

SIGNING DOCUMENTS

- [Publish Documents](#)
- [Sign Documents Sync](#)

Publish Documents

Integrating Digital Document Signing with ADO Technologies

For clients looking to incorporate digital document signing capabilities into their platforms, ADO Technologies offers a robust solution that requires the implementation of specific web services. This guide outlines the necessary steps to enable ADO's solution to retrieve documents for signing, focusing on the RESTful web service that utilizes OAuth2 authentication and exposes essential methods for the digital signing process.

Required Web Service Methods

To facilitate digital document signing, your platform must expose a RESTful web service with OAuth2 authentication, detailing the following methods:

Token Generation (Authentication)

- **Method:** POST
- **Description:** Generates an authentication token to access other methods of the service.
- **Parameters:**
 - `grant_type` (String, FormData): Specifies the HTTP authentication type.
 - `username` (String, FormData): Assigned username for token retrieval.
 - `password` (String, FormData): Corresponding password for the assigned username.

Example Request:

```
“ POST /api/token HTTP/1.1
Host: localhost:62859
Content-Type: application/x-www-form-urlencoded
username=admin&password=password&grant_type=password
```

Response Fields:

- `access_token` (String): The issued token.
- `token_type` (String): Type of the generated token.

- `expires_in` (Int): Token validity period in minutes.
- `issued` (String): Token issuance date and time.
- `expires` (String): Token expiration date and time.

Retrieve Documents for Signing

- **Method:** POST
- **Description:** Returns a list of documents to be signed.
- **Parameters:**
 - `JsonTransaction` (Json, Body): JSON object containing all transaction information in process.
 - `Authorization` (String, Header): Authentication token prefixed with "Bearer ".

Example Request

```
“ POST /api/Integration/Documents HTTP/1.1
Host: localhost:62859
Authorization: Bearer your_access_token
Content-Type: application/json
{
  "JsonTransaction": {
    // Transaction details
  }
}
```

Response

An array of strings, each containing a document in base64 format to be signed.

Implementing the Service

- **OAuth2 Authentication:** Ensure your service supports OAuth2 for secure access control. The token endpoint must correctly handle the provided credentials to issue tokens.
- **Service Endpoints:** Implement the `Token` and `GetDocuments` methods according to the specifications, ensuring they process requests and return the expected responses.
- **Error Handling:** Properly manage exceptions and validate request parameters to return appropriate HTTP status codes and messages for error conditions.

Sign Documents Sync

Integrating Document Signing with ADO Technologies

The synchronous document signing process allows clients to sign PDF documents in real-time. This process involves obtaining an authentication token and then using that token to sign the documents. The following steps outline how to interact with the API to achieve this.

1. **Obtain Authentication Token:** First, authenticate the service and obtain an access token via the OpenID Connect `client_credentials` grant type.
2. **Sign Documents:** Use the obtained token to submit PDF documents for signing, along with the client information or identity validation transaction number.

Token Generation (Authentication)

This endpoint authenticates the service and obtains an access token via OpenID Connect.

- **Method:** POST
- **Description:** Generates an authentication token to access other methods of the service.
- **Parameters:**
 - `client_id` (String, FormData): Specifies the client ID assigned for token retrieval.
 - `client_secret` (String, FormData): Specifies the client secret assigned for token retrieval.
 - `grant_type` (String, FormData): Specifies the grant type for token retrieval. It should be `client_credentials`.

Example Request

```
curl -X 'POST' \
  'https://example.com/token' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  -d
  'client_id=your_client_id&client_secret=your_client_secret&grant_type=client_cr
```

redentials'

Responses

200 OK: Access token obtained successfully.

```
“ {
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "token_type": "Bearer",
  "expires_in": 3600
}
```

400 Bad Request: Invalid request.

```
“ {
  "error": "Invalid credentials."
}
```

Document Signing Endpoint

This endpoint receives PDF documents and the client information or identity validation transaction number, and returns the list of signed documents with their reference, internal document number, signed document, status, and error reason if applicable.

- **Method:** POST

Example Request

```
“ curl -X 'POST' \
  'https://example.com/sign-documents' \
  -H 'accept: application/json' \
  -H 'Content-Type: multipart/form-data' \
  -H 'x-account-id: your_account_id' \
  -H 'x-project-id: your_project_id' \
  -H 'Authorization: Bearer your_access_token' \
  -F 'documentPairs[0].referenceNumber=ref123' \
  -F 'documentPairs[0].document=@/path/to/your/document.pdf' \
```

```
-F 'clientInfo=transactionNumber' \  
-F 'x1=300' \  
-F 'y1=300' \  
-F 'x2=500' \  
-F 'y2=150' \  
-F 'signaturePage=0'
```

Responses:

200 OK: List of signed documents with their reference, internal document number, signed document, status, and error reason if applicable.

```
{  
  "signedDocuments": [  
    {  
      "referenceNumber": "ref123",  
      "documentId": "doc001",  
      "signedDocument": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",  
      "status": "SUCCESS",  
      "errorReason": null  
    },  
    {  
      "referenceNumber": "ref456",  
      "documentId": "doc002",  
      "signedDocument": null,  
      "status": "FAIL",  
      "errorReason": "Identity validation error."  
    }  
  ]  
}
```

400 Bad Request: Invalid request.

```
{  
  "error": "Missing documents or client information."  
}
```

401 Unauthorized: Unauthorized. The token was not provided or is invalid.

```
{  
  "error": "Token not provided or invalid."  
}
```

403 Forbidden: Forbidden. The token has expired.

```
“ {  
  "error": "Token expired."  
}
```

500 Internal Server Error: Internal server error.

```
“ {  
  "error": "Error signing the documents."  
}
```