



ADOBE SIGN IMPLEMENTATION OF THE CLOUD SIGNATURE CONSORTIUM API AND PROTOCOLS

Adobe Systems Inc.

VERSION 6

Andrea Valle
avalle@adobe.com

Contents

Revision history	1
1.1 Remote Service Base URI.....	2
1.2 Integrity and confidentiality	2
1.3 Service authorization and authentication	2
1.4 Credential authorization.....	2
1.5 OAuth 2.0 Authorization.....	2
1.6 Creating a Remote Signature.....	5
1.7 The Remote Service APIs	5
info	6
auth/login	7
auth/revoke	8
oauth2/token.....	9
credentials/list	11
credentials/info.....	12
credentials/authorize	15
credentials/sendOTP.....	16
signatures/signHash.....	17
1.8 General Error messages.....	19

Revision history

Version	Date	Author	Version change details
1	9/12/2016	Andrea Valle	Specifications for 9.1 implementation (2017)
2	3/1/2017	Andrea Valle	Updated error messages
3	13/3/2017	Andrea Valle	Added support for Explicit/Offline OTP
4	7/4/2017	Andrea Valle	Corrected value of cert/status in credentials/info
5	15/5/2017	Andrea Valle	Corrected information about supported SCAL and update of Presence values for Input parameters
6	20/7/2018	Andrea Valle	General review and multiple parameter edits

1.1 Remote Service Base URI

`https://service.domain.org/csc/v0/`

where `service.domain.org` is the domain URL of the Remote Service.

1.2 Integrity and confidentiality

TLS 1.2 shall be implemented and supported by the Remote Service. Adobe Sign has deprecated the use of TLS 1.1, TLS 1.0 and SSL 3.

1.3 Service authorization and authentication

An OAuth 2.0 authorization mechanism shall be supported by the Remote Service. For security and confidentiality reasons, the Adobe Sign will not collect any security elements from the user, but instead it will redirect to the Remote Service that will authenticate the user in its own web based user interface via a standard OAuth 2.0 mechanism.

HTTP Basic authentication can be supported for test reasons, but it is not used by Adobe Sign.

1.4 Credential authorization

The Signature Activation Module (SAM) implemented by the Remote Service shall be determined by the level of compliance to meet (e.g. Advanced or Qualified).

At present, Adobe Sign supports the following types of credential authorizations:

- Explicit authorization, with a combination of static PIN and/or dynamic OTP based on the implementation of the Remote Service. At least one parameter from PIN and OTP shall be declared as present.
- Implicit authorization. This type of authorization shall be completed by the user within 1 minute, otherwise a timeout error will be generated.
- OAuth 2.0 credential authorization is being developed and planned to be delivered in October 2018. An update of this specification will be provided when the implementation will be available.

1.5 OAuth 2.0 Authorization

For service authentication, Adobe Sign only supports OAuth 2.0 Authorization Code flow implementations as described in RFC 6749.

In order to make the OAuth 2.0 authorization mechanism work, the Remote Service provider should provide Adobe with pairs of client credentials (Client ID and Client Secret as required) and register a list of Redirect URI addresses provided by Adobe. The means through which the Remote Service will exchange such information with Adobe are agreed upon between the parties and are described in other documents.

OAuth 2.0 Authorization Code [oauth2/authorize]

Parameter	Presence	Value	Description
<i>response_type</i>	Yes	<i>String</i> code	The value shall be “code”.
<i>client_id</i>	Yes	<i>String</i>	This is the unique “client ID” previously assigned to the Signature Application by the Remote Service.
<i>redirect_uri</i>	Yes	<i>String</i>	The URL where the user will be redirected after the authorization process has completed. Only a valid URI pre-registered with the Remote Service shall be passed. If omitted, the Remote Service will use the default redirect URI pre-registered by the Signature Application.
<i>scope</i>	Yes	<i>String</i> service	The scope of the access request as described by Section 3.3 of RFC 6749. <ul style="list-style-type: none"> “service”: it shall be used to obtain an access token suitable for service authorization.
<i>state</i>	Optional	<i>String</i>	Up to 255 bytes of arbitrary data from the Signature Application that will be passed back to the redirect URI. It can be used to handle a transaction identifier or other application-specific data. Normally this parameter is not used, but it could be activated by Adobe to prevent cross-site request forgery.
<i>lang</i>	Optional	<i>String</i> en-US	Request a preferred language according to RFC 3066. If specified, the Remote Service should render the authorization web page in this language, if supported. If omitted, the Remote Service shall render the authorization web page in its own default language.

Attribute	Presence	Value	Description
<i>code</i>	Required	<i>String</i>	The authorization code generated by the authorization server. It shall be bound to the client identifier and the redirection URI. It shall expire shortly after it is issued to mitigate the risk of leaks. The Signature Application cannot use the value more than once.
<i>state</i>	Conditional	<i>String</i>	Contains the arbitrary data from the signature application that was specified in the input request. It shall be returned only when specified in the request.
<i>error</i>	Conditional	<i>String</i>	A single error code string. See Paragraph 1.8 at the end of this document.
<i>error_description</i>	Conditional	<i>String</i>	Human-readable text providing additional error information. See Paragraph 1.8 at the end of this document.

Error Case	Status Code	Error	Error Description
Forbidden “client_id”: the user has clicked the “forbid” button of the client_id approval page	200 (OK) after redirect	unauthorized_client	client_id not allowed
The “client_id” has been accepted from the approval page, “access_token” parameter is missing though	200 (OK) after redirect	server_error	Missing access_token
The “client_id” has been accepted from the approval page, “credential_id” parameter is missing though	200 (OK) after redirect	server_error	Missing credential_id
Generated access_token is invalid	200 (OK) after redirect	server_error	Invalid access_token

Missing or not String "response_type" parameter	200 (OK) after redirect	invalid_request	Missing (or invalid type) string parameter response_type
Invalid "response_type" parameter	200 (OK) after redirect	invalid_request	Invalid parameter response_type
Missing or not String "client_id" parameter	200 (OK) after redirect	invalid_request	Missing (or invalid type) string parameter client_id
Missing or not String "redirect_uri" parameter	200 (OK) after redirect	invalid_request	Missing (or invalid type) string parameter redirect_uri
The "redirect_uri" parameter does not match any of the client_id registered redirect_uri regex	200 (OK) after redirect	invalid_request	redirect_uri parameter not allowed
When present, invalid "scope"	200 (OK) after redirect	invalid_request	Invalid parameter scope
Generated access_token has been lost after redirect from approval page	200 (OK) after redirect	server_error	Missing access_token
Generated access_token has been corrupted	200 (OK) after redirect	server_error	Invalid access_token
Invalid credentialID	200 (OK) after redirect	invalid_request	Invalid parameter credentialID
Missing or not integer "numSignatures" parameter	200 (OK) after redirect	invalid_request	Missing (or invalid type) integer parameter numSignatures
Invalid "numSignatures" value	200 (OK) after redirect	invalid_request	Invalid parameter numSignatures
Credential is locked	200 (OK) after redirect	invalid_request	Credential locked
Invalid PIN	200 (OK) after redirect	invalid_request	Invalid PIN
Invalid OTP	200 (OK) after redirect	invalid_request	Invalid OTP
Internal management: invalid user token (in this case the "token" meaning is linked with the OTP)	200 (OK) after redirect	invalid_request	Invalid token
Internal management: user token not found (in this case the "token" meaning is linked with the OTP)	200 (OK) after redirect	invalid_request	No login token found for this account

1.6 Creating a Remote Signature

The RSCD of the Remote Service shall be able to manage at least the remote signature of a single hash. At present, Adobe Sign only supports the creation of a single signature at once.

1.7 The Remote Service APIs

The following table summarizes all the APIs required or supported by the Adobe Sign.

URI	Description
info	Returns information on the Remote Service and the list of API methods it has implemented.
auth/login	Authorize the Remote Service with HTTP Basic authentication.
auth/revoke	Disable the service access token or refresh token.
oauth2/token	Obtain an OAuth 2.0 access token.
credentials/list	Returns the list of credentials associated to a user.
credentials/info	Retrieve a signing credential, its associated certificate and a description of the supported authorization mechanism.
credentials/authorize	Authorize the access to the credential for signing.
credentials/sendOTP	Start the online OTP mechanism associated to a credential.
signatures/signHash	Calculate a raw digital signature from one or more hash values.

info

Parameter	Presence	Value	Description
<i>lang</i>	Optional	<i>String</i>	Request a preferred language of the response to the Remote Service, specified according to RFC 3066. If present, the Remote Service shall provide language-specific responses using the specified language. If the specified language is not supported then it shall provide these responses in the language as specified in the <i>lang</i> output parameter.

Attribute	Presence	Value	Description
<i>specs</i>	Required	<i>String</i> 0.1.7.9	The version of this specification implemented by the provider. Adobe supports version 0.1.7.9g currently, the same this is currently available from the CSC website.
<i>name</i>	Required	<i>String</i>	The commercial name of the Remote Service.
<i>logo</i>	Required	<i>String</i>	The URI of the image file containing the logo of the Remote Service which shall be published online. The image shall be in either JPEG or PNG format and not larger than 256x256 pixels.
<i>region</i>	Required	<i>String</i>	The ISO 3166-1 Alpha-2 code of the Country where the Remote Service provider is established (e.g. ES for Spain).
<i>lang</i>	Required	<i>String</i>	The language used in the responses, specified according to RFC 3066. For example: en-US for US English.
<i>description</i>	Required	<i>String</i>	Contains a free form description of the Remote Service in the <i>lang</i> language. The maximum size of the string shall be 255 characters.
<i>authType</i>	Required	<i>Array of String</i>	Specifies one or more values corresponding to the authentication mechanisms supported by the Remote Service to authorize the access to the API: <ul style="list-style-type: none"> • “basic”: in case of HTTP Basic Authentication. • “oauth2code”: in case of OAuth 2.0 with authorization code flow.
<i>oauth2</i>	Required Conditional	<i>String</i>	Specifies the complete URI of the OAuth 2.0 service authorization endpoint provided by the Remote Service. The parameter is required only if the <i>authType</i> parameter contains “oauth2code”.
<i>methods</i>	Required	<i>Array of String</i>	Specifies the list of names of all the API methods described in this specification that are supported by the Remote Service.

auth/login

Parameter	Presence	Value	Description
<i>clientData</i>	Optional	<i>String</i>	Arbitrary data from the Signature Application. It can be used to handle a transaction identifier or other application-specific data.

Attribute	Presence	Value	Description
<i>access_token</i>	Required	<i>String</i>	The short-lived service access token used to authenticate the subsequent API requests within the same session. This shall be used as the value of the "Authorization: Bearer" in the HTTP header of the API requests. The access token has a limited time validity. In case of expired token, the Remote Service returns an error and a new auth/login request will be required.
<i>expires_in</i>	Optional	<i>Number</i>	The lifetime in seconds of the service access token. If omitted, the default expiration time is 3600 (1 hour).

Error Case	Status Code	Error	Error Description
<i>The authorization header does not match the basic HTTP authentication pattern ("Basic [base64]") - if refresh token is not present</i>	401 (unauthorized)	invalid_request	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.
<i>Decoded credentials are not in the form "username:password"</i>	400 (bad request)	invalid_request	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.
<i>Invalid refresh_token parameter format</i>	400 (bad request)	invalid_request	Invalid string parameter: refresh_token
<i>Invalid refresh_token value</i>	400 (bad request)	invalid_request	Invalid refresh_token
<i>Authentication error – login failed</i>	400 (bad request)	authentication_error	An error occurred during authentication process

auth/revoke

Parameter	Presence	Value	Description
<i>token</i>	Yes	<i>String</i>	The token that the Signature Application wants to get revoked.
<i>token_type_hint</i>	Yes	<i>String</i> access_token refresh_token	Specifies an optional hint about the type of the token submitted for revocation. If the parameter is omitted, the authorization server should try to locate the token across all the available tokens.
<i>clientData</i>	Optional	<i>String</i>	Arbitrary data from the Signature Application. It can be used to handle a transaction identifier or other application-specific data.

Error Case	Status Code	Error	Error Description
<i>The authorization header does not match the pattern "Bearer [sessionKey]"</i>	400 (bad request)	invalid_request	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.
<i>Missing or not String "token" parameter</i>	400 (bad request)	invalid_request	Missing (or invalid type) string parameter token
<i>"token_hint" parameter present, not equal to "access_token" nor "refresh_token"</i>	400 (bad request)	invalid_request	Invalid string parameter token_type_hint
<i>Invalid access_token or refresh_token</i>	400 (bad request)	invalid_request	Invalid string parameter token

oauth2/token

Parameter	Presence	Value	Description
<i>grant_type</i>	Yes	<i>String</i> authorization_code refresh_token	The grant type, which depends on the type of OAuth 2.0 flow: <ul style="list-style-type: none"> “authorization_code”: shall be used in case of Authorization Code Grant. “refresh_token”: shall be used in case of Refresh Token flow.
<i>code</i>	Conditional	<i>String</i>	The authorization code returned by the authorization server. It shall be bound to the client identifier and the redirection URI. This shall be used only when <i>grant_type</i> is “authorization_code”.
<i>refresh_token</i>	Conditional	<i>String</i>	The long-lived refresh token returned from the previous session. This shall be used only when the <i>grant_type</i> is “refresh_token” to reauthenticate the user according to the method described in Section 1.5 of RFC 6749.
<i>client_id</i>	Yes	<i>String</i>	This is the unique “client ID” previously assigned to the Signature Application by the Remote Service.
<i>client_secret</i>	Yes	<i>String</i>	This is the “client secret” previously assigned to the Signature Application by the Remote Service.
<i>redirect_uri</i>	Yes	<i>String</i>	The URL where the user was redirected after the authorization process completed. It is used to validate that it matches the original value previously passed to the Authorization Server. This shall be used only if the <i>redirect_uri</i> parameter was included in the authorization request, and their values shall be identical.
<i>clientData</i>	Optional	<i>String</i>	Arbitrary data from the Signature Application. It can be used to handle a transaction identifier or other application-specific data.

Attribute	Presence	Value	Description
<i>access_token</i>	Required	<i>String</i>	The short-lived access token to be used depending on the <i>scope</i> of the OAuth 2.0 authorization request. The Authorization Server returns a bearer token to be used as the value of the “Authorization: Bearer” in the HTTP header of the subsequent API requests within the same session.
<i>refresh_token</i>	Optional	<i>String</i>	The long-lived refresh token used to re-authenticate the user on the subsequent session based on the method described in Section 1.5 of RFC 6749. The presence of this parameter is controlled by the user.
<i>token_type</i>	Required	<i>String</i> Bearer	Specifies a “Bearer” token type as defined in RFC6750.
<i>expires_in</i>	Optional	<i>Number</i>	The lifetime in seconds of the service access token. If omitted, the default expiration time is 3600 (1 hour).

Error Case	Status Code	Error	Error Description
Missing or not String “client_id” parameter	400 (bad request)	invalid_request	parameter [client_id] cannot be empty
Missing or not String “client_secret” parameter	400 (bad request)	invalid_request	parameter [client_secret] cannot be empty
Missing or not String “grant_type” parameter	400 (bad request)	invalid_request	parameter [grant_type] cannot be empty
Invalid parameter “grant_type”	400 (bad request)	invalid_request	Invalid parameter grant_type
Missing or not String “code” parameter	400 (bad request)	invalid_request	parameter [code] cannot be empty

<i>Missing or not String "refresh_token" parameter</i>	400 (bad request)	invalid_request	parameter [refresh_token] cannot be empty
<i>Invalid "client_id" parameter</i>	400 (bad request)	invalid_request	Invalid parameter client_id
<i>Invalid "client_secret" parameter</i>	400 (bad request)	invalid_request	Invalid parameter client_secret
<i>The "redirect_uri" parameter does not match any of the client_id registered redirect_uri regex</i>	400 (bad request)	invalid_request	redirect_uri parameter not allowed
<i>Expired "access_token"</i>	400 (bad request)	invalid_request	Session expired
<i>The authorization header does not match the pattern "Bearer [sessionKey]"</i>	401 (unauthorized)	invalid_request	Missing access_token
<i>Missing access_token</i>	400 (bad request)	access_denied	Missing access_token
<i>Invalid access_token</i>	400 (bad request)	access_denied	Invalid access_token
<i>Expired "SAD"</i>	400 (bad request)	invalid_request	SAD expired
<i>Invalid "refresh_token" parameter</i>	400 (bad request)	invalid_request	Invalid parameter refresh_token
<i>Authorization code expired</i>	400 (bad request)	expired_token	Authorization code expired

credentials/list

Parameter	Presence	Value	Description
<i>maxResults</i>	No	<i>Number</i>	The Adobe Sign assumes a maximum number of 10 items is returned. However this number is implicit and the parameter is omitted.
<i>pageToken</i>	Optional	<i>String</i>	The page token for the new page of items. The parameter is only required to retrieve results other than the first page.
<i>clientData</i>	Optional	<i>String</i>	Arbitrary data from the Signature Application. It can be used to handle a transaction identifier or other application-specific data.

Attribute	Presence	Value	Description
<i>credentialIDs</i>	Required	<i>Array of String</i>	One or more credentialID associated with the provided or implicit <i>userID</i> .

Error Case	Status Code	Error	Error Description
<i>The authorization header does not match the pattern "Bearer [sessionKey]"</i>	400 (bad request)	invalid_request	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.
<i>"maxResults" < 1 or "maxResults" > 300</i>	400 (bad request)	invalid_request	Invalid parameter maxResults
<i>Invalid "pageToken" string format (not numeric)</i>	400 (bad request)	invalid_request	Invalid parameter pageToken
<i>Not empty "userID" parameter in case of user-specific authorization</i>	400 (bad request)	invalid_request	userID parameter must be null
<i>Invalid "userID" format in case of no user-specific authorization</i>	400 (bad request)	invalid_request	Invalid parameter userID

credentials/info

Parameter	Presence	Value	Description
<i>credentialID</i>	Yes	<i>String</i>	The identifier associated to the credential.
<i>certificates</i>	Yes	<i>String</i> chain	Specifies which certificates from the certificate chain shall be returned in <i>certs/certificates</i> . <ul style="list-style-type: none"> “chain”: the full certificate chain is returned.
<i>certInfo</i>	Yes	true	Specifies if the information on the end entity certificate shall be returned as printable strings. This is useful in case the Signature Application wants to retrieve some details of the certificate without having to decode it. The default value is “false”, so if the parameter is omitted then the information will not be returned.
<i>authInfo</i>	Yes	true	Specifies if the information on the authorization mechanisms supported by this credential (PIN and OTP groups) shall be returned. The default value is “false”, so if the parameter is omitted then the information will not be returned.
<i>lang</i>	Yes	<i>String</i>	Request a preferred language according to RFC 3066. If specified, the Remote Service should return the label and ddescription parameters in this language, if supported. If the language is not supported by the Remote Service, it should specify the default language in the output <i>lang</i> parameter. If omitted, the Remote Service should return these parameters in its own default language.
<i>clientData</i>	Optional	<i>String</i>	Arbitrary data from the Signature Application. It can be used to handle a transaction identifier or other application-specific data.

Attribute	Presence	Value	Description
<i>description</i>	Optional	<i>String</i>	A free form description of the credential in the <i>lang</i> language. The maximum size of the string is 255 characters.
<i>key/status</i>	Required	<i>String</i> enabled disabled	The status of enablement of the signing key of the credential: <ul style="list-style-type: none"> “enabled”: the signing key is enabled and can be used for signing. “disabled”: the signing key is disabled and cannot be used for signing. This may occur when the owner has disabled it or when the RSSP has detected that the associated certificate is expired or revoked.
<i>key/algo</i>	Required	<i>Array of String</i>	The list of OIDs of the supported key algorithms. The Adobe Sign currently supports at least 1.2.840.113549.1.1.11 = sha256WithRSAEncryption
<i>key/len</i>	Required	<i>Number</i>	The length of the cryptographic key in bits. The Adobe Signature Application requires a key length of 2048 bit or larger.
<i>cert/status</i>	Required	<i>String</i> valid expired revoked suspended	The status of validity of the end entity certificate. The Remote Service should only return a value that is consistent with the actual validity status of the certificate at the time the response is generated.
<i>cert/certificates</i>	Required Conditional	<i>Array of String</i>	Contains one or more Base64-encoded X.509v3 certificates from the certificate chain. If the <i>certificates</i> parameter is “chain”, the entire certificate chain shall be returned with the end entity certificate at the beginning of the array. If the <i>certificates</i> parameter is “single”, only the end entity certificate shall not be returned. If the

			<i>certificates</i> parameter is “none”, this parameter shall not be returned.
<i>cert/issuerDN</i>	Required Conditional	String	The Issuer Subject Distinguished Name from the X.509v3 end entity certificate in printable string format, UTF-8-encoded according to RFC 2253. This parameter shall be returned when <i>certInfo</i> is “true”.
<i>cert/serialNumber</i>	Required Conditional	String	The Serial Number from the X.509v3 certificate in hex encoded format. This parameter shall be returned when <i>certInfo</i> is “true”.
<i>cert/subjectDN</i>	Required Conditional	String	The Distinguished Name from the X.509v3 certificate in printable string format, UTF-8-encoded according to RFC 2253. This parameter shall be returned when <i>certInfo</i> is “true”.
<i>cert/validFrom</i>	Required Conditional	String	The validity start date from the X.509v3 certificate in printable string format, encoded as GeneralizedTime format (RFC 2459) (e.g. “YYYYMMDDHHMMSSZ”). This parameter shall be returned when <i>certInfo</i> is “true”.
<i>cert/validTo</i>	Required Conditional	String	The validity end date from the X.509v3 certificate in printable string format, encoded as GeneralizedTime format (RFC 2459) (e.g. “YYYYMMDDHHMMSSZ”). This parameter shall be returned when <i>certInfo</i> is “true”.
<i>authMode</i>	Required Conditional	String explicit	Specifies the authorization mode: <ul style="list-style-type: none"> • “explicit”: the signature application shall collect up to two levels of security elements.
<i>SCAL</i>	Optional	String 1 2	Specifies the Sole Control Assurance Level required by the credential, as defined in CEN EN 419 241-1: <ul style="list-style-type: none"> • “1”: at least a basic authorization is required (SCAL1). This level does neither require to invoke any of the credentials authorization methods nor to pass the Signature Activation Data (SAD) to the signatures/signHash method. • “2”: at least a two-factor authorization is required (SCAL2). This level requires the Remote Service to generate the Signature Activation Data (SAD). This parameter is optional and the default value is “1”.
<i>PIN/presence</i>	Required Conditional	String true false	Specifies if a PIN is required or not.
<i>PIN/format</i>	Required Conditional	String A N	Specifies the format of the PIN: <ul style="list-style-type: none"> • “A”: the PIN contains alphanumeric text. • “N”: the PIN contains numeric text. This parameter shall be present only when <i>PIN/presence</i> is not “false”. The size of the PIN is not specified to improve its security.
<i>OTP/presence</i>	Required Conditional	String true false	Specifies if an OTP is required or not.
<i>OTP/type</i>	Required Conditional	String offline online	Specifies the type of the OTP: <ul style="list-style-type: none"> • “offline”: The OTP is generated offline by a dedicated device and does not require the client to invoke the credentials/sendOTP method. • “online”: The OTP is generated online by the Remote Service by invoking the credentials/sendOTP method. This parameter shall be present only when <i>OTP/presence</i> is not “false”.
<i>OTP/format</i>	Required Conditional	String A N	Specifies the data format of the OTP: <ul style="list-style-type: none"> • “A”: the OTP contains alphanumeric text. • “N”: the OTP contains numeric text. This parameter shall be present only when <i>OTP/presence</i> is not “false”.

<i>multisign</i>	No	<i>Boolean</i>	Specifies if the credential supports multiple signatures to be created with a single authorization request. The Adobe Sign does not support multisign.
<i>lang</i>	Optional	<i>String</i>	The language used in the responses, specified according to RFC 3066.

Error Case	Status Code	Error	Error Description
<i>The authorization header does not match the pattern "Bearer [sessionKey]"</i>	400 (bad request)	invalid_request	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.
<i>Missing or not String "credentialID" parameter</i>	400 (bad request)	invalid_request	Missing (or invalid type) string parameter credentialID
<i>Invalid "credentialID" parameter</i>	400 (bad request)	invalid_request	Invalid parameter credentialID
<i>Invalid "certificates" parameter</i>	400 (bad request)	invalid_request	Invalid parameter certificates

NOTE: At present, the Adobe Sign Application requires at least one of *PIN/presence* or *OTP/presence* parameters to be `true`.

credentials/authorize

Parameter	Presence	Value	Description
<i>credentialID</i>	Yes	<i>String</i>	The identifier associated to the credential.
<i>numSignatures</i>	Yes	<i>Number</i> 1	The number of signatures to authorize. The Adobe Sign does not support multisign.
<i>hash</i>	Yes	<i>Array of String</i>	One Base64-encoded hash value to be signed. It is sent by Adobe Sign regardless the value of the <i>SCAL</i> parameter returned by credentials/info .
<i>PIN</i>	Conditional	<i>String</i>	The PIN collected from the user. It shall be used only when <i>PIN/presence</i> is not "false".
<i>OTP</i>	Conditional	<i>String</i>	The OTP collected from the user. It shall be used only when <i>OTP/presence</i> is not "false".
<i>clientData</i>	Optional	<i>String</i>	Arbitrary data from the Signature Application. It can be used to handle a transaction identifier or other application-specific data.

Parameter	Presence	Value	Description
<i>SAD</i>	Required	<i>String</i>	The Signature Activation Data to provide as input to the signatures/signHash method.
<i>expiresIn</i>	Optional	<i>Number</i>	The lifetime in seconds of the service access token. If omitted, the default expiration time is 3600 (1 hour).

Error Case	Status Code	Error	Error Description
<i>The authorization header does not match the pattern "Bearer [sessionKey]"</i>	400 (bad request)	invalid_request	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.
<i>Missing or not String "credentialID" parameter</i>	400 (bad request)	invalid_request	Missing (or invalid type) string parameter credentialID
<i>Invalid "credentialID" parameter</i>	400 (bad request)	invalid_request	Invalid parameter credentialID
<i>Missing or not integer "numSignatures" parameter</i>	400 (bad request)	invalid_request	Missing (or invalid type) integer parameter numSignatures
<i>"numSignatures" < 1</i>	400 (bad request)	invalid_request	Invalid parameter numSignatures
<i>When present, invalid "clientData" format (not string)</i>	400 (bad request)	invalid_request	Invalid parameter clientData
<i>Invalid OTP</i>	400 (bad request)	invalid_otp	The OTP is invalid
<i>Invalid PIN</i>	400 (bad request)	invalid_pin	The PIN is invalid
<i>PIN locked</i>	400 (bad request)	invalid_request	PIN locked
<i>OTP locked</i>	400 (bad request)	invalid_request	OTP locked

credentials/sendOTP

Parameter	Presence	Value	Description
<i>credentialID</i>	Yes	<i>String</i>	The identifier associated to the credential.
<i>clientData</i>	Optional	<i>String</i>	Arbitrary data from the Signature Application. It can be used to handle a transaction identifier or other application-specific data.

Error Case	Status Code	Error	Error Description
<i>The authorization header does not match the pattern "Bearer [sessionKey]"</i>	400 (bad request)	invalid_request	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.
<i>Missing or not String "credentialID" parameter</i>	400 (bad request)	invalid_request	Missing (or invalid type) string parameter credentialID
<i>Invalid "credentialID" parameter</i>	400 (bad request)	invalid_request	Invalid parameter credentialID
<i>OTP locked</i>	400 (bad request)	invalid_request	OTP locked

signatures/signHash

Parameter	Presence	Value	Description
<i>credentialID</i>	Yes	<i>String</i>	The identifier associated to the credential.
<i>SAD</i>	Yes	<i>String</i>	The Signature Activation Data returned by the Credential Authorization methods.
<i>hash</i>	Yes	<i>Array of String</i>	One Base64-encoded hash value to be signed.
<i>hashAlgo</i>	Conditional	<i>String</i>	Specifies the OID of the algorithm used to calculate the hash value(s), in case it's not implicitly specified by the <i>signAlgo</i> algorithm. Only hashing algorithms as strong or stronger than SHA256 shall be used.
<i>signAlgo</i>	Yes	<i>String</i> 1.2.840.113549.1.1.11 or 1.2.840.113549.1.1.10	Specifies the OID of the algorithm to use for signing. The Adobe Sign will only sign using the sha256WithRSAEncryption or RSASSA-PSS algorithms.
<i>signAlgoParams</i>	Conditional	<i>String</i>	Specifies the Base64-encoded or DER-encoded ASN.1 signature parameters, if required by the signature algorithm. It's used for RSA-PSS [RFC 3447]. Contact Adobe to get more information about RSA-PSS support.
<i>clientData</i>	Optional	<i>String</i>	Arbitrary data from the Signature Application. It can be used to handle a transaction identifier or other application-specific data.

Parameter	Presence	Value	Description
<i>signatures</i>	Required	<i>Array of String</i>	One Base64-encoded signed hash.

Error Case	Status Code	Error	Error Description
<i>The authorization header does not match the pattern "Bearer [sessionKey]"</i>	400 (bad request)	invalid_request	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.
<i>Missing or not String "SAD" parameter</i>	400 (bad request)	invalid_request	Missing (or invalid type) string parameter SAD
<i>Invalid "SAD" parameter</i>	400 (bad request)	invalid_request	Invalid parameter SAD
<i>Missing or not String "credentialID" parameter</i>	400 (bad request)	invalid_request	Missing (or invalid type) string parameter credentialID
<i>Invalid "credentialID" parameter</i>	400 (bad request)	invalid_request	Invalid parameter credentialID
<i>Missing or not Array "hash" parameter</i>	400 (bad request)	invalid_request	Missing (or invalid type) array parameter hash
<i>Empty hash parameter</i>	400 (bad request)	invalid_request	Empty hash array
<i>Invalid Base64 hash element</i>	400 (bad request)	invalid_request	Invalid Base64 hash string parameter
<i>Missing or not String "signAlgo" parameter</i>	400 (bad request)	invalid_request	Missing (or invalid type) string parameter signAlgo
<i>Missing or not String "hashAlgo" parameter when "signAlgo" is equal to "1.2.840.113549.1.1.1"</i>	400 (bad request)	invalid_request	Missing (or invalid type) string parameter hashAlgo
<i>Invalid "hashAlgo" parameter</i>	400 (bad request)	invalid_request	Invalid parameter hashAlgo
<i>Invalid "signAlgo" parameter</i>	400 (bad request)	invalid_request	Invalid parameter signAlgo

<i>When present, invalid "clientData" format (not string)</i>	400 (bad request)	invalid_request	Invalid parameter clientData
<i>Invalid "hash" length</i>	400 (bad request)	invalid_request	Invalid digest value length
<i>The OTP used to generate the "SAD" is invalid</i>	400 (bad request)	invalid_otp	The OTP is invalid
<i>Expired credential</i>	400 (bad request)	invalid_request	Signing certificate 'O=[organization],CN=[common_name]' is expired.

1.8 General Error messages

These error messages are not bound to a specific method and may occur in several situations.

Error Case	Status Code	Error	Error Description
<i>Invalid access_token</i>	400 (bad request)	invalid_session_key	Session is invalid
<i>Unexpected error</i>	500 (internal server error)	server_error	An unexpected error occurred.
<i>Invalid request body format</i>	400 (bad request)	invalid_request_format	Request format is invalid
<i>Wrong service name/version</i>	501 (not implemented)	access_denied	The user or Remote Service denied the request.