



PUBLIC

Document Version: 2608 – 2026-08

SAP Ariba Custom Forms API

SAP Ariba Buying

SAP Ariba Buying and Invoicing

SAP Ariba Catalog

SAP Ariba Contract Invoicing

SAP Ariba Invoice Management



Content

- Custom Forms API. 3**
 - Use of the Custom Forms API. 3
 - Forms API. 4
 - Workflow of a Form Design API Application. 5
 - Form Documents API. 6
 - Workflow of a Form Document Application. 8
 - Response Example. 8
 - Forms Asynchronous Reporting API. 9
 - Workflow of the Forms Asynchronous Reporting API. 10
 - Creation of a Client Application to Use the Forms Asynchronous Reporting API. 10
 - Response Example When the Forms Asynchronous Reporting API Request Is Successful. 12
 - Response Example When the Forms Asynchronous Reporting API Request Fails. 16
- Version History for the SAP Ariba Custom Forms API. 18**
- API-Specific Disclaimers and Legal Information. 19**

Custom Forms API

This guide is for developers seeking to use the Custom Forms API set for extracting forms data.

Developers can create a client application on the SAP Ariba Developer Portal to extract forms data synchronously using the Forms API and the Forms Documents API, or asynchronously using the Forms Asynchronous Reporting API.

A typical use case for this API set is to get form document data for the purpose of reporting or sending to an external ERP system.

This guide applies to:

- SAP Ariba APIs
- SAP Ariba Developer Portal
- SAP Ariba Buying and Invoicing
- SAP Ariba Buying
- SAP Ariba Invoice Management

Related guides

[Help for the SAP Ariba Developer Portal](#)

[Forms Builder Quick Reference](#)

[Building Forms in the Forms Builder](#)

[Use of the Custom Forms API \[page 3\]](#)

[Forms API \[page 4\]](#)

[Form Documents API \[page 6\]](#)

[Forms Asynchronous Reporting API \[page 9\]](#)

Use of the Custom Forms API

For most use cases that involve synchronous data retrieval, you use the custom forms APIs in the following order:

1. Use the Forms API (forms-service-api) to get IDs and other information about form designs.
2. Once you have form design ID information from step 1, use the Form Documents API (form-documents-service-api) to get data from the submitted form documents associated with a specific form design.

To get form document data asynchronously, you use the Forms Asynchronous Reporting API.

After SAP Ariba maps your application to your SAP Ariba site, you can use the custom forms API as follows:

📌 Note

In the sample URLs in this topic, replace `{{runtime_url}}` with the desired runtime URL from the **Environment Details** table on the SAP Ariba Developer Portal discovery page for this API.

When multiple versions of the API are available on the developer portal, use the runtime URL specific to that version. Different versions distinguish their runtime URLs by modifying the version number, for example, v1, v2, and so on.

Request type: GET

URL: `{{runtime_url}}/forms`

Headers:

- `Authorization:Bearer{your access token}`
- `apikey{your application key}`
- `X-System{GuidedBuying, GuidedBuying_FuncDoc_EM, or Buyer}`
- `X-Tenant{name of your SAP Ariba site}`
- `X-AnId`

Forms API

The Forms API enables your application to get information about published form designs created using the forms builder.

The primary use case for this API is to get form design IDs. You then use the IDs with the Form Documents API (form-documents-api) to get the submitted form documents associated with a specific form design, based on the form design ID.

You can get information for all form design versions that were ever published on your SAP Ariba site, or you can use the `rsqlfilter` parameter to narrow the result set.

No changes can be made to form designs with this API.

Filtering the result set

If you don't filter the result set, the API returns all versions of all form designs that have ever been published on your site.

The `rsqlfilter` parameter lets you filter the result set using the following attributes: `status`, `name`, and `publishedBy`. These attributes are described in the following sections.

Separator to use when specifying multiple attributes: `and`

Filter Attribute	Description
status	Includes only versions of form designs that have the specified status. Valid status values are: • published: Form design versions that are published, have not been replaced by a later version, and have not been deleted. • obsolete: Form design versions that have been replaced by a later version and have not been deleted. • deleted: Form design versions that have been deleted. Operator to use with the status attribute: =in= Examples: • status=in=obsolete • status=in=published,obsolete • status=in=obsolete,deleted • status=in=published,obsolete,deleted
name	Includes versions of form designs that have the specified form title. You can't specify more than one name, but you can use the * wildcard. Operator to use with the name attribute: == Examples: • name==Charity Form • name==*rity* • name==Charity* • name==*Form
publishedBy	Includes version of form designs published by the user with the specified user name. You can't specify more than one user name. Also, values are case sensitive. Operator to use with the publishedBy attribute: == Example: • publishedBy==SPisa

[Workflow of a Form Design API Application \[page 5\]](#)

Workflow of a Form Design API Application

📌 Note

In the sample URLs in this topic, replace `{{runtime_url}}` with the desired runtime URL from the **Environment Details** table on the SAP Ariba Developer Portal discovery page for this API.

When multiple versions of the API are available on the developer portal, use the runtime URL specific to that version. Different versions distinguish their runtime URLs by modifying the version number, for example, v1, v2, and so on.

1. Form designers publish versions of form designs. They can also delete published versions of form designs.
2. The SAP Ariba application records information about the form designs, for example, form design IDs and titles.
3. The client application uses forms-service-api to request a list of form designs that have been published. The following example request URL gets information about published and deleted form designs that have the name that starts with "Charity" and were published by user SPisa:

```
{{runtime_url}}/forms?  
rsqlfilter=(status=in=(published,deleted) and name==Charity* and  
publishedBy==SPisa)&limit=10&offset=0
```

In the preceding example:

- *rsqlfilter* filters the list to include form design versions that are published or deleted, that have a form title starting with "Charity," and that were published by user SPisa. The filter doesn't include obsolete versions.
 - *limit* is set to 10, which limits the maximum number of results to 10.
 - *offset* is set to 0, which says to start returning results from the beginning of the result set.
4. The API call returns the requested information.
 5. The client application, using some of the information returned, uses form-documents-api to get form document data associated with a particular form design.

Form Documents API

The Form Documents API enables your application to get data from submitted form documents associated with a particular form design that was created using the forms builder.

You can get data from all form documents associated with a particular form design, or you can use the OData filter to narrow the result set.

No changes can be made to form documents with this API.

Filtering the result set

If you don't filter the result set, the API returns data for all form documents associated with all versions of a particular form design.

The OData filter lets you filter the result set using the following attributes: `status`, `formVersion`, `createDate`, `submittedDate`, `approvedDate`, and `lastUpdatedDate`. These attributes are described in the following sections.

Separator to use when specifying multiple attributes: `and`

Filter Attribute	Description
status	Includes only versions of form designs that have the specified approval status. Valid status values are: • submitted: Form documents that have been submitted. • approved: Form documents that are fully approved. • denied: Form documents that have been denied. Operator to use with the status attribute: eq Examples: • status eq approved • status eq approved or status eq denied • status eq approved or status eq denied or status eq submitted
createdDate	Includes form documents created during a specified date range. Operators to use with the createdDate attribute: ge for the "from" date, lt for the "to" date. Example: • createdDate ge 2016-07-25T11:00:00.000-07:00 and createdDate lt 2016-11-30T13:00:00.000-07:00 Returns form documents submitted during the following date range: On or after July 25, 2016, 11:00AM (PST), and before November 30, 2016, 01:00PM (PST)
submittedDate	Includes form documents submitted during a specified date range. Operators to use with the submittedDate attribute: ge for the "from" date, lt for the "to" date.
approvedDate	Includes form documents approved during a specified date range. Operators to use with the approvedDate attribute: ge for the "from" date, lt for the "to" date.
lastUpdatedDate	Includes form documents last updated on a specified date. Operators to use with the lastUpdatedDate attribute: eq.
formVersion	Includes form documents associated with a specific version of the form design. You can specify only one version at a time. Operator to use with the formVersion attribute: eq Example: • formVersion eq 1

[Workflow of a Form Document Application \[page 8\]](#)

[Response Example \[page 8\]](#)

Workflow of a Form Document Application

Note

In the sample URLs in this topic, replace `{{runtime_url}}` with the desired runtime URL from the **Environment Details** table on the SAP Ariba Developer Portal discovery page for this API.

When multiple versions of the API are available on the developer portal, use the runtime URL specific to that version. Different versions distinguish their runtime URLs by modifying the version number, for example, v1, v2, v3, and so on.

1. Users submit form documents. Approvers approve or deny the documents.
2. The SAP Ariba application records information about the form documents, such as the name of the submitter and the approval status.
3. The client application uses Form Documents API to get form document data associated with a particular form design. The API requires a form design ID, which the client application can get using Forms API. The following example request URL gets information about form documents associated with form design `bdff6e94-0f0e-4462-91fb-78a8e9198cd0`. The filter narrows the results by approval status, date range, and form design version.

```
{{runtime_url}}/documents
Request body:{
  "formId": "bdff6e94-0f0e-4462-91fb-78a8e9198cd0",
  "formVersion": "3"
  "filter": "(status eq 'approved' or status eq 'denied') and createDate ge
2016-07-25T11:00:00.000-07:00 and
createDate lt 2016-11-30T13:00:00.000-07:00 and )&$limit=10&$offset=0"
```

In the preceding example:

- The query filters the list to include approved and denied form documents that correspond to version 3 of the specified form ID and that were submitted within the following date range: on or after July 25, 2016, 11:00AM (PST) and before November 30, 2016, 01:00PM (PST)
 - `$limit` is set to 10, which limits the maximum number of results to 10.
 - `$offset` is set to 0, which says to start returning results from the beginning of the result set.
4. The API call returns the requested information. For sample response, see [Response Example \[page 8\]](#).

Response Example

The Form Documents API v3 extracts the following fields in the response.

See the following sample response:

```
{
  "content": [{
    "id": "26055",
    "uniqueName": "CF175",
    "name": "DT - public1 Document",
    "createdDate": "2018-10-26T23:51:37.307Z",
    "formId": "5073c4be-ed17-4e1d-8549-134bbb5cc7ab",
    "formVersion": 1,
    "userId": "cnoll",
```

```

    "data": [{
      "id": "54136251-37eb-44a2-83ce-838c40cc8ddb",
      "valueList": [{
        "lookupKeyValues": {
          "Domain": "unspsc",
          "UniqueName": "20121106"
        },
        "name": "Cement floating bulk units"
      }
    ],
    {
      "id": "4e84a5be-96f7-48a3-a926-5c6f233ee37d",
      "valueList": [
        [
          [
            {
              "name": "Wetting agents",
              "lookupKeyValues": {
                "Domain": "unspsc",
                "UniqueName": "12161906"
              }
            }
          ],
          [
            {
              "name": "Zuo Wei Wu",
              "lookupKeyValues": {
                "UniqueName": "zwwu",
                "PasswordAdapter": "PasswordAdapter1"
              }
            }
          ],
          "with lookup",
          "Option 1"
        ]
      ]
    },
    "status": "Submitted",
    "submittedDate": "2019-09-21T17:25:49.594Z",
    "approvedDate": null,
    "lastUpdatedDate": "2019-09-21T17:25:49.798Z"
  ]],
  "offset": 0,
  "limit": 20,
  "totalElements": 1,
  "lastPage": true,
  "firstPage": true,
  "totalOffsets": 1
}

```

Forms Asynchronous Reporting API

Using the new Forms Asynchronous Reporting API, you can create a client application on the SAP Ariba Developer Portal to filter form documents data and download them at scheduled intervals. The data can be downloaded as CSV or JSON files with a maximum size of 10 MB, and used for reporting or any other needs. The default download file format is JSON. If the data download fails, the API will retry the download for up to 3 times. If the download still does not work then it updates the status of the job to failure.

[Workflow of the Forms Asynchronous Reporting API \[page 10\]](#)

[Creation of a Client Application to Use the Forms Asynchronous Reporting API \[page 10\]](#)

[Response Example When the Forms Asynchronous Reporting API Request Is Successful \[page 12\]](#)

Workflow of the Forms Asynchronous Reporting API

The workflow for the Forms Asynchronous Reporting API includes the following high-level steps.

1. The client application posts a reporting job to the job queue, specifying the job type **REPORT** (form data, including contents and information about approvals for the form) or **DOCUMENT_EXPORT** (only approval information). The client application receives a job ID in response.
2. The client application polls the job status using the job ID in the API until the job is complete.
3. When the job is complete, the output is ready for download.
4. The client application requests the output document, specifying the job ID.
5. The downloaded ZIP file contains the output as a CSV or JSON file, depending on what was specified in the reporting job.
6. Optional. The client application requests a list of all submitted reporting jobs for a site, specifying the desired filter.

Creation of a Client Application to Use the Forms Asynchronous Reporting API

Note

In the sample URLs in this topic, replace `{{runtime_url}}` with the desired runtime URL from the **Environment Details** table on the SAP Ariba Developer Portal discovery page for this API.

Follow these steps to create a client application using the asynchronous form documents API for SAP Ariba Procurement solutions:

1. Post the initial request to the jobs queue, using the `/jobs` endpoint of the Forms Asynchronous Reporting API, specifying the user name of the user initiating the request in the request header, with the job type in the query string parameter and the criteria for filtering the form documents data in the request body. This request posts a job to the queue and returns the job ID.

Note

The SAP Ariba Developer Portal has a rate limit (requests) of 10 per second, 50 per minute, and 100 per hour. In addition to these rate limits, the Forms Asynchronous Reporting API has the following per-user and per-site limits:

The total number of report jobs that a user can create in an hour is limited to 2.

The total number of report jobs that a customer/site can create in an hour is limited to 5.

```
POST {{runtime_url}}/jobs?jobType=REPORT
Request body:{
  "formId": "abc12345-def67890",
```

```

    "formVersion": "1",
    "outputFormat": "CSV"
    "filter": "(documentStatus eq 'approved' or documentStatus eq 'submitted')
               and createdAt gt '2019-03-23T06:59:59.000Z' and
               (submittedDate ge '2019-03-23T06:59:59.000Z' or
                approvedDate eq '2019-03-23T06:59:59.000Z') )"
  }

```

Table 1: Request Header Parameter:

X-username	The user name of the user posting the request to create the job; this user name is returned in the response in the requestedBy field.
-------------------	--

Table 2: Request Query String Parameter:

jobType	The type of job to create. Specify REPORT to request all information about the form (contents and approvals for the form) or DOCUMENT_EXPORT to request only approval information.
----------------	--

Table 3: Request Body Parameters:

formId	The form ID of the form for which to get the documents.
formVersion	The version of the form for the specified form ID, for which to get the documents.
outputFormat	The output format in which the data is downloaded. The supported formats are CSV and JSON. The default format is JSON.
filter	The OData filter to identify form document data to download. The filter consists of the following query options: • documentStatus: Status of the form document. The status can be one of the following: • Approved • Denied • Submitted • Composing • createdAt: Date of creation of the document in ISO8601 or GMT format. • submittedDate: Date of submission of the document in ISO8601 or GMT format. • approvedDate: Date of approval of the document in ISO8601 or GMT format. • lastUpdatedDate: Date of last update of the document in GMT epoch Time. The following is an example of a valid date format: '2001-07-04T12:08:56.235+05:30'

2. Query the status of your reporting job using the `/jobs/{jobId}` endpoint of the Forms Asynchronous Reporting API. This request returns the job status.

```
GET {{runtime_url}}/jobs/{jobId}
```

Job status can be one of the following:

- **SUBMITTED**
- **In INPROGRESS**
- **COMPLETED**

- FAILED
3. Download the form document identified by your job, using the `/reports/{reportId}` endpoint of the Forms Asynchronous Reporting API. The size of the downloaded file can be up to 10 MB.

```
GET {{runtime_url}}/jobs/{jobId}/files
```

Table 4: Request Path Variable:

jobId	The job ID of your reporting job.
--------------	-----------------------------------

This request downloads the report file as a ZIP file with the file name in the following format: `{Form name}-{jobId}.zip`.

4. Optional. Request a list of all the reporting jobs by site using the `/jobs` endpoint of the Forms Asynchronous Reporting API.

```
GET {{runtime_url}}/jobs?$filter={filter}&$top={top}&$skip={skip}
```

Table 5: Request Header Parameters:

filter	The OData filter to identify the jobs to download. The filter consists of the following query options: • formId: The form ID of the form for which the report was downloaded. • formVersion: The version of the form for which the report was downloaded. • status: Status of the form document. The status can be one of the following: • Submitted • Retry • Cancelled • Inprogress • Failed • timeCreated: Date of creation of the document in ISO8601 or GMT format. • formName : Name of the form.
---------------	--

Response Example When the Forms Asynchronous Reporting API Request Is Successful

If a submitted API request is successful, the corresponding response body contains the fields described in this topic.

Table 6: API Requests and Responses:

API Request	API Response
Reporting job submission	Sample response: <pre data-bbox="820 336 1421 493">{ "jobId" : "abc1234-9090-def5678", "status": "SUBMITTED", "message" : "" }</pre> • jobId: The job ID of the submitted job. • status: The initial status of the job, which is SUBMITTED. • message: A message related to the job.
Reporting job status	Sample response: <pre data-bbox="820 714 1421 871">{ "jobId": "abc1234-9090-def5678", "status": "SUBMITTED", "message": "" }</pre> • jobId: The job ID of the job whose status is requested. • status: The retrieved status of the job. Status can be one of the following: • SUBMITTED • INPROGRESS • COMPLETED • FAILED • message: A message related to the job.

API Request

Reporting job list

API Response

Sample response:

```
[
  {
    "jobId": "",
    "status": "",
    "initiatedOn": "",
    "finishedOn": "",
    "errorDetails": ""
  },
  {
    "formId": "64407d21-b91c-46c0-bada-7208a0dc3fe3",
    "formVersion": "1",
    "filter": "documentStatus eq 'submitted'",
    "requestedBy": "cnoll",
    "timeCreated": "2019-08-19T14:08:14.379Z",
    "status": "COMPLETED",
    "jobId": "f3689890-b6ba-4739-b4f2-70af874eef24"
  },
  {
    "jobId": "",
    "status": "",
    "initiatedOn": "",
    "finishedOn": "",
    "errorDetails": ""
  }
]
```

- **jobId**: The job ID of a job.
- **status**: The status of the job.
- **initiatedOn**: The start date of the job.
- **finishedOn**: The end date of the job.
- **errorDetails**: Error details related to the job.

DOCUMENT_EXPORT Report Example

The following is an excerpt of a report created from a job using the **jobType** value **DOCUMENT_EXPORT**, which contains the following information about the approvals for the form:

- **approval_records**: Entries that record the events that occurred for the approval task.
- **approval_requests**: Entries that record the state changes for the approval task.
- **approval_comments**: Entries for comments submitted in the approval task.

```
{
  "id": 1045,
  "status": "Submitted",
  "requester": "areed",
  "document-link": "35900",
  "creation_time": "Apr 22, 2023 11:14:07 AM",
  "modification_time": "Apr 22, 2023 11:14:08 AM",
  "approval_records": [
    {
```

```

      "id": 1047,
      "creationTime": "Apr 22, 2023 11:14:08 AM",
      "modificationTime": "Apr 22, 2023 11:14:08 AM",
      "approvalEvent": "DocumentSubmitted",
      "author": "areed"
    },
    {
      "id": 1049,
      "creationTime": "Apr 22, 2023 11:14:35 AM",
      "modificationTime": "Apr 22, 2023 11:14:35 AM",
      "approvalEvent": "ApprovalGraphEdited",
      "author": "areed"
    },
    {
      "id": 1051,
      "creationTime": "Apr 22, 2023 11:15:19 AM",
      "modificationTime": "Apr 22, 2023 11:15:19 AM",
      "approvalEvent": "CommentAdded",
      "author": "afontaine"
    },
    {
      "id": 1053,
      "creationTime": "Apr 22, 2023 11:15:36 AM",
      "modificationTime": "Apr 22, 2023 11:15:36 AM",
      "approvalEvent": "RequestApproved",
      "author": "afontaine"
    }
  ],
  "approval_requests": [
    {
      "id": 1046,
      "creationTime": "Apr 22, 2023 11:14:08 AM",
      "modificationTime": "Apr 22, 2023 11:14:08 AM",
      "allApprovalsRequired": false,
      "approvalRequired": true,
      "ruleName": "node_1",
      "source": "FORM_SUBMIT",
      "status": "Active"
    },
    {
      "id": 1048,
      "creationTime": "Apr 22, 2023 11:14:34 AM",
      "modificationTime": "Apr 22, 2023 11:15:36 AM",
      "allApprovalsRequired": false,
      "approvalRequired": true,
      "approvedBy": "afontaine",
      "approverTime": "Apr 22, 2023 11:15:36 AM",
      "ruleName": "edit_1682162064746_1",
      "source": "USER_EDIT",
      "status": "Approved"
    }
  ],
  "approval_comments": [
    {
      "id": 1050,
      "creationTime": "Apr 22, 2023 11:15:19 AM",
      "modificationTime": "Apr 22, 2023 11:15:19 AM",
      "author": "afontaine",
      "active": true,
      "commentString": "first comment"
    },
    {
      "id": 1052,
      "creationTime": "Apr 22, 2023 11:15:36 AM",
      "modificationTime": "Apr 22, 2023 11:15:36 AM",
      "author": "afontaine",
      "active": true,
      "commentString": "approve time comment"
    }
  ]
}

```

```

    },
  ],
  "[object Object]": null
}

```

Response Example When the Forms Asynchronous Reporting API Request Fails

If a submitted API request fails, the corresponding response body is as in the following example:

```

{"fields": {}, "extraMessages":
{"exception": "ariba.form.exception.ReportDownloadException",
  "exceptionCause": "not available", "exceptionMessage": "The report with ID
abc1234-9090-def5678 has not been started"},

"status": 500, "code": "FRM-0110", "error": "ReportDownloadException", "service": "Forms",
  "msg": "The report with ID abc1234-9090-def5678 has not been started",
  "timestamp": 1568181395803, "errorId": "aDajPMH6-Form"}

```

The following is a list of errors to help you interpret the response when an API request fails:

Error code	Error message
FRM-0101	Error while creating new report job : %s
FRM-0102	The status of the report task is not Submitted or Inprogress
FRM-0103	Currently this report is in %s status and it cannot be updated to %s status
FRM-0104	The given action %s is not supported
FRM-0105	Error occurred while triggering reporting job, %s
FRM-0106	Form version is missing
FRM-0107	The report with ID %s has been cancelled
FRM-0108	The report with ID %s is still in progress
FRM-0109	The report with ID %s has failed
FRM-0110	The report with ID %s has not been started
FRM-0111	The report with ID %s was created successfully, but the file could not be found to download.
FRM-0112	The report with ID %s was not found in the system
FRM-0113	Form ID is missing
FRM-0114	The generation of report \"%s\" was interrupted, hence returning.
FRM-0115	Form metadata not found for form ID: %s and form version: %s
FRM-0116	Wrong status value provided: %s

Error code	Error message
FRM-0117	Generated file size exceeds 10 MB. Use filters to narrow the scope of the report.
FRM-0123	Date value cannot be a future date in: %s
FRM-0124	Invalid document status: %s. Valid status values are: %s.
FRM-0125	Incorrect date range specified for the field: %s
FRM-0126	Error while creating report detail entry : %s
FRM-0127	The action %s is not supported
FRM-0128	The number of records is %s. This exceeds the limit of %s.
FRM-0129	User not found for unique name: %s
FRM-0130	Document format is missing
FRM-0131	The specified document format is not supported. Specify CSV or JSON.

Version History for the SAP Ariba Custom Forms API

The Ariba Administrator Custom Forms API versions are listed in the following table.

API version	Status	As of release
v1	Deprecated	February 2019
v2	Deprecated	March 202
v3	Deprecated	2306 (June 2023)
v4	Active	2306 (June 2023)

Related Information

[API versioning and deprecation on the developer portal](#)

API-Specific Disclaimers and Legal Information

The SAP Ariba Developer Portal included in the SAP Ariba APIs product and the APIs made available on this site are provided solely at the discretion of SAP without warranty of any kind, and SAP may change, suspend, or cancel any or all features or functions of the SAP Ariba APIs product or revise the website at any time. Any production use of or commercialization of applications containing any APIs provided on this website is prohibited without a written agreement between your company and SAP governing such activities.

Access to this API is available to you as a subscriber to this solution as part of the SAP Cloud Service Level Agreement. However, it is not considered part of the solution. Use of this API is purely optional and is subject to restrictions stated in the documentation, including the Terms of Use and the documentation found at the SAP Ariba Developer Portal (see <https://developer.ariba.com/api>). If you wish to connect a third-party service using this API, first confirm that the company is participating in the SAP partner program and is authorized to provide a connection to this solution using this API. You will be required to submit written consent to SAP to authorize the exchange of data with the third-party service.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.

About the icons:

- Links with the icon  : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
 - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
 - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon  : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.

The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

Copyright © 2026 Ariba, Inc. All rights reserved.

This documentation, as well as the Ariba solutions, software and/or services described in it, contain proprietary information. They are provided under a license or other agreement containing restrictions on use and disclosure and are also protected by copyright, patent and/or other intellectual property laws. Except as permitted by such agreement, no part of the document may be reproduced or transmitted in any form by any means, electronic, mechanical or otherwise, without the prior written permission of Ariba, Inc.

Ariba, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in the documentation. The information contained in the documentation is subject to change without notice.

Ariba and Ariba products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Ariba, Inc. in the United States and other countries. Please see <http://www.ariba.com/legal/trademarks> for additional trademark information and notices.

Ariba Sourcing solutions (On Demand and software) are protected by one or more of the following patents, including without limitation: U.S. Patent Nos. 6,199,050; 6,216,114; 6,223,167; 6,230,146; 6,230,147; 6,285,989; 6,408,283; 6,499,018; 6,564,192; 6,871,191; 6,952,682; 7,010,511; 7,072,061; 7,130,815; 7,146,331; 7,152,043; 7,225,152; 7,277,878; 7,249,085; 7,283,979; 7,283,980; 7,296,001; 7,346,574; 7,383,206; 7,395,238; 7,401,035; 7,407,035; 7,444,299; 7,483,852; 7,499,876; 7,536,362; 7,558,746; 7,558,752; 7,571,137; 7,599,878; 7,634,439; 7,657,461; 7,693,747; 8,364,577; and 8,392,317. Patents pending.

Other Ariba product solutions are protected by one or more of the following patents:

U.S. Patent Nos. 6,199,050, 6,216,114, 6,223,167, 6,230,146, 6,230,147, 6,285,989, 6,408,283, 6,499,018, 6,564,192, 6,584,451, 6,606,603, 6,714,939, 6,871,191, 6,952,682, 7,010,511, 7,047,318, 7,072,061, 7,084,998; 7,117,165; 7,225,145; 7,324,936; 7,536,362; 8,364,577; and 8,392,317. Patents pending.

Certain Ariba products may include third party software or other intellectual property licensed from a third party. For information regarding software or other intellectual property licensed from a third party, go to <http://www.ariba.com/copyrights.cfm>.

