

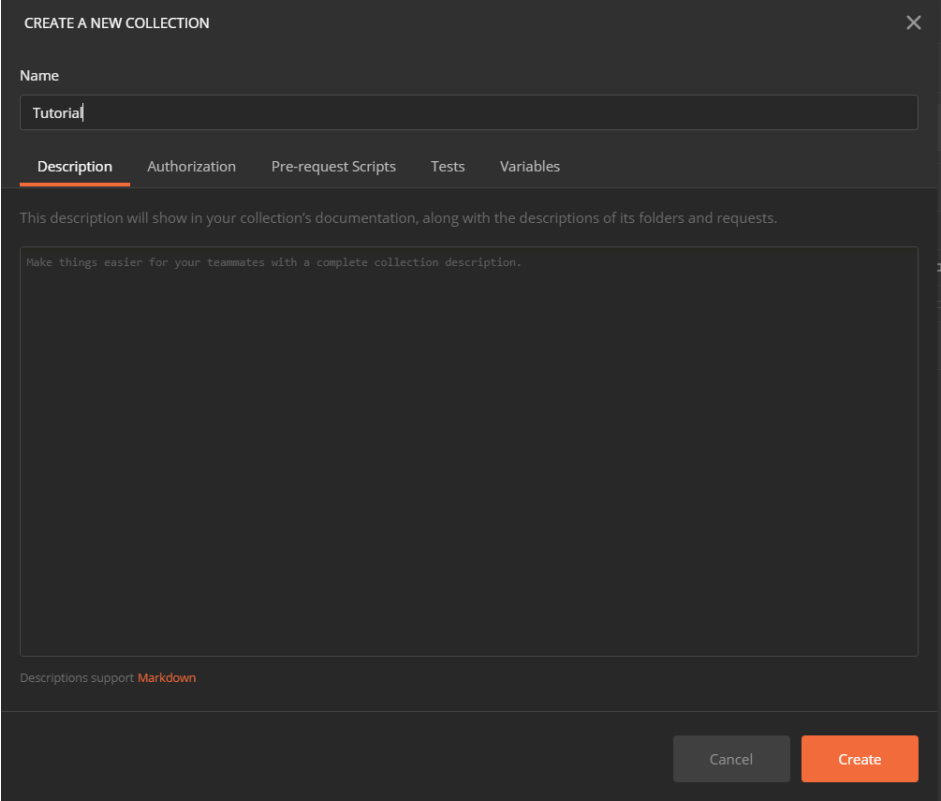
REST tutorial using Postman

- [How to integrate with Postman](#)
 - [Collection](#)
 - [Create a request](#)
 - [Authentication](#)
 - [First API Call \(version\)](#)
 - [Upload a file](#)
 - [Sending an envelope](#)
 - [Open the envelope](#)
 - [Download the signed document](#)
- [How to use variables in Postman](#)
 - [Authentication](#)
 - [Upload a file](#)
 - [Send an envelope](#)
 - [Download the file](#)
 - [Overview of all variables](#)
 - [Advantages of using variables in Postman](#)

How to integrate with Postman

Collection

In this tutorial you will do the first API call as well as sending envelopes, upload documents and download signed papers. Let us start with creating a collection so that we can save all requests there.

Figure	Steps
	Steps: 1. Go to register "new" 2. Select "Collection"

Create a request

After creating a new Collection, we will save our first request into it.

Figure	Description
--------	-------------

SAVE REQUEST

Request name

version

Request description (Optional)

Make things easier for your teammates with a complete request description.

Descriptions support Markdown

Select a collection or folder to save to:

Q Search for a collection or folder

Tutorial + Create Folder

Cancel

Save to Tutorial

Steps:

1. Go to register "new"

2. Select "request"

3. Choose a name and save the request in the collection

Authentication

The next request will show you an authorization. This is an important part of the requests for sure but not only because of security but also because we will need the authorization for all following requests.

For authentication, you'll need an api token.

v 20.42

Please see the next sample authorization (Bearer token):

Key	Value
"Authorization"	"Bearer asdfngtmvv8pfmsuaxpzz85zux3e63dd9ztrwix9mln6qka6tds83du3p3lroe"

Please see the next sample authorization (api token):

Key	Value
"ApiToken"	"asdfngtmvv8pfmsuaxpzz85zux3e63dd9ztrwix9mln6qka6tds83du3p3lroe"

You can find this information in Settings / Api Tokens and Apps. Note that all API methods require an authentication. The API token is a user specific secret which should not be shared with other users. We recommend to create different API keys for different application integrations, to avoid configuring the same key in various integration systems. This allows, e.g. in case of sharing a key by mistake, to disable one key while keeping other integrations working with their existing configuration.

The authentication with the organizationKey and the userLoginName is not recommended! It is only available for api version 1 to 5.

First API Call (version)

Let us start now with the first API call. As the name of our request tells we will first make an easy call to examine the current version.

System

GET

/v6/system/version Returns the current version of eSignAnyWhere.

⌵ 🔒

Figure	Description
--------	-------------

GEThttps://demo.esignanywhere.net/Api/v6/system/versionSend

ParamsAuthorizationHeaders (8)BodyPre-request ScriptTestsSettingsCookies

noneform-datax-www-form-urlencodedrawbinaryGraphQL

This request does not have a body

BodyCookies (2)Headers (15)Test Results

Status: 200 OKTime: 310 msSize: 802 BSave Response

PrettyRawPreviewVisualizeJSON

1"22.20.0.754"

Steps:

1. Set the request type to GET

2. Set the URI: <https://demo.esignanywhere.net/Api/v6/system/version>

3. Add the authentication (Api Token) in the header

Response

"22.20.0.754"

Upload a file



It is not allowed to upload multiple files at once (when more files are selected).

Before we can send an envelope, we must make sure that we have already uploaded a file. Therefore, the next request will be about uploading a file.

POST/v6/file/uploadUpload a file for temporary use.

Figure

POSThttps://demo.esignanywhere.net/Api/v6/file/uploadSend

ParamsAuthorizationHeaders (10)BodyPre-request ScriptTestsSettingsCookies

noneform-datax-www-form-urlencodedrawbinaryGraphQL

KEY	VALUE	DESCRIPTION
☒	Test.pdf	
Key	Value	Description

BodyCookies (2)Headers (15)Test Results

Status: 200 OKTime: 689 msSize: 835 BSave Response

PrettyRawPreviewVisualizeJSON

1{"FileId": "8a4fd938-fcbb-45f9-9866-f0fe466781d3"}

Description

Steps:

1. Set the request type to POST

2. Set the URI: <https://demo.esignanywhere.net/Api/v6/file/upload>

3. Add a file in the section "Body" form-data

4. Add the authentication (Api Token) in the header

Response

{
 "FileId": "8a4fd938-1234-1234-9866-f0fe466781d3"
}

Please note: The FileId is needed for the next api call.

Sending an envelope

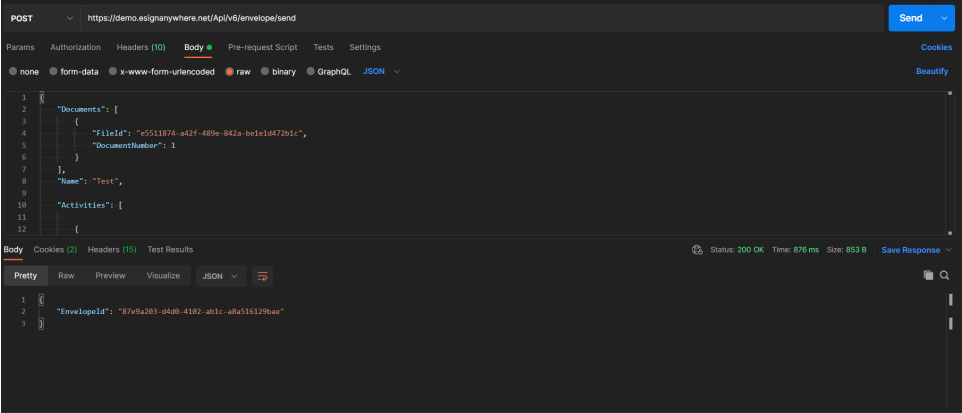
Send the first envelope with the document you have chosen from the upload call.

https://demo.esignanywhere.net/Api/swagger/ui/index#/Envelope/Envelope_Send

and paste it in the body of the sending request. Then change from “none” to “raw” and change the File ID to the one you get from the upload file request.

POST /v6/envelope/send Directly create and send an envelope.

Figure



Description

Steps:

1. Set the request type to POST
2. Set the URI: <https://demo.esignanywhere.net/Api/v6/envelope/send>
3. Add the envelope configuration in "Body" (sample configuration below)
 - a. Please change the type to "raw"
 - b. Change the "Text" to JSON format
4. Add the authentication (Api Token) in the header

Response

```
{  "EnvelopeId": "1393adcf-1234-1234-9173-aba385426549"}  
```

Please note: The EnvelopeId is needed for the next api call.

The following sample configuration contains two recipients with the following workflow:

- 1. First recipient needs to sign with a ClickToSign
- 2. Second recipient just receives a copy of the signed document

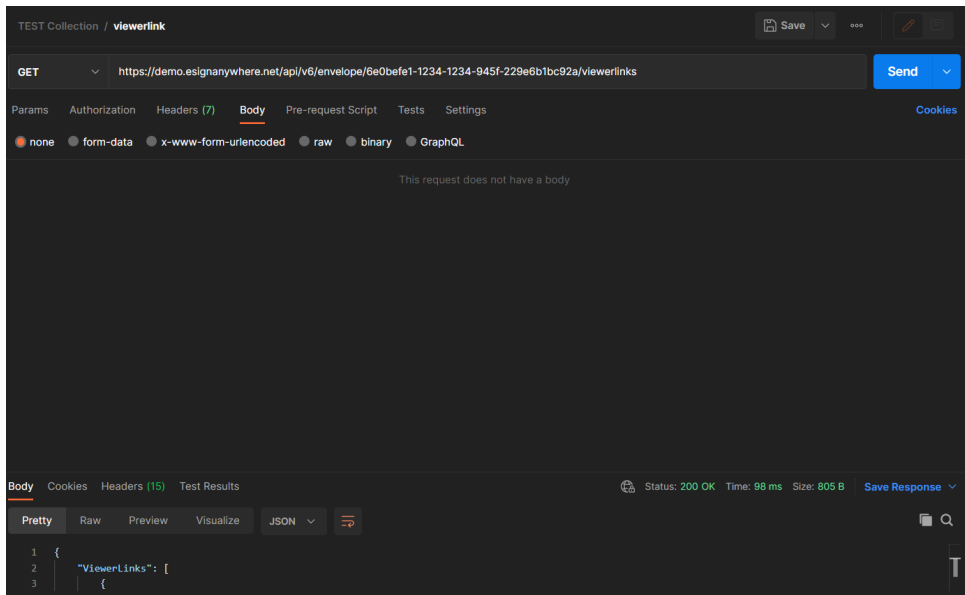
```

{
  "Documents": [{
    "FileId": "d33d43ca-1234-1234-1234-b645fc4e0fb2",
    "DocumentNumber": 1
  }
],
  "Name": "Test",
  "Activities": [{
    "Action": {
      "Sign": {
        "RecipientConfiguration": {
          "ContactInformation": {
            "Email": "jane.doe@sample.com",
            "GivenName": "Jane",
            "Surname": "Doe",
            "LanguageCode": "EN"
          }
        },
        "Elements": {
          "Signatures": [{
            "ElementId": "sample sig click2sign",
            "Required": true,
            "DocumentNumber": 1,
            "DisplayName": "Sign here",
            "AllowedSignatureTypes": {
              "ClickToSign": {
            }
          }
        },
        "FieldDefinition": {
          "Position": {
            "PageNumber": 1,
            "X": 100,
            "Y": 200
          },
          "Size": {
            "Width": 100,
            "Height": 70
          }
        }
      }
    }
  }],
  "SigningGroup": "firstSigner"
}, {
  "Action": {
    "SendCopy": {
      "RecipientConfiguration": {
        "ContactInformation": {
          "Email": "john.doe@sample.com",
          "GivenName": "John",
          "Surname": "Doe",
          "LanguageCode": "EN"
        }
      }
    }
  }
}
]
}

```

Open the envelope

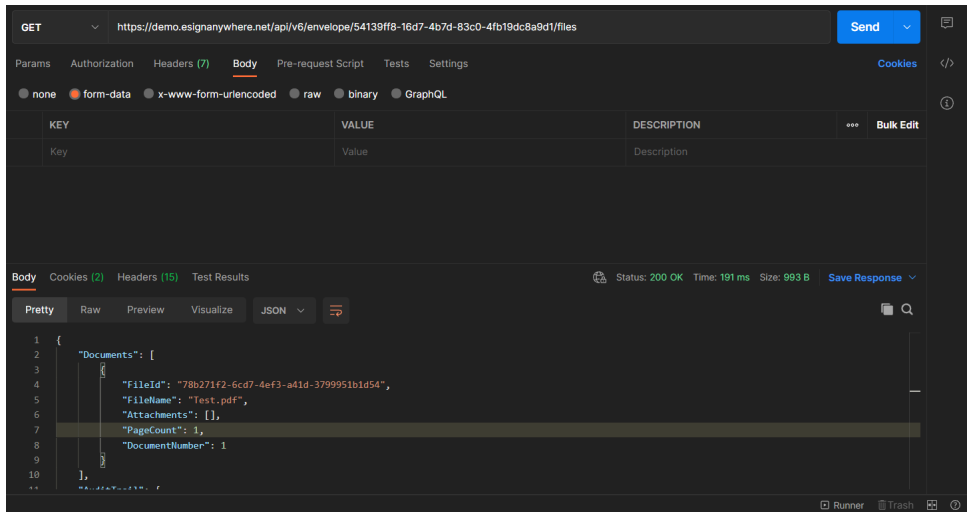
After you have sent the envelope and you have got the envelopeld you can call the method <https://demo.esignanywhere.net/api/v6/envelope/{envelopeld}/viewerlinks>. With the result of this method you get information about the envelope and you can find the `workstepRedirectionURL` (Viewerlink).

Figure	Description
	Steps: 1. Set the request type to GET 2. Set the URI: https://demo.esignanywhere.net/api/v6/envelope/{envelopeId}/viewerlinks 3. Add the authentication (Api Token) in the header Response <pre>{ "ViewerLinks": [{ "ActivityId": "5609d2ea-1234-1234-1959e63fde64", "Email": "john.doe@sample.com", "ViewerLink": "https://demo.esignanywhere.net/workstepredirector/sign?identifier=mdIkzVC1234512Igi0LD3Iodfynd~12345RbJQXusyZuC Np7FR6PniOT12345B~fQ1234A==" }]}</pre> With this redirection link it is possible to open and sign the document in your web portal.

Download the signed document

Before downloading the documents we need the file Ids. Therefore the following api call is necessary:

Figure	Description
--------	-------------



Steps:

- 1. Set the request type to GET
- 2. Set the URI: <https://demo.esignanywhere.net/Api/v6/envelope/{envelopeId}/files>
- 3. Add the authentication (Api Token) in the header

Response

```
{
  "Documents": [
    {
      "FileId":
"78b271f2-1234-1234-1234-
3799951b1d54",
      "FileName":
"Test.pdf",
      "Attachments":
[],
      "PageCount": 1,

      "DocumentNumber": 1
    }
  ],
  "AuditTrail": {
    "FileId": "7a86456c-
1234-1234-1234-
032757afa88e",
    "XmlFileId":
"dc60dcd4-1234-1234-1234-
8828c79a2c88"
  },
  "Disclaimers": []
}
```

Here you can see all three Ids for the documents which can be downloaded.

After this call just add the Ids in the following api call URI and download the documents:

GET /v6/file/{fileId} Downloads a file. Only files belonging to completed envelopes are available.

Figure	Description
--------	-------------

GEThttps://demo.esignanywhere.net/api/v6/file/78b271f2-6cd7-4ef3-a41d-3799951b1d54Send

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettings

AcceptAccept-EncodingConnectionApiToken

*/*gzip, deflate, brkeep-alivetq85ri6f6tzy7xtiozfn0j2o4pg1fnzuz2no28ugfxmbmgqw...

Key

ValueDescription

BodyCookies (2)Headers (14)Test ResultsStatus: 200 OKTime: 494 msSize: 106.28 KBSave Response

Test

Steps:

1. Set the request type to GET

2. Set the URI: <https://demo.esignanywhere.net/api/v6/file/{fileId}>

3. Add the authentication (Api Token) in the header

Response

Document to download

How to use variables in Postman

Let us start with the authentication but this time we will use variables, so we do not have to remember the api token for all other requests.

Manage environments

The image shows the Postman application interface. At the top, there's a request bar with a GET method and a URL. Below it, the 'Headers' tab is active, showing several headers like 'Accept', 'Accept-Encoding', 'Connection', and 'ApiToken'. The 'Body' tab is also visible. On the right side, there's a 'Steps' section with three numbered steps: 1. Set the request type to GET, 2. Set the URI to a specific URL, and 3. Add the authentication (Api Token) in the header. Below the steps is a 'Response' section with a 'Document to download' link. At the bottom of the interface, there's a 'Manage environments' section with a 'No Environment' dropdown and a gear icon for configuration.

After selecting the configuration button following window should appear:

There you must fill in a name for the environment and we have to create one variable which is necessary for the authentication. Further details in the next Figure:

MANAGE ENVIRONMENTS

×

Add Environment

api token environments

	VARIABLE	INITIAL VALUE ⓘ	CURRENT VALUE ⓘ	...	Persist All	Reset All
☒	ApiToken	60ujx3hgqsd0ag41bl8...	60ujx3hgqsd0ag41bl8z12345o1kke5tabcd30se123450mq...			
	Add a new variable					

ⓘ

Use variables to reuse values in different places. Work with the current value of a variable to prevent sharing sensitive values with your team. [Learn more about variable values](#)

×

Cancel

Add

First give your variable a name (it does not matter how you name them). Then fill in "initial value" and "current value" your api token. After filling up the environments for the variable, we now can use it in the header of the next request.

Authentication

The next figure shows you the same request we have already done in the tutorial above but this time we will use the variables we have just created.

GET

https://demo.esignanywhere.net/Api/v6.0/authorization/whoami

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Headers

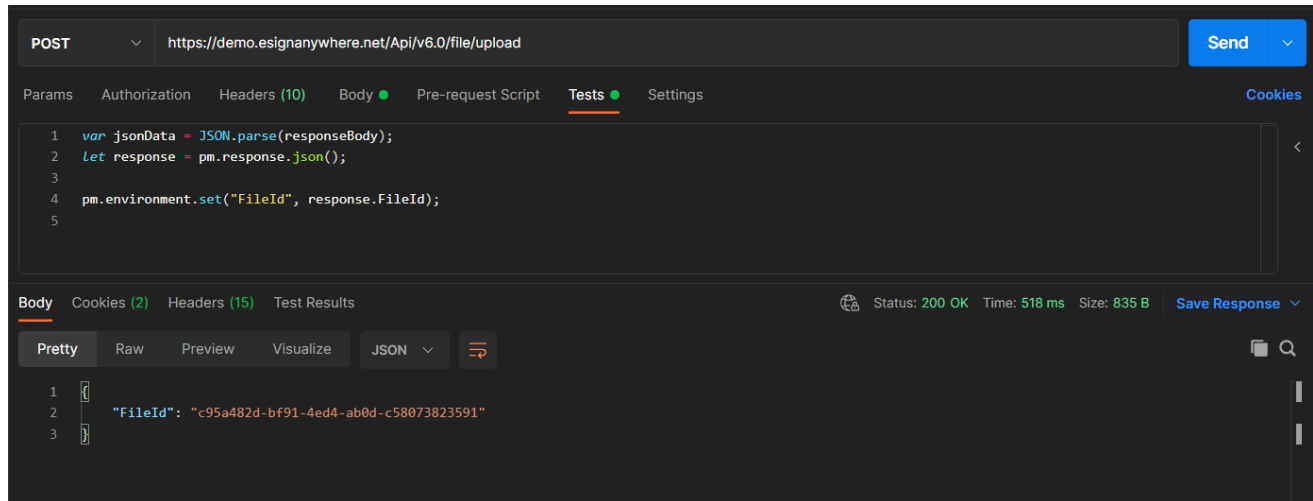
Hide auto-generated headers

	KEY	VALUE	DESCRIPTION	...	Bulk Edit	Presets
☒	Cookie ⓘ	TS0104e0e5=01c709a1afc853bf9feed3eb1f0d34887825...				
☒	Postman-Token ⓘ	<calculated when request is sent>				
☒	Host ⓘ	<calculated when request is sent>				
☒	User-Agent ⓘ	PostmanRuntime/7.29.0				
☒	Accept ⓘ	*/*				
☒	Accept-Encoding ⓘ	gzip, deflate, br				
☒	Connection ⓘ	keep-alive				
☒	ApiToken	{{ApiToken}}				
	Key	Value	Description			

In the first column we have our api token. In the second column we now do not have to fill in our api token, but we must write `{{name_of_the_variable}}`. From now on you can always use the variable instead of continual writing your api token because all the dates are saved in the environments. If you run this request, you should get the status "200 OK".

Upload a file

Now we will upload a file and save the file ID which we will get from the request in another variable in the environments.



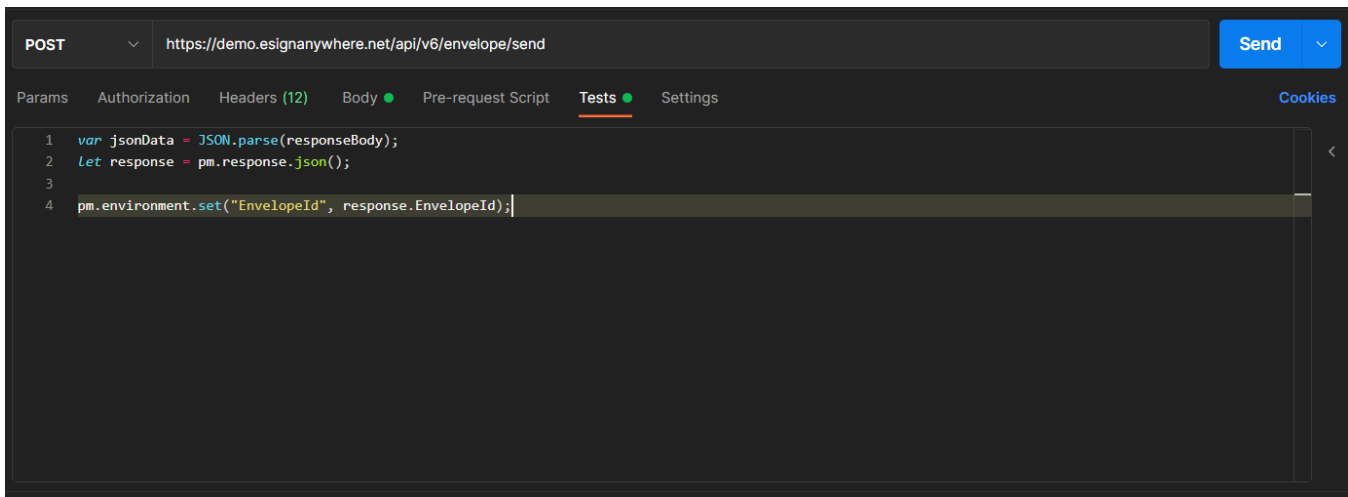
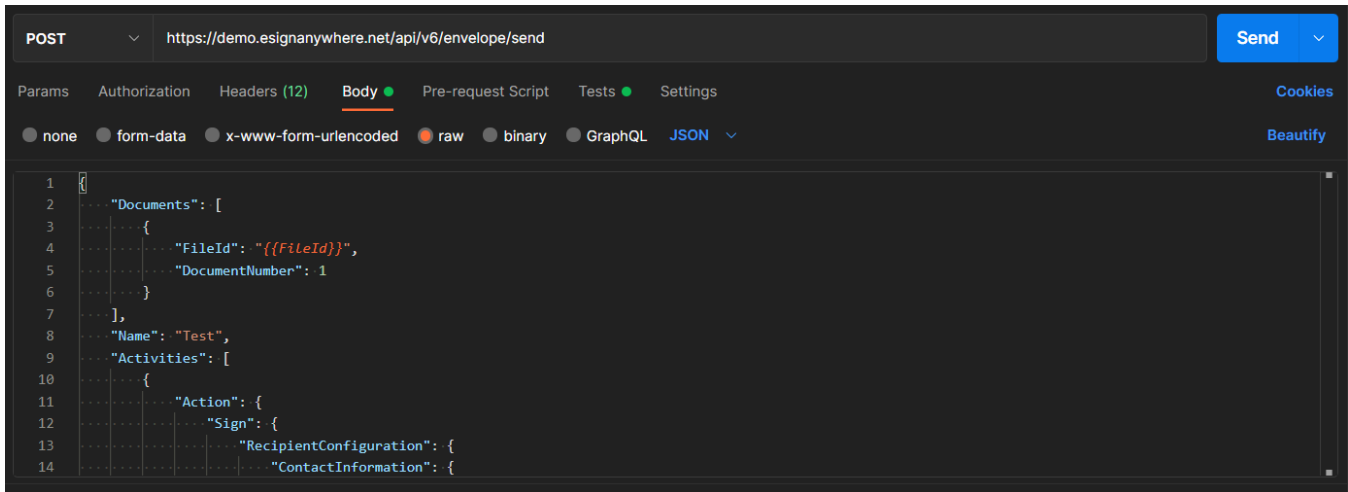
Just these three lines of code are necessary to save the file ID in another variable. First we will parse the response, then we set the file ID with the command ("pm.environment.set("name_of_the_variable", response." name_of_the_response").

After running this request, the new variable "FileId" should be in the environments. See next Figure:

Variables			Edit
VARIABLE	INITIAL VALUE	CURRENT VALUE	
ApiToken			
FileId		bb4ad3a0-7218-4443-a603-c418d8b5b847	

Send an envelope

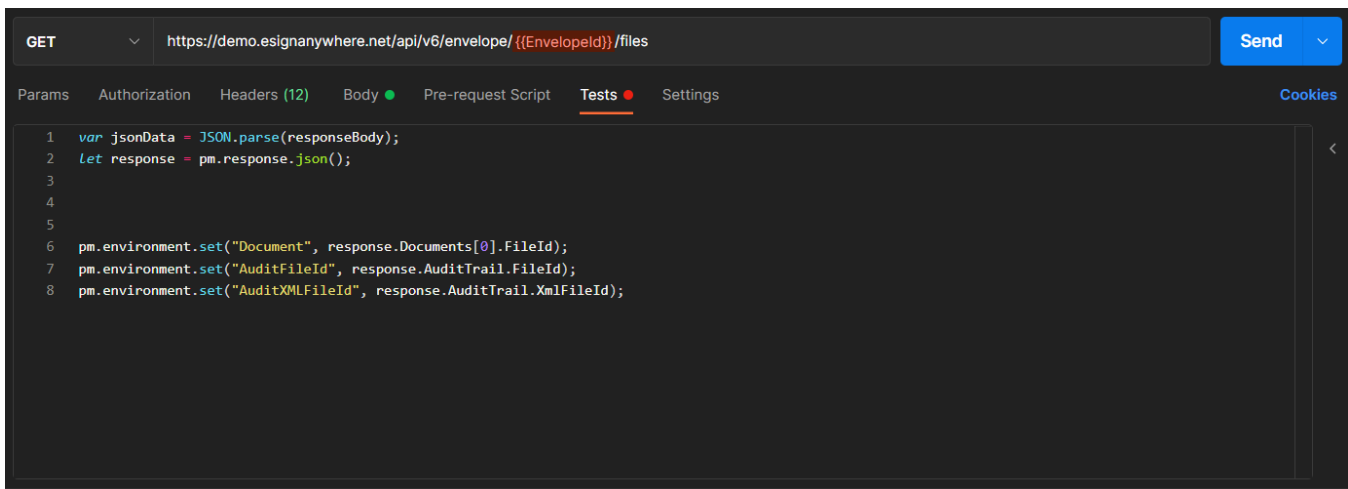
After saving the file id we can now send the envelope with an integration of the variable. Therefore we just fill in {{ name_of_the_variable_for_the_fileID}} instead of copy paste the file ID. This is shown in the next Figure. In the Figure 17 there is another example of how you save a date from the response. In this case we will save the envelope ID.



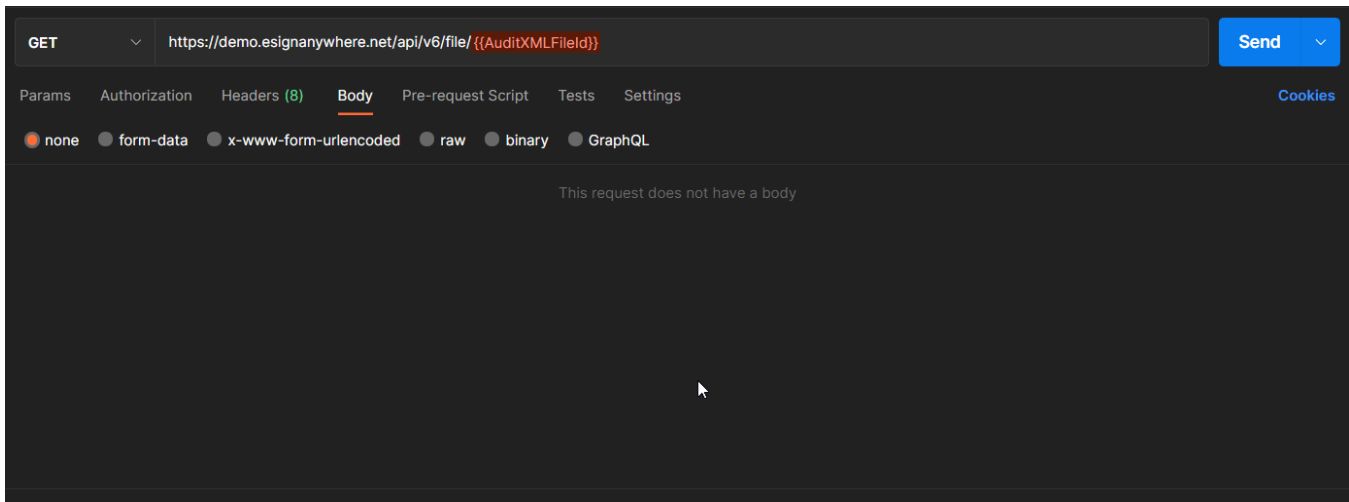
Download the file

First we get the File Ids and save them in our environment:

Please note: The document must be finished! Otherwise it is not possible to retrieve the document Ids.



After we have saved all document Ids we can use them for the download request.



Overview of all variables

Variables

Fork

Save

Share

	VARIABLE	TYPE ⓘ	INITIAL VALUE ⓘ	CURRENT VALUE ⓘ		Persist All	Reset All
☒	ApiToken	secret					
☒	FileId	default		873c1852-1234-1234-b488-3d29da8e8f81			
☒	EnvelopId	default		9d1e4022-1234-1234-1234-62237dc5e5db			
☒	Document	default		62ad466f-1234-1234-1234-989c6197a890			
☒	AuditFileId	default		2d3719c1-1234-1234-1234-2e74d3366d04			
☒	AuditXMLFileId	default		2d0f1580-1234-1234-1234-2bc017b6c5ba			
	Add a new variable						

Advantages of using variables in Postman

First, we do not have to remember all dates which we get through all the requests. Furthermore, we can simply change the organization key and the username and wen can play all requests without other changes because the variables auto fill in the new dates in the requests. Moreover we have a better overview of all ID`s so that we can easily match them with the requirements of the requests.