

ACONEX API SAMPLE

MAIL SCHEMA

Introduction

This API sample sheet reviews the Aconex API Mail Schema service.

The Aconex application has large set of metadata fields of different types available. When a new project is started in Aconex, it is configured to use a subset of the available fields. Different labels of fields might be applied and list fields are selected or populated with unique values for the project.

The mail schema contains the selected projects current document metadata configuration and unique values.

To successfully use Aconex API services related to project mails, it is important to have a good understanding of the mail schema, how to read it and make use of it.

With the version 2 of mail schema, it's now broken down to four different schemas (create, search, forward or reply), depending on the API service to be used together with the schema. Also note that a schema can contain references to "mail form fields" or "restricted fields" schema service.

Mail schema

To get the create mail schema for a project, make the following API request.

Sample request:

<https://demo.aconex.co.uk/api/projects/1879048400/mail/schema/creation>

Sample response:

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<ComposeMailSchema>
  <MultiValueSchemaField>
    <Attributes>
      <EntityField MandatoryStatus="NOT_MANDATORY" />
    </Attributes>
    <DataType>STRING</DataType>
    <FieldName>Attribute 2</FieldName>
    <Identifier>Attribute2</Identifier>
    <SchemaValues>
      <SchemaValue>
        <Value>WP3.04 - Gutters</Value>
      </SchemaValue>
    ...
  </SchemaValues>
</MultiValueSchemaField>
<SingleValueSchemaField>
  <Attributes>
    <EntityField MandatoryStatus="CONDITIONAL">
      <MandatoryRules>
        <Rule>
          <Identifier>MailTypeId</Identifier>
          <SchemaValues>
            <SchemaValue>
              <Id>10</Id>
            </SchemaValue>
          </SchemaValues>
        </Rule>
      </MandatoryRules>
    </EntityField>
  </Attributes>
</SingleValueSchemaField>
</ComposeMailSchema>
```

```

        </Rule>
        <Rule>
            <Identifier>ResponseRequired</Identifier>
        </Rule>
    </MandatoryRules>
</EntityField>
</Attributes>
<DataType>DATE</DataType>
<FieldName>Response Required Date</FieldName>
<Identifier>ResponseRequiredDate</Identifier>
</SingleValueSchemaField>
</EntityCreationSchemaFields>
<SearchSchemaFields>
    <SingleValueSchemaField>
        <Attributes>
            <SearchResultField sortable="false" />
            <SearchableField />
        </Attributes>
        <DataType>DATE</DataType>
        <FieldName>Due</FieldName>
        <Identifier>responsedate</Identifier>
    </SingleValueSchemaField>
    ...
</ComposeMailSchema>

```

Each schema (create, search, forward or reply) contains enabled fields and valid values for that service. For the “search” schema, in the field attributes declaration, it is specified for example if a field is searchable and / or can be specified as a return field in the search result, see below sample where field can be both search on and returned in the response:

```

<Attributes>
    <SearchableField />
    <SearchResultField sortable="true" />
</Attributes>

```

Also note that the parameter identifiers for the same field, not always are spelled the same way.

Another important aspect of the mail schema is that it does **only** show the enable fields in a project. There is currently no API service which list all fields available, regardless if enabled or not. To be able to anticipate which fields that might appear in the schema, a full list of available fields is provided for reference below in this document.

A projects configuration can change during its lifespan. Any change will be reflected in the mail schema. The mail schema can also be different depending on user credentials; different user can have different access to the same project.

Some list value fields use id values along with the field value. To create mails, the id value needs to be used for these fields. To search, either id or string can be used, depending on selected identifier.

The list value fields, called “MultiValueSchemaField” exist in two variants, lists where only one value can be selected, and lists where multiple values can be selected. This information is currently not available in the schema. See table below to find this information.

A schema field can have mandatory status “Conditional”. It means that it depends on other selections made by user if this field is mandatory or not. This is specified in one or more rules. If any of the rules are met, field is conditional.

List of all available fields

Following is a table of all document fields available in Aconex. Additional to fields below, project can also contain specially setup “mail form fields” and “restricted fields”. Special API schema services exist to view these fields.

Field Name (Default)	Identifier	Field Control	Custom Values	Data Type	Size	Multi select	Id	Search Identifier
Mail type	MailTypeId	List	Yes	Integer		No	Yes	corrtypeid
To	ToUserId	User picklist	N.A.	Integer		Yes	No	
Cc	CcUserId	User picklist	N.A.	Integer		Yes	No	
Bcc	BccUserId	User picklist	N.A.	Integer		Yes	No	
Attribute 1	Attribute1	List	Yes	String	200	Yes	No	attribute
Attribute 2	Attribute2	List	Yes	String	200	Yes	No	secondaryattribute
Attribute 3	Attribute3	List	Yes	String	200	Yes	No	thirdattribute
Attribute 4	Attribute4	List	Yes	String	200	Yes	No	fourthattribute
Response Required	Response Required	List	No	Integer		No	Yes	
Response Required Date	Response RequiredDate	Date picker	N.A.	DateTime		N.A.	N.A.	
Reason for issue	ReasonForIssue	List	Yes	Integer		No	Yes	reasonforissuedid
Subject	MailSubject	Textbox	N.A.	String		N.A.	N.A.	subject
Body	MailBody	Textbox	N.A.	String		N.A.	N.A.	corrdata
Total Attachment Count	Total AttachmentCount	N.A.	N.A.	Integer		N.A.	N.A.	
Confidential	Confidential	N.A.	N.A.	Boolean		N.A.	N.A.	toconfidential fromconfidential
Rich Mail	RichMailText	N.A.	N.A.	Boolean		N.A.	N.A.	
Mail No	(Only search)	N.A.	N.A.	String		N.A.	N.A.	docno
Sent date	(Only search)	N.A.	N.A.	DateTime		N.A.	N.A.	sentdate
Due	(Only search)	N.A.	N.A.	DateTime		N.A.	N.A.	responsedate
From	(Only return)	N.A.	N.A.	Complex		N.A.	N.A.	fromUserDetails
From	(Only search)	N.A.	N.A.	String		N.A.	N.A.	fromuserfullname
From Organization Name	(Only search)	N.A.	N.A.	String		N.A.	N.A.	fromtradingname
Closed Out	(Only search)	N.A.	N.A.	DateTime		N.A.	N.A.	closedoutdate
Closed Out Details	(Only return)	N.A.	N.A.	Complex		N.A.	N.A.	closedoutdetails

To	(Only search)	N.A.	N.A.	String		N.A.	N.A.	tosuserfullname
To Organization Name	(Only search)	N.A.	N.A.	String		N.A.	N.A.	totradingname
Total file size all attachments	(Only return)	N.A.	N.A.	Long		N.A.	N.A.	totalAttachment Size
Status	(Only search)	List	N.A.	Integer		No	Yes	tostatusid
Mail ID	(Only search)	N.A.	N.A.	Integer		No	Yes	id
"All fields"	(Only search)	N.A.	N.A.	String		N.A.	N.A.	(use no parameter)

The "DateTime" format follows ISO8601 standard (<http://www.w3.org/TR/NOTE-datetime>), and all times should be in UTC/GMT absolute time.

All parameter identifiers are case sensitive.

References

Aconex Web Services - Developer's Guide.pdf
Aconex_Web_Services_Developer_Support_Information.pdf
Aconex_Web_Services_OAuth_Developer's_Guide.pdf
Aconex API Sample - Web Requests (C#).pdf
Aconex API Sample - Web Requests (Java).pdf
Aconex API Sample - Document Schema.pdf
Aconex API Sample - Document Parameters.pdf
Aconex API Sample - Mail Schema.pdf
Aconex API Sample - Paging.pdf
Aconex API Sample - Mail Download.pdf
Aconex API Sample - Performance Throttling.pdf
Aconex API Sample - Mail Form Fields.pdf
Aconex API Sample - OAuth.pdf

Disclaimer

This sample sheet information is only provided to exemplify how certain requests can be made using the Aconex API Web Services and should be treated as such. This sample sheet should not be considered best practice or an Aconex recommended implementation, as each integration project has its own unique challenges and need to be adopted thereafter.