GETTING STARTED WITH KITEWORKS DEVELOPER GUIDE

Version 1.0

Version 1.0

Table of Contents

Introduction **3**

Get started **4**

kiteworks OAuth 2.0 Flow **5**

Obtaining Access Token **6**

Requirements 6

Sequence Overview 6

Step-by-step Usage 6

Execute API Calls **10**

APIs Overview **12**

Content APIs 13

Collaboration APIs 14

Preferences APIs 14

Contacts APIs 15

Security APIs 15

Client Management APIs 17

kiteworks maintenance APIs 17

Introduction

Welcome to **kite**works Development.

The kiteworks RESTful Enterprise APIs enable you to quickly develop mobile apps that leverage the power of the kiteworks Mobile Content Platform. The platform provides secure, ubiquitous connectivity to any type of content from any content storage supported by kiteworks, on any device, on-premises or in the cloud.

The platform provides an infrastructure to securely share, sync, email, and collaborate on content with employees and contacts. It saves you years of development because it is enterprise-ready out of the box. In addition to its own content storage, it integrates natively with Enterprise Content Management systems such as Microsoft SharePoint to easily make their content available to your mobile applications.

You can build custom apps that specialize the Mobile Content Platform to your industry and its use cases.

- For the auto insurance industry, an insurance broker can manage a repository for the documents related to each claim. Meanwhile, the claimants, adjusters, vendors and lawyers can use custom mobile apps to check off tasks as they add and view their photos, documents and forms related to the claim.
- A catalog of sales items and marketing collateral can be distributed to a worldwide sales force using tablets. Automated folder creation and permissions can ensure the right items and collateral are available to the right people's tablets at the right time.
- A bank can create a deal room for each major deal, and provide secure access to outside parties such as attorneys, agents and inspectors to upload and review documents using their tablets instead of a fax machine.

kiteworks REST APIs

kiteworks comes with a RESTful API set accessible from https://<hostname>/rest/index.html. You can familiarize yourself and test drive the APIs on this page, before developing your custom app.

Get started

To start your custom application development, perform the steps below.

1. **Go to the Accellion Developer Portal and signup for the kiteworks developer package.**
   You will receive a welcome email with instructions for downloading kiteworks. After you download kiteworks follow the instruction guide for installing and launching kiteworks.

2. **Sign in to kiteworks.**
   Once you have your instance of kiteworks up and running, sign in to the kiteworks admin interface with your user credentials. The admin interface can be accessed from the hostname of your kiteworks server /admin.
3. **Create your custom application to obtain the identifying information: the Client ID and Secret Key.**
   a. On the kiteworks Administrator's Dashboard, go to Application > Client Management > Custom Applications and add your custom application.

      - Specify the name for your app.
      - Under Flows select Authorization Code
      - Specify the Redirect URI using this format https://<kiteworks_server>/oauth_callback.php
      - Choose your Access Token Lifetime

   b. Add your Custom Application.
      You will be given the Client Application ID and Client Secret Key for your application.

      IMPORTANT: You must copy this information and keep it in a secure location. The Client Secret Key is required for authenticating your app. If you lose this information, you will have to start over and re-register your app.

4. **Configure your Custom Application.**

The kiteworks APIs are used by the custom applications to access user resources on a kiteworks server.

API Usage – The APIs follow the REST architectural style and use the scheme of addressing a resource and invoking a method on that resource.

- **The API URI** – All APIs can be called using the following URI scheme:
  https://<hostname>/rest/<resource>/<endpoint>
- **API Output** – The API result is returned in JSON format.

kiteworks OAuth 2.0 Flow

The kiteworks APIs allows any new client application (client for short) to be developed for the kiteworks solution. The APIs can be used by the client to gain access to resources belonging to a user on the kiteworks server. The APIs can only be used by a client that is registered on the kiteworks server.

A client must provide an access token to access resources belonging to a user on the kiteworks server. The kiteworks server provides access token provisioning flows based on the OAuth 2.0 (https://tools.ietf.org/html/rfc6749). The majority of clients will consume the so-called Authorization Code Flow to obtain an access token. This flow is developed based on the authorization code grant type of the OAuth 2.0 specification.

This document provides a step-by-step guide for application developers to build a client for consuming the Authorization Code Flow to obtain an access token and use the access token to access users' resources on a kiteworks server. Example codes for Android based clients are also provided.

Obtaining Access Token

Requirements

For obtaining an access token using the kiteworks Authorization Code Flow, you need the client registration information recorded in the previous steps:

- **client_id –** This is a unique system generated id of your client.
- **client_secret** – This secret serves as a password for your client to authenticate itself to the kiteworks server.
- **redirect_uri** – This is the URI on which your client must listen for the authorization result. For mobile clients or for clients that cannot be redirected to another service, the landing page https://<kiteworks_server>/oauth_callback.php can be used.
- **scope** – This is the set of API services that your client wants to access. Consult with your administrator regarding which scopes are available for your client.

Sequence Overview

The sequence of the Authorization Code Flow is as follows:

The client initiates the flow by redirecting the user-agent (browser or web view component) to the appropriate authorization page on the server. The client includes its id and a redirect URI to which the server will send the user back once access is granted or denied.

The server authenticates the user using a login page similar to web client login page and establishes whether the user grants or denies the client's access request.

If the user grants access, the server redirects the user-agent to the redirection URI provided earlier. The URI also includes an authorization code that can be used to request an access token for that user.

The client requests an access token from the server by authenticating itself (using its id and secret) and including the authorization code received in the previous step.

The server validates the client credentials and the authorization code and responds with the access token.

The client uses the access token to invoke APIs for accessing user's resources.

Step-by-step Usage

The request-response of this flow follows the specification of OAuth 2.0 protocol (http://tools.ietf.org/html/rfc6749#section-4.1). All requests for authorization and for calling service must be done through HTTPS. The URI end-points of this flow are as follows:

- Authorization end-point: https://<hostname>/oauth/authorize
- Token end-point: https://<hostname>/oauth/token

All request parameters, unless otherwise specified, must be passed through HTTP POST parameters. The response body will be in JSON format. The following information describes this in more detail.

Step 1 Authorization Request

The first step is to call the Authorization end-point with the request parameters passed via HTTP GET. Depending on the case, the user may be prompted with a dialog to authenticate and then to authorize the request for access permission by the client application. The following parameters must be passed in the request to the Authorize URI (this follows the OAuth 2 specification).

- **client_id –** is the identifier of the client-application as registered in the server. For example 'playground'.
- **redirect_uri –** is the URI to which the result of the authorization will be passed. This redirect URI must start with the URI specified at the time of the creation/registration of the client application. For example, if the client application had registered with the redirect URI of https://mydomain.com/oauth then the client application may provide https://mydomain.com/oauth/callback as `redirect_uri` parameter in this request. Please note that this parameter must be properly URL-encoded.
- **response_type –** the value of this parameter must be set to "code".
- **scope –** is the scope of the API services that the client wants to access. This is a space-separated string consisting of the name of the method and API services that the application requires. For example: "GET/users/* */files/*". The requested scope must be a sub-set of the client application's registered scope in the server. If a blank scope is provided, the registered scope will be assumed.
- **m (optional parameter) –** set to 1 to display mobile friendly authorization page.
- **state (optional parameter) –** is an optional parameter that the client application may pass in order to maintain the state of its process. The server will pass back this parameter as-is in the response.

Example:

(Note that line break is used only for clarity)

```
GET https://kiteworks_server/oauth/authorize?
client_id=abc&response_type=code&scope=&redirect_uri= https%3A%2F%2Fkiteworks_server%2Foauth_callback.php HTTP/1.1
```

Successful Response

After the server finishes the authorization and authentication procedure with the user, the server will redirect the user (via HTTP 302) to the `redirect_uri` provided in the Authorize call. Two parameters will be passed through this redirection URI: code and state. The code parameter is the authorization code that can be used to obtain the access token in the second step.

Example

```
HTTP/1.1 302 Found
Location: https://kiteworks_server/oauth_callback.php?code=60cc146c8dced75e26e
```

Error Response

If an error occurs (such as invalid consumer id, or invalid redirect URI), an error message will be displayed immediately within the user's browser. For other errors (such as invalid scope or denied access by the user) the server will redirect the user (via HTTP302) to the redirect_URI. The parameters are given below.

- **`error` –** is the error code. The following are the possible values of the error code:

  `access_denied`: The user denied the permission request.

`invalid_scope`: The requested scope is invalid.

`invalid_request`: The request is missing a required parameter, includes an unsupported parameter or parameter value, or is otherwise malformed.

`unauthorized_client`: The client-application is not authorized to use this flow.

- **`state` –** is set to the exact value received in the request.

Example:

```
HTTP/1.1 302 Found
Location: https:// kiteworks_server/oauth_callback.php?error=access_denied
```

Step 2 - Access Token Request

The authorization code obtained in the first step can be exchanged for the final access token by making a request to the access token end-point. The following parameters must be passed to the token end-point as POST parameters:

- **`client_id` –** is the ID of the client as registered in the server. E.g. '`playground`'.
- **`client_secret` –** is the client's secret phrase as registered in the server.
- **`grant_type` –** its value must be set to `authorization_code`.
- **`redirect_uri` –** is exactly the same redirect URI as used in the first step.
- **`code` –** is the authorization code obtained in the first step.
- **`install_tag_id` (optional parameter) –** is a string to uniquely identify the device from which the API call has initiated.
- **`install_name` (optional parameter) –** is the friendly name of the device from which the API call has initiated.

Example:

(Note that line breaks on the message content are used only for clarity)

```
POST /oauth/token HTTP/1.1
Host: kiteworks_server
Content-type: application/x-www-form-urlencoded

client_id=abc&client_secret=TheSecret&grant_type=authorization_code&code=c88bc36f751549adf60658c2c607a03b52e417bc&
redirect_uri= https%3A%2F%2Fkiteworks_server%2Foauth_callback.php
&install_tag_id=device_123&install_name=user_ipad
```

Successful Response

If the credentials of the client and the authorization code are valid and there is no other error, the server will return a HTTP response 200 OK. The body of the response is in JSON format with the following information:

- **access_token –** is the token that can be used to request an API service.
- **expires_in –** is the number in seconds after which the access token would expire.

- **token_type –** is set to “bearer”
- **scope –** is the scope for which this token is valid, normally it will be the same as the requested scope.
- **refresh_token –** is the refresh token that can be used to get a new access token without going through step 1 of Authorization. This refresh token will be provided only if the client is allowed to use refresh tokens as specified during client registration.

Example:

```
HTTP/1.1 200 OK
Cache-Control: no-store
Content-Type: application/json

{"access_token":"d932e1d32d89140163345d47fa97bfa60eeba1a5","expires_in":"360000","token_type":"bearer",
"scope":"GET\/users\/* *\/files\/*","refresh_token":"d7ce54d721e8das60943f3fc7cb159e4b11d0ee5"}
```

This access token can then be used to access user's resources through API services.

Error Response

If the credentials of the client or the authorization code is invalid or there is some other error, the server will respond with HTTP 400 Bad Request. The body of the response will contain the following error information in JSON format:

- **`error` –** is the error code. The following are the possible values :

  `invalid_client` – Client authentication failed. The client ID and/or secret key provided is invalid.

  `invalid_grant` – The authorization code or redirect URI provided is invalid.

  `invalid_scope` – The requested scope is invalid or exceeds the previously granted scope.

  `invalid_request` – The request is missing a required parameter, includes an unsupported parameter or parameter value, or is otherwise malformed.

  `unauthorized_client` – The client is not authorized to use this flow.

Execute API Calls

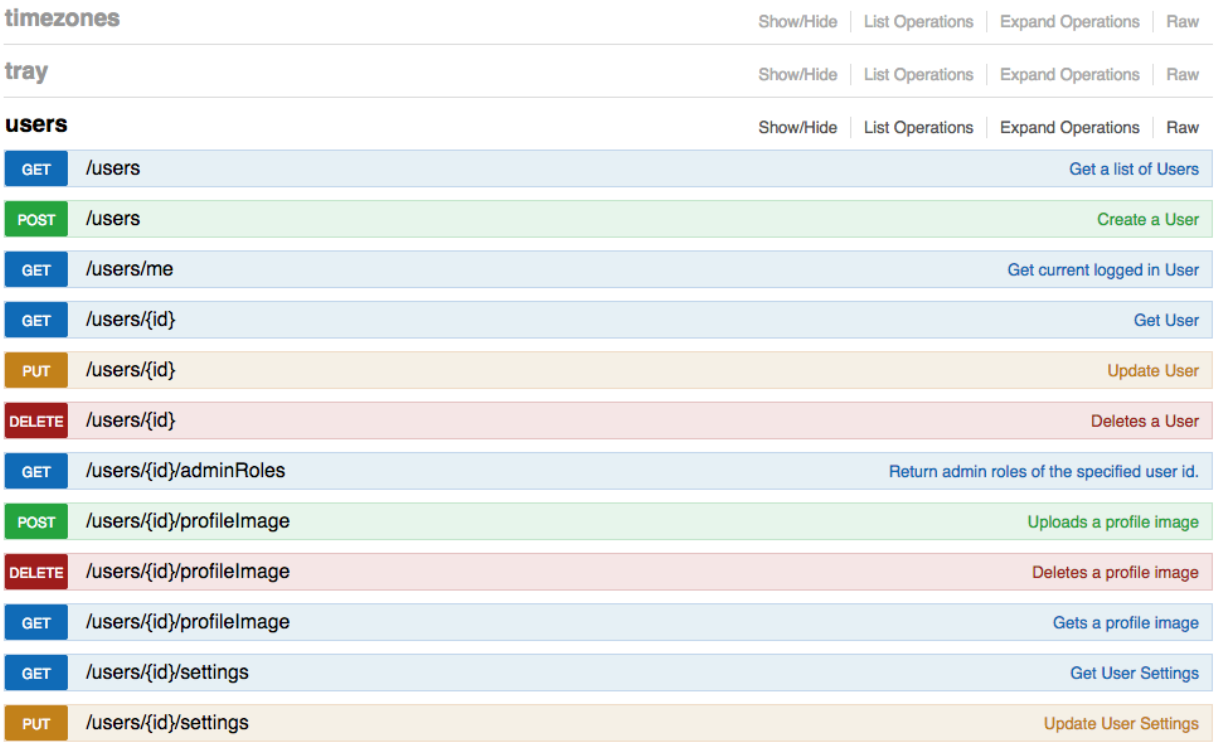
Now that we have the access token, it is time to get started with using the kiteworks API in your mobile application.

Before we begin, select the value of the access token returned from the response executed in the previous step. From the example response the value is: d932e1d32d89140163345d47fa97bfa60eeba1a5

Paste the token in the input field of the Developer Documentation page as shown below.

Let's begin with performing a basic API call. In the body of the webpage, there is a list of entities, each of which correspond to a different part of kiteworks and is represented with a JSON payload. When an entity name is clicked, the interface will expand to show all of the endpoints associated with that entity. An endpoint is a web request that performs a task related to the entity it falls under.

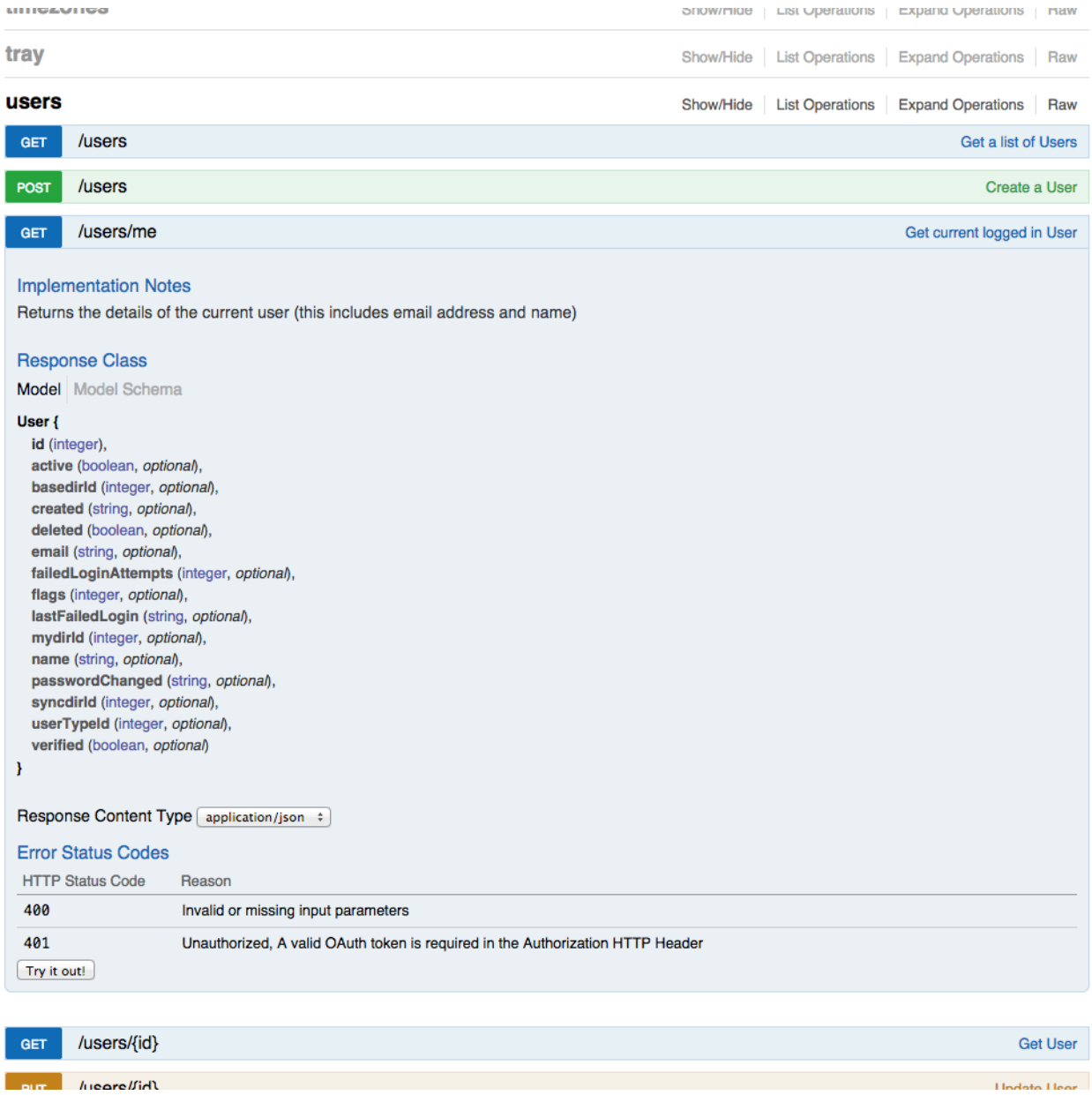
As an example, navigate to the **users** entity.



Here, you can see the list of endpoints associated with the **users** entity. For clarity purposes, they are color coded based on what method is being used. To the right of each list item, there is a brief description of what the entity does.

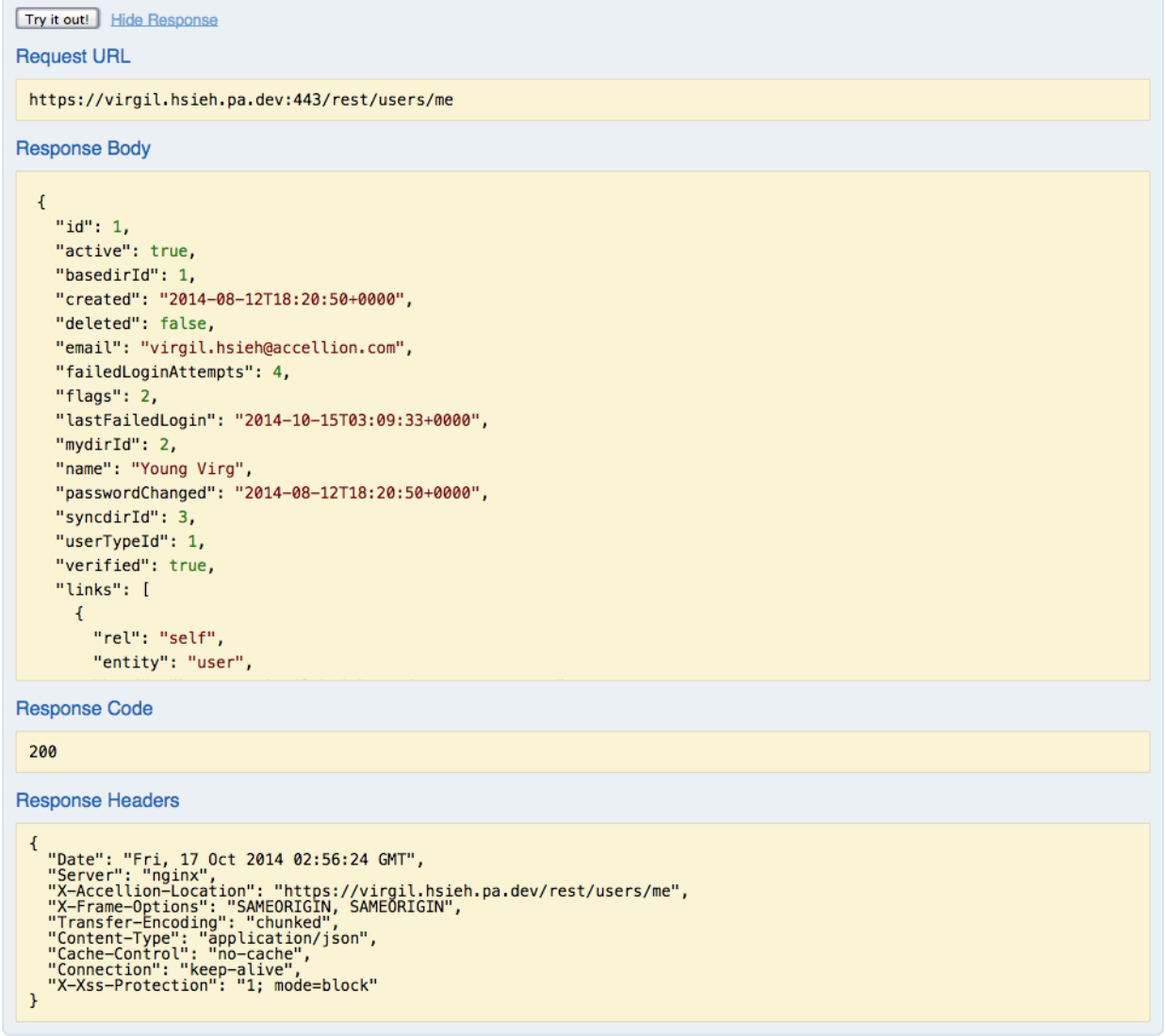
When an item in this list is clicked, the interface will expand again to show information relevant to the selected endpoint.

Click on the endpoint **GET /users/me**.



Go over the details for this endpoint. In summary, it returns a JSON representation of the entity that is populated with information that the access token represents. This information is tied to the current user.

Once you've gone over the information for the endpoint, click the **Try it out!** button.



In the **Response Body** section, there will be a JSON object of the users class, and it will contain the current user's information.

APIs Overview

kiteworks APIs provide broad coverage of the platform. The APIs can be categorized into Content, Collaboration, Preferences, Contacts, Security, Clients, and **kite**works Maintenance APIs.

Content APIs

Content-related APIs provide access to user content in your application. You will be able to access and manage files and folders as a part of the business flows of your app and work with files from various enterprise content sources like Microsoft SharePoint or EMC Documentum.

users

users APIs enable your application to obtain basic information about the user, user's root folders, and provide a starting point to further navigate through the files and folders the user has access to. By using the **/users/me** endpoint, you can obtain the ID of the user, the IDs of the root folder for this user, the email address or the name of the user, and status of the user (active, deleted).

Complete details of the **users** API is available at <your installation URL>/rest/index.html#!/users

folders

The next step is to work with the files and folders accessible to the authenticated user.

Complete details of the **folder** API is available at <your installation URL>/rest/index.html#!/folders

files

Together with **folders**, **files** are another fundamental entity that your application will have at its disposal.

Complete details of the **files** API is available at <your installation URL>/rest/index.html#!/files

sources

One of the advantages of the **kite**works Mobile Content Platform is its ability to securely connect to existing enterprise content sources through a single user interface. Using the **sources** APIs, your application can access and manage EC content sources in a similar fashion.

Complete details of the **sources** API is available at <your installation URL>/rest/index.html#!/sources

trays

The Move Tray is a powerful tool that allows end users to collect references to files they have access to, and later on copy or move them to a different folder, or mail them to other users.

Complete details of the **trays** API is available at <your installation URL>/rest/index.html#!/trays

Collaboration APIs

The collaboration-related APIs are intended to provide your application with the powerful collaboration tools that users have in **kite**works. In addition to being able to invite users to shared folders, these APIs allow users to collaborate on files and folders, construct and receive mail, add comments, and assign tasks.

mail

The **mail** APIs allow you to access emails sent and received on behalf of the user authenticated through your application.

Complete details of the **mail** API is available at <your installation URL>/rest/index.html#!/mail

comments

In addition to comments-related endpoints in **/files**, the **/comments/** endpoints allow you direct access to existing comments for update and delete actions.

Complete details of the **comments** API is available at <your installation URL>/rest/index.html#!/comments

tasks

Similar to the **/comments** endpoint, the **/tasks/** endpoints allow you direct access to existing tasks for update and delete actions.

Complete details of the **tasks** API is available at <your installation URL>/rest/index.html#!/tasks

Preferences APIs

kiteworks Mobile Content Management platform provides a set of APIs for preferences-related entities: folder notifications, favorite folders, languages, and time zones.

notifications

The **notifications** entity endpoints allow the management of notification settings for important folders in the system for the given user.

Complete details of the **notifications** API is available at <your installation URL>/rest/index.html#!/notifications

favorites

The **favorites** entity endpoints enable your application to manage the favorite folders for the authenticated user.

Complete details of the **favorites** API is available at <your installation URL>/rest/index.html#!/favorites

languages

The **languages** APIs provide your application with the ability to retrieve the languages supported by the kiteworks system.

Complete details of the **languages** API is available at <your installation URL>/rest/index.html#!/languages

timezones

The **timezones** APIs enable your application with the ability to list all the time zones supported in the system, and get the details about the time zones, like the name and time offset.

Complete details of the **timezones** API is available at <your installation URL>/rest/index.html#!/timezones

Contacts APIs

contacts

The **contacts** APIs provide your application with the ability to manage the user contacts.

Complete details of the **contacts** API is available at <your installation URL>/rest/index.html#!/contacts

groups

kiteworks provides end users with the ability to define personal contact groups. Personal groups can be used for allowing access to a folder, or for sending mail.

Complete details of the **groups** API is available at <your installation URL>/rest/index.html#!/groups

LDAP Groups

If your installation is integrated with LDAP, Administrators can enable LDAP groups to be available to end users when sharing folders. Your application will be able to add, update, or remove LDAP groups that are available to end users by utilizing the **/ldapGroups/** endpoints.

Complete details of the **ldapGroups** API is available at <your installation URL>/rest/index.html#!/ldapGroups

Security APIs

profiles

The **/profiles/** endpoints allow your application to manage the privileges assigned to **kite**works users. Your application can identify the list of User Profiles in the system and inspect the features and settings associated with each User Profile.

Complete details of the **profiles** API is available at <your installation URL>/rest/index.html#!/profiles

roles

The **roles** APIs allow you to get the details of folder roles in the system.

Complete details of the **ldapGroups** API is available at <your installation URL>/rest/index.html#!/ldapGroups

adminRoles

The **adminRoles** APIs allow your application to manage the assignment of Administrator roles to users.

Note: in order to use the adminRoles APIs your application will need to authenticate with an Administrator user. Additionally, only System Administrator can promote a user to System Administrator, Application Administrator. User cannot self-promote to System Administrator.

Complete details of the **adminRoles** API is available at <your installation URL>/rest/index.html#!/adminRoles

devices

Users may access **kite**works from various devices: mobile phones, tablets, etc. Device endpoints can be used to track access of individual devices to user accounts, and perform remote wipe on any device.

Complete details of the **device**s API is available at <your installation URL>/rest/index.html#!/devices

admin

The **admin** APIs allow your application to perform administrative actions on many entities that exist in the system - the endpoints are very similar to the endpoints of the actual entities, the difference is that your application will need to authenticate with an Administrator user in order to perform the calls.

- **Client Applications**: using **/admin/clients/** endpoints, you can create and configure the configuration settings of the client applications registered with **kite**works, and list API Scopes available.
- **Devices**: using /admin/devices/ endpoints you can find the list of user devices that are allowed to connect to **kite**works, log their access to the platform, and update, or remove them as necessary
- **LDAP Groups**: List, create or delete LDAP groups in the system
- **License**: Upload a new kiteworks license
- **Locations**: List, create, or delete kiteworks Locations: requires System Administrator, and is applicable for on-premises Enterprise and Enterprise Connect Packages only
- **Profiles**: List user Profiles. for details on profiles please refer to kiteworks Administration Guide
- **Sources**: List, add, update, remove Enterprise Connect sources
- **Users**: List, add, update, delete users, user settings, admin roles, profile image.

Complete details of the **admin** API is available at <your installation URL>/rest/index.html#!/admin

Client Management APIs

clients

Using the **clients** APIs, you can register a new application, and manage the configurations for the applications allowed to connect to **kite**works.

Complete details of the **clients** API is available at <your installation URL>/rest/index.html#!/clients

scopes

One of the main security and safety mechanisms for preventing unauthorized or accidental application use of the platform resources is the Administrator ability to set the API scopes for each application. **kite**works Mobile Content Platform provides the **/scopes** endpoint that allows applications to determine the APIs supported by the platform, so the application can properly construct the API calls to the platform.

Complete details of the **scopes** API is available at <your installation URL>/rest/index.html#!/scopes

kiteworks maintenance APIs

licenses

If your application is in charge of updating the license of your **kite**works installation it can use the **/licenses** endpoint to upload a new license.

Complete details of the **licenses** API is available at <your installation URL>/rest/index.html#!/licenses