



Headless Chrome Docker Container

Last Updated: August 3rd, 2019

[Intro](#)

[Functionality](#)

[Experimenting & Testing](#)

[HTML to PDF](#)

[Request](#)

[Response](#)

[Sample Python Code](#)

[URL to PDF](#)

[Request](#)

[Response](#)

[Sample Python Code](#)

[HTML to Image](#)

[Request](#)

[Response](#)

[Sample Python Code](#)

[URL to Image](#)

[Request](#)

[Response](#)

[Sample Python Code](#)

[Advanced Options](#)

[Upload to Third Party \(such as Amazon S3\)](#)

[Sample Python Code](#)

[Fire and Forget Asynchronous Calls](#)

[Custom HTTP Headers for URL Requests](#)

[Puppeteer WaitFor Options](#)

[Hosting Options](#)

Intro

This document outlines the use of the Api2Pdf Headless Chrome Docker Container. The container supports much of the Puppeteer functionality for generating PDFs and Screenshots.

Any questions, inquiries, or bug reports can be emailed to support@api2pdf.com.

Functionality

Endpoints

- HTML to PDF
- URL to PDF
- HTML to Screenshot / Image
- URL to Screenshot / Image

Control over file generation

- Generate file and download immediately
- Generate file and upload to third party (such as Amazon S3)
- Generate file asynchronously (fire and forget) and upload to third party. Useful for long running requests.

Advanced Options

Puppeteer supports many advanced options for PDF and screenshot generation. The request form that lives at the base route <https://a2p-v2-demo.azurewebsites.net/> is designed to help you understand what is available for constructing the JSON payloads. It does not actually generate any PDFs or screenshots.

Bells & Whistles

- Specify custom HTTP headers on URL requests
- Use Puppeteer WaitFor options

Experimenting & Testing

Testing and experimentation can be done on this base url:

<https://a2p-v2-demo.azurewebsites.net>

Resulting files will have an API2PDF watermark on them. The server is 1.75 gb memory and is shared among all testers and cannot be used for production. Please be courteous.

Advanced Options

Puppeteer supports many advanced options for PDF and screenshot generation. The request form that lives at the base route <https://a2p-v2-demo.azurewebsites.net/> is designed to help you understand what is available for constructing the JSON payloads. It does not actually generate any PDFs or screenshots.

HTML to PDF

Request

METHOD

POST

URL

/pdf/html?format=pdf

HEADERS

Content-type: application/json

Authorization: your-api-key

MINIMUM REQUIRED BODY

```
{  
  'RequestId': '12345',  
  'Html': '<p>Hello World</p>'  
}
```

ADVANCED PDF OPTIONS BODY (<https://a2p-v2-demo.azurewebsites.net/>)

```
{  
  'RequestId': '12345',  
  'Html': '<p>Hello World</p>',  
  "UsePrintCss": true,  
  "PdfOptions": {  
    "Scale": 1.0,  
    "DisplayHeaderFooter": false,  
    "HeaderTemplate": "<span></span>",  
    "FooterTemplate": "<span></span>",  
    "PrintBackground": true,  
    "Landscape": false,  
    "Width": "8.27in",  
    "Height": "11.69in",  
    "MarginOptions": {  
      "Top": ".4in",
```

```
    "Left": ".4in",
    "Bottom": ".4in",
    "Right": ".4in"
  }
}
```

Response

Binary PDF file

Sample Python Code

<https://gist.githubusercontent.com/apexdodge/31041b1d6b7cd6c02ea2784cee47cb32/raw/96090760336f8262118204fbc58aacb776a324a3/gistfile1.txt>

URL to PDF

Request

METHOD

POST

URL

/pdf/url?format=pdf

HEADERS

Content-type: application/json

Authorization: your-api-key

MINIMUM REQUIRED BODY

```
{
  'RequestId': '12345',
  'Url': 'https://www.api2pdf.com'
}
```

ADVANCED PDF OPTIONS BODY (<https://a2p-v2-demo.azurewebsites.net/>)

```
{
  'RequestId': '12345',
  'Url': 'https://www.api2pdf.com',
  "UsePrintCss": true,
  "PdfOptions": {
    "Scale": 1.0,
```

```
"DisplayHeaderFooter": false,
"HeaderTemplate": "<span></span>",
"FooterTemplate": "<span></span>",
"PrintBackground": true,
"Landscape": false,
"Width": "8.27in",
"Height": "11.69in",
"MarginOptions": {
  "Top": ".4in",
  "Left": ".4in",
  "Bottom": ".4in",
  "Right": ".4in"
}
}
```

Response

Binary PDF file

Sample Python Code

<https://gist.githubusercontent.com/apexdodge/002e6478a375249a2d14a1ec9a679ba8/raw/04558fe77648aa12b94644d9468652cb81f431c1/gistfile1.txt>

HTML to Image

Request

METHOD

POST

URL

/screenshot/html?format=png

HEADERS

Content-type: application/json

Authorization: your-api-key

MINIMUM REQUIRED BODY

```
{
  'RequestId': '12345',
  'Html': '<p>Hello World</p>'
}
```

```
}
```

ADVANCED SCREENSHOT OPTIONS BODY (<https://a2p-v2-demo.azurewebsites.net/>)

```
{  
  'RequestId': '12345',  
  'Html': '<p>Hello World</p>',  
  "ScreenshotOptions": {  
    "FullPage": true,  
    "OmitBackground": false  
  }  
}
```

Response

Binary PDF file

Sample Python Code

<https://gist.githubusercontent.com/apexdodge/c0b69876709e11dc560a9d09c2eef1e4/raw/d01cf3735632aecbd83a91fa4c9f588f2cb94986/gistfile1.txt>

URL to Image

Request

METHOD

POST

URL

/screenshot/url?format=png

HEADERS

Content-type: application/json

Authorization: your-api-key

MINIMUM REQUIRED BODY

```
{  
  'RequestId': '12345',  
  'Url': 'https://www.api2pdf.com'  
}
```

ADVANCED SCREENSHOT OPTIONS BODY (<https://a2p-v2-demo.azurewebsites.net/>)

```
{
```

```
{
  'RequestId': '12345',
  'Url': 'https://www.api2pdf.com'
  "ScreenshotOptions": {
    "FullPage": true,
    "OmitBackground": false
  }
}
```

Response

Binary PDF file

Sample Python Code

<https://gist.githubusercontent.com/apexdodge/01bb12e7754d1ba7a0078316b13500de/raw/1b1314ce85d6ed7ea0b8766e6ec3444e2c6d2df3/gistfile1.txt>

Advanced Options

Upload to Third Party (such as Amazon S3)

Included is the ability to upload the resulting file directly to an endpoint that you specify. It is common to have the file end up in a file storage provider like Amazon S3.

1. Remove the query parameter ?format=pdf from the endpoint urls to enable this functionality.
2. Add the following options to the JSON payload.

```
{
  "StorageOptions": {
    "Method": "PUT",
    "Url": "https://your-url-to-put-the-file",
    "ExtraHTTPHeaders": { }
  }
}
```

After successful file generation, the container will attempt a PUT or POST depending on what you provided in the options (defaults to PUT) at the Url you specified.

To accomplish this with Amazon S3 specifically, Amazon allows you to [generate presigned urls](#).

Sample Python Code

Uses the boto3 python library to communicate with Amazon Web Services.

<https://gist.githubusercontent.com/apexdodge/aea6a6e4ecd86a23b6ebb2a4a007bac5/raw/e1169061b8efb46a68e3a15480b2c1649f8c5022/gistfile1.txt>

Fire and Forget Asynchronous Calls

If a request is going to take a long time to generate the PDF or image files then it may be necessary to run the request in a background task. You can make a call to the API to fire off the request and specify a notification url that will be called when the file or request has been completed.

Note that this approach requires Upload to Third Party Storage option enabled from the section above.

1. Add /async to the end of whichever route you want to run asynchronously. For example /pdf/html will become /pdf/html/async.
2. Add the following options to the JSON payload.

```
"StorageOptions": {  
  "Method": "PUT",  
  "Url": "https://your-url-to-put-the-file",  
  "ExtraHTTPHeaders": { }  
},  
"WebhookOptions": {  
  "Url": "https://your-url-to-post-notification",  
  "ExtraHTTPHeaders": { }  
}
```

Custom HTTP Headers for URL Requests

For both URL to PDF and URL to Image requests, we support the inclusion of extra HTTP headers. This is helpful if you need to access a url that is behind some basic security or requires some kind of custom defined http header.

Add the following to the JSON payload:

```
"ExtraHTTPHeaders": {  
  "HeaderName1": "HeaderValue1",  
  "HeaderName2": "HeaderValue2"  
}
```

Puppeteer WaitFor Options

For those who need to generate fancy looking PDFs that require javascript and graphs to be rendered may run into issues with PDFs and Screenshots. The PDF and Screenshots fire before the javascript has a chance to complete its rendering process.

We now support the Puppeteer WaitFor options. Add the following to the JSON Payload:

```
"WaitForOptions": {  
  "Method": "WaitForExpression",  
  "Value": "window.IsPageLoaded"  
}
```

That is one example, but we support all below:

- WaitForNavigation (domcontentloaded, load, networkidle0, networkidle2)
- WaitForExpression
- WaitForFunction
- WaitForSelector

The documentation is extensive and not covered here. See usage details on Puppeteer's documentation:
<https://github.com/GoogleChrome/puppeteer/blob/master/docs/api.md#framewaitfornavigationoptions>

Hosting Options

If you would like to discuss hosting options and pricing for the docker container, please contact support@api2pdf.com for details.

Managed Hosting on Microsoft Azure

API2PDF can host the docker container for you on Microsoft Azure on an isolated virtual machine.

Enterprise

The docker container is for sale for enterprise customers who need to run the container in their own environment.

SaaS

We are not currently offering SaaS version of this web service.