, along with the items (in braces) that are inserted into the message at run time.

The following table summarizes the error codes that can be returned by Real-Time Transformation Interface:

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
0	Success	None		
1	Exception	System	An error occurred processing job 5edbb411-86a8-42a5-b63c-59fd1f91beb0. Worker process has exited.	An uncategorized operating system or .NET error was detected by RTTI. This is an unknown error that is returned by the Operating System or .NET runtime.
101	Exception	Parameter	An error occurred processing job 061094ad-235b-43e0-a999-be1ba552bb00. Specified projectMapping MobileIDTest not found. Parameter name: ProjectMappingNotFound	The project mapping provided in the PUT or POST request is not defined in the RTTI Web.config file.
102	Exception	Parameter	An error occurred processing job 2f075214-fc01-4970-8f07-9bf26156c502. Project mapping not found in request data. Parameter name: ProjectMappingNotFoundPost	No project mapping value was found in the provided POST request.
103	Error	Parameter	Json data not found in PUT request.	An exported app stats file was not provided in the PUT request to the App Stats reporting service.
104	Exception	Parameter	Invalid validate parameter.	In the Validation Web service request the required parameter and value validate=true is not specified.

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
201	Error	Image	Invalid image - unknown error.	This error type is a "backstop" and will be thrown only for scenarios that are unknown to us now during processing of the image prior to image processing.
202	Error	Image	Invalid image - file name is blank.	An RTTI internal error due to a catastrophic system-level failure. This error can only occur in a significant system event such as a hardware failure (e.g. disk drive).
203	Error	Image	Invalid image - file <IMAGE FILE NAME> does not exist.	An RTTI internal error due to a catastrophic system-level failure. This error can only occur in a significant system event such as a hardware failure (e.g. disk drive).
204	Error	Image	Invalid image - file <IMAGE FILE NAME> is empty.	An image provided in the PUT or POST request is empty. The likely cause is that an empty image file is passed to the system.
205	Error	Image	PDF conversion failed: Password Protected, error code 5. PDF conversion failed: Failed to load document: -2, error code 4.	A PDF file provided in the PUT or POST request cannot be read or processed by the PDF conversion software. A likely cause is that the PDF file is password protected.
206	Error	Image	Invalid image content found: expected = JPG, found = PDF. Invalid image content found: expected = PNG, found = GIF. Invalid image content found: expected = PNG, found = unsupported file type. Invalid image content found: expected = supported file type, found = GIF.	RTTI has detected an image provided in the PUT or POST request is an invalid type or does not match the content type specified in the Web service request. The likely cause is that the request specifies that the file is of one type (e.g. JPG) but the actual content is of a different type (e.g. GIF).

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
207	Error or Exception	Image	Image processing failed - image processor memory could not be allocated. [-1] Image processing failed - image processor could not open file. [-2] image processing failed - image processor detected invalid DPI. [-3] Image processing failed - image processor detected invalid height or width. [-4] Image processing failed - image processor detected in invalid line width. [-5] Image processing failed - image processor detected an illegal channel depth specified. [-6] Image processing failed - image processor could not save the metadata in the limited output buffer. [-7] Image processing failed - image processor detected file read error. [-8] Image processing failed - image processor detected an illegal parameter. [-9] Image processing failed - image processor could not process the image. [-10] Image processing failed - image processor could not write the image. [-11] Image processing failed - image processor detected an unknown file type. [-12] Image processing failed - image processor detected a nonexistent image. [-13] Image processing failed - image processor detected an illegal internal file format. [-14] Image processing failed - image processor could not append to the file specified. [-15] Image processing failed - image processor cannot append to the file type specified. [-16] Image processing failed - image processor detected a bad PDF file. [-17]	RTTI has detected an error during image processing of one of the images provided in the PUT or POST request.

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
207 (cont)	Error or Exception	Image	Image processing failed - image processor detected an encryption appending issue. [-18] Image processing failed - image processor detected the PDF file is too large to append to. [-19] Image processing failed - image processor detected a speed accuracy issue. [-20] Image processing failed - image processor detected error reading from the network. [-21] Image processing failed - image processor detected a file metadata issue. [-22] Image processing failed - image processor detected a bad handle. [-23] Image processing failed - image processor detected a bad external page. [-88] Image processing failed - image processor detected a bad histogram. [-101] Image processing failed - image processor detected an illegal pointer. [-102] Image processing failed - image processor barcode driver failure. [-104] Image processing failed - image perfection profile operations string error. [-1007]	RTTI has detected an error during image processing of one of the images provided in the PUT or POST request.
209	Error	Image	Image processing failed - image failed dimension check: below minimum. [-500]	An image provided in the PUT or POST request is below the minimum dimension requirement specified in the image processing profile.
210	Error	Image	Image processing failed - image failed dimension check: above maximum. [-501]	An image provided in the PUT or POST request exceeds the maximum dimension requirement specified in the image processing profile.
211	Error	Image	Image processing failed - scanned image failed DPI check: below minimum. [-502]	An image provided in the PUT or POST request is below the minimum DPI requirement specified in the image processing profile.

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
212	Error	Image	Image processing failed - scanned image failed DPI check: above maximum. [-503]	An image provided in the PUT or POST request exceeds the maximum DPI requirement specified in the image processing profile.
401	Error	Image Processing	Invalid image processing profile - unknown error.	This error type is a "backstop" and will be thrown only for scenarios that are unknown to us now during image processing.
402	Error	Image Processing	Invalid image processing profile - file not specified for project mapping ID.	A file containing an image processing profile was not specified in the project element in the RTTI Web.config file.
403	Error	Image Processing	Invalid image processing profile - profile name mobileidtest not found in profiles list.	The image processing profile parameter provided in the PUT or POST request specifies a name that cannot be found in the ipProfiles section of the RTTI Web.config file.
404	Error	Image Processing	Invalid image processing profile - file C:\ip \TestIpProfile.txt does not exist.	The file specified for the image processing profile in either the project element of the RTTI Web.config file or the PUT or POST request is missing.
405	Error	Image Processing	Invalid image processing profile - file C:\ip \TestIpProfile.txt is empty.	The file specified for the image processing profile in either the project element of the RTTI Web.config file or the PUT or POST request is empty.
406	Error	Image Processing	Invalid alternate image processing profile - unknown error.	An RTTI internal error due to a catastrophic system-level error.
407	Error	Image Processing	Invalid alternate image processing profile - profile name mobileidtest not found in profiles list.	The alternate image processing profile parameter provided in the PUT or POST request specifies a name that cannot be found in the ipProfiles section of the RTTI Web.config file.

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
408	Error	Image Processing	Invalid alternate image processing profile - file C:\ip\mobileIdProfileTest.txt does not exist.	The file specified for the alternate image processing profile in the PUT or POST request is missing.
409	Error	Image Processing	Invalid alternate image processing profile - file C:\ip\mobileIdProfileTest.txt is empty.	The file specified for the alternate image processing profile in the PUT or POST request is empty.
410	Error	Image Processing	Exception occurred parsing IP profile C:\ip \mobileIdProfileTest.txt, message = Value cannot be null. Parameter name: node.	The file specified for the image processing profile in either the project element of the RTTI Web.config file or the PUT or POST request is formatted incorrectly.
411	Error	Image Processing	Exception occurred parsing alternate IP profile C:\ip\mobileIdProfileTest.txt, message = Value cannot be null. Parameter name: node.	The file specified for the alternate image processing profile in the PUT or POST request is formatted incorrectly.
601	Exception	Project	Error in Project Setup: <Project-specific error message>	RTTI detected an error during internal project setup. This error type is a "backstop" and will be thrown only for scenarios that are unknown to us now.
602	Error	Project	Project file C:\Program Files\Kofax \MobileIDCapture\KofaxIdentity2.fpr not found.	The Kofax Transformation Modules project file specified in the project element of the RTTI Web.config file is missing.
603	Exception	Project	Failed to load project because: Could not open file: C:\Program Files\Kofax \MobileIDCapture\KofaxIdentity2.fpr	The project file specified in the project element of the RTTI Web.config file could not be loaded by Kofax Transformation Modules. A likely cause is that the file does not exist.
301	Error or Exception	Classification	Error in Classification: <Project-specific error message>	RTTI detected a non-specific error during image classification. This error can only occur in a significant system event such as a hardware failure (disk drive).

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
302	Error	Classification	Image quality check: image may be missing borders.	RTTI detected an error during image classification that may have been caused by missing borders in the image. The likely cause is that the entire document cannot be seen in the image.
303	Error	Classification	Image quality check: image may have glare.	RTTI detected an error during image classification that may have been caused by glare in the image. The likely cause is that flash is being used or a bright, point light source sits right above the location where the image was taken.
304	Error	Classification	Image quality check: image may be poorly lit.	RTTI detected an error during image classification that may have been caused by a poorly lit image. The likely cause is that the image was taken in dark room.
305	Error	Classification	Image quality check: image may be overly lit.	RTTI detected an error during image classification that may have been caused by a overly lit image. The likely cause is that the flash is on, or an extremely bright light sits directly above the location where the image was taken. A dirty lens can also cause some of the same "washout" as an overly lit image.
306	Error	Classification	Image quality check: image may be blurred.	RTTI detected an error during image classification that may have been caused by a blurry image. The likely cause is that the user did not hold the camera steady.

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
307	Error	Classification	Image quality check: image may have shadows.	RTTI detected an error during image classification that may have been caused by shadows in the image. The likely cause is that something is obstructing the direct light over a portion of the document.
308	Error	Classification	Image quality check: image may be overly skewed.	RTTI detected an error during image classification that may have been caused by an overly skewed image. The likely cause is that the user is not holding their device parallel to the document they are capturing.
309	Error	Classification	Image quality check: image may have low contrast between the document and the background.	RTTI detected an error during image classification that may have been caused by a bad background in the image. The likely cause is that the background of the document and the background of the surface on which the document is being captured are similar in color.
310	Error	Classification	Error in Classification: Script execution has been stopped because of runtime error: Project: Line 155, Offset 0, (&H80131500) State not Recognized (Additional info: FireLocateAlternatives)	RTTI detected an error during image classification that indicates the provided image is not a valid ID. This is the error that will be returned if none of the other image checks (blur, glare, etc…) return true.
311	Error	Classification	Error in Classification: Script execution has been stopped because of runtime error: Project: Line 155, Offset 0, (&H80131500) Image not valid. (Additional info: FireLocateAlternatives)	RTTI detected an error during image classification that indicates the provided image cannot or has not been processed into a JPG or TIFF. The likely cause is that image processing on the server has been turned off.

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
801	Exception	Recognition	Error in Recognition: <Project-specific error message>	RTTI detected a non-specific error during image OCR. This will only occur if the KTM project has been modified an an error is returned from this custom KTM logic.
501	Exception	Extraction	Error in Extraction: <Project-specific error message>	RTTI detected a non-specific error during image data extraction. This error type is a "backstop" and will be thrown only for scenarios that are unknown to us now.
502	Error	Extraction	Image quality check: image may be missing borders.	RTTI detected an error during data extraction that may have been caused by missing borders in the image. he likely cause is that the entire document cannot be seen in the image.
503	Error	Extraction	Image quality check: image may have glare.	RTTI detected an error during data extraction that may have been caused by glare in the image. The likely cause is that flash is being used or a bright, point light source sits right above the location where the image was taken.
504	Error	Extraction	Image quality check: image may be poorly lit.	RTTI detected an error during data extraction that may have been caused by a poorly lit image. The likely cause is that the image was taken in dark room.
505	Error	Extraction	Image quality check: image may be overly lit.	RTTI detected an error during data extraction that may have been caused by a overly lit image.
506	Error	Extraction	Image quality check: image may be blurred.	RTTI detected an error during data extraction that may have been caused by a blurry image. The likely cause is that the user did not hold the camera steady.

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
507	Error	Extraction	Image quality check: image may have shadows.	RTTI detected an error during data extraction that may have been caused by shadows in the image. The likely cause is that something is obstructing the direct light over a portion of the document.
508	Error	Extraction	Image quality check: image may be overly skewed.	RTTI detected an error during data extraction that may have been caused by an overly skewed image. The likely cause is that the user is not holding their device parallel to the document they are capturing.
509	Error	Extraction	Image quality check: image may have low contrast between the document and the background.	RTTI detected an error during data extraction that may have been caused by a bad background in the image. The likely cause is that the background of the document and the background of the surface on which the document is being captured are similar in color.
510	Error	Extraction	Error in Extraction: Script execution has been stopped because of runtime error: Line 31, Offset 0, (&H80131500) Bad Parameters	RTTI detected an error during data extraction that indicates the provided image is not a valid passport. This is the error that will be returned if none of the other image checks (blur, glare, etc…) return true.
701	Exception	Validation	Error in Validation: <Project-specific error message>	RTTI detected a non-specific error during field validation. These are KTM validation errors and are not returned for any of our products out-of-the-box.

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
702	Error	Validation	Field {0} does not exist in the class {1}.	The field name provided in the Validation Web service request does not exist in the project class. RTTI detected a non-specific error during field validation. These are KTM validation errors and are not returned for any of our products out-of-the-box.
703	Error	Validation	Field values not found.	No field names were provided in the Validation Web service request. RTTI detected a non-specific error during field validation. These are KTM validation errors and are not returned for any of our products out-of-the-box.
704	Error	Validation	Project not found for class {0}.	The class name provided in the Validation Web service request could not be found in the Kofax Transformation Modules project.
705	Error	Validation	Invalid class parameter for field validation service.	A class name was not provided in the Validation Web service request. RTTI detected a non-specific error during field validation. These are KTM validation errors and are not returned for any of our products out-of-the-box.
901	Exception	License	An error occurred processing job b33f054e-1a7b-48ab-8219-4ad542fbe523. License 'Kofax Real-Time Transformation Interface' is not available.	The System License Kofax Real-Time Transformation Interface is not available.

Error Code	Result	Error Type	Example Error Description	Cause/Scenario
902	Exception	License	An error occurred processing job 6c70a2ad-c0e1-45a6-b92e-c242382491e1. License 'KTM Unlimited Fields Extraction' is not available. Error in Extraction: Script execution has been stopped because of runtime error: Line 31, Offset 0, (&H80131500) License 'Mobile ID - Server Extraction' is not available. Error in Classification: Script execution has been stopped because of runtime error: Project: Line 155, Offset 0, (&H80131500) License 'Mobile ID - Server Extraction' is not available. (Additional info: FireLocateAlternatives)	A Volume License for Kofax Transformation Modules is not available.
903	Exception	License	An error occurred processing job 0c7bf04f-1860-4523-af30-5322a561a1ca. [7001] SALicClnt: Unable to connect to the remote server - Unable to connect to the remote server An error occurred processing job 20d9790f-d0a4-4bbe-b9c0-558040592d9b. [7004] SALicClnt: No license servers configured. Check your license server configuration.	RTTI is unable to connect to or access the configured license server for this installation.

RTTI also supports the following operation string keywords:

- • _ Do90DegreeRotation_8: automatically rotate the image so the text is oriented normally and then, if necessary to make the output image be landscape orientation, rotate the image an additional 90 degrees clockwise.
- • _Do90DegreeRotation_9: automatically rotate the image so the text is oriented normally and then, if necessary to make the output image be landscape orientation, rotate the image an additional 270 degrees clockwise.