HEPLA GmbH

Reseller Handbook

Part D: API & Configurator iframe

Product data · Stock levels · Preview rendering · iframe configurator

As of: May 18, 2026

This guide describes how you, as a reseller, can integrate the HEPLA API and the iframe configurator into your own website or online shop. It is part of a package consisting of this PDF and two demo files (logify_remote.py and iframe_demo.html), which serve as working templates.

1 Overview

The HEPLA interface consists of two independent parts, which you can use individually or combined.

Component	Purpose	Access via
Data API	Product data, prices, stock levels, image and clip URLs.	HTTPS calls with API token.
Render API	3D preview clips (MP4) with individually placed imprints.	HTTPS call with image upload (multipart/form-data).
Configurator iframe	Interactive configuration UI for embedding on your own product page. The end customer designs the imprint themselves.	iframe + postMessage messages.

1.1 Bundled demo files

File	Purpose
logify_remote.py	Ready-to-run Python script for calling the Render API. Generates a 3D preview clip from a product number and image file. Standalone file, can be used directly or as a template for an integration.
corners.py	Helper script for determining the corner points (corners) of a print area from a print mask PNG. Returns the 8 per-mille values that logify_remote.py and the Render API expect as corner points for logo placement.
iframe_demo.html	Standalone HTML file with a fully functional iframe integration, including postMessage receiver. Open the file in your browser, enter

	your token, done.

ℹ All three demo files are standalone and run without installation. logify_remote.py only requires Python 3.7+ and the "requests" package. corners.py requires Python 3.7+, Pillow and NumPy. iframe_demo.html requires nothing but a browser.

1.2 Prerequisites

- API token: available from HEPLA on request. The token is personal and must be treated like a password.
- HTTPS connection to www.hepla.de.
- For the Render API: image files (PNG or JPG) of your logos.
- For the iframe: a web page on which you can embed an iframe and, ideally, a small JavaScript adjustment for receiving the postMessage responses.

1.3 Base URL

All endpoints are reachable at:

```
https://www.hepla.de
```

Example of a complete API call:

```
https://www.hepla.de/api/v1/artikeldaten/?token=YOUR_TOKEN&artnr=7262-0
```

2 Authentication

All API calls are authenticated via a personal token. The token is accepted in two ways; choose whichever fits your integration best:

Variant	Example
Query parameter (simple, good for testing in the browser)	?token=YOUR_TOKEN
HTTP header (cleaner, recommended for production use)	X-Api-Token: YOUR_TOKEN

⚠ Token hygiene: treat the token like a password. Do not expose it in public repositories or frontend code. If lost, contact HEPLA – the token can be invalidated and a new one issued.

3 Data API

The Data API returns product data, prices and stock levels as JSON or CSV. All endpoints are GET calls.

3.1 Endpoint overview

Endpoint	Purpose	Format
/api/v1/artikeldaten/	Master data and prices of a product (live database).	JSON
/api/v1/artikeldaten/vorschau/	Like /artikeldaten/, but with prices from the test database	JSON

	(year-end preparation).	
/api/v1/lagerbestand/	Current stock levels. Available as JSON or CSV.	JSON or CSV
/api/v1/render/	Generate 3D preview clip with imprint (POST).	multipart/form-data → JSON response

3.2 Product data

Returns master data, description, image and clip URLs and – if you are authenticated and the product is enabled for you – also the price scales in your pricing group.

```
GET https://www.hepla.de/api/v1/artikeldaten/?token=YOUR_TOKEN&artnr=7262-0
```

Parameters:

Parameter	Required	Description
token	yes	API token (alternatively as X-Api-Token header).
artnr	yes	Product number, e.g. 7262-0.
sprache	no	Language of the texts: de (default), en, fr, es, nl, it.

Response (excerpt):

```
{
  "ok": true,
  "artnr":         "7262-0",
  "bezeichnung":   "Ballpoint pen Lyno",
  "beschreibung":  "...",
  "bild_url":      "https://www.hepla.de/media/Artikelbilder/7262-0.jpg",
  "clip_url":      "https://www.hepla.de/media/Artikelclips/7262-0.mp4",
  "preise": [
    {"ab_menge": 100, "preis": 0.89},
    {"ab_menge": 250, "preis": 0.79},
    {"ab_menge": 500, "preis": 0.71}
  ],
  "druckflaechen": [
    {"code": "T1",  "breite_mm": 35, "hoehe_mm": 7, "max_farben": 1},
    {"code": "D1",  "breite_mm": 35, "hoehe_mm": 7, "max_farben": null}
  ]
}
```

ℹ Image and clip URLs follow the scheme /media/Artikelbilder/<artnr>.jpg and /media/Artikelclips/<artnr>.mp4. You can embed these URLs directly on your website – an additional download via the API is not necessary.

3.3 Stock levels

Current stock levels for all products visible to you. Optionally also as CSV download:

```
GET https://www.hepla.de/api/v1/lagerbestand/?token=YOUR_TOKEN
GET https://www.hepla.de/api/v1/lagerbestand/?token=YOUR_TOKEN&format=csv
```

Optional filters:

Parameter	Description
artnr	Only one specific product (otherwise all).
format	csv = CSV download with column header. Otherwise JSON.

3.4 Year-end change: preview of new prices

At the end of the year, HEPLA provides the prices for the coming year in a test database before they officially come into effect. You can retrieve these prices in advance to prepare your costing, price lists or data feeds:

```
GET https://www.hepla.de/api/v1/artikeldaten/vorschau/?token=YOUR_TOKEN&artnr=7262-0
```

The response format is identical to the live endpoint; however, the prices are the new annual prices. Between years, the values often do not differ at all or only slightly; verified changes are communicated through the regular price list.

4 Render API: 3D preview clips

The Render API generates 3D preview clips (MP4) that show your product with a specifically placed logo – ideal for product detail pages, customer offers or mailings.

4.1 Demo script logify_remote.py

The easiest way to use the Render API is the bundled demo script logify_remote.py. It takes your token, a product number and an image file and writes the rendered MP4 to the target directory of your choice.

Prerequisite: Python 3.7 or newer and the requests package:

```
pip install requests
```

Simplest call:

```
python logify_remote.py 7262-0 /tmp/out D1=logo.png \
    --token=YOUR_TOKEN
```

Result: /tmp/out/ then contains 7262-0.mp4.

With corner points (logo not across the entire print area but at a specific position):

```
python logify_remote.py 7262-0 /tmp/out \
    D1[120,85,890,85,890,930,120,930]=logo.png \
    --token=YOUR_TOKEN
```

With color reduction (for screen / pad printing):

```
python logify_remote.py 7262-0 /tmp/out \
    S1_3[120,85,890,85,890,930,120,930]=logo.png \
    --token=YOUR_TOKEN
```

Multiple print areas in one call:

```
python logify_remote.py 7254-3 /tmp/out \
    D1[656,52,656,936,350,936,350,52]=logo.png \
    D2[67,572,950,572,950,866,67,866]=logo.png \
    --token=YOUR_TOKEN
```

The file logify_remote.py contains an extensive doc string at the top with all further options and examples. You can also use the script as a basis for your own integration in your language – the HTTP calls are implemented there compactly and clearly.

4.2 Determining corner points ([corners])

The corner points describe where exactly the logo should sit on the print area. They are specified as eight per-mille values (0 to 1000), relative to the print mask sketch of the product. The order is:

```
[ul_x, ul_y, ur_x, ur_y, lr_x, lr_y, ll_x, ll_y]
  ul = upper left
  ur = upper right
  lr = lower right
  ll = lower left
```

If you do not specify corner points, the logo fills the entire print area – this is the simplest and most common case. You only need your own corner points if the logo is intentionally smaller than the full print area or requires a specific position.

4.2.1 Helper script corners.py

The bundled script corners.py reads a HEPLA print mask PNG and outputs the eight corner points of the printable area in the correct format. The print mask PNGs for your products are available for download at https://www.hepla.de/media/Artikelclips/<artnr>#<druckcode>.png (e.g. 7262-0#D1.png for product 7262-0, print code D1).

⚠ URL note: the "#" in the filename is a reserved character in the URL path (fragment separator) and must be escaped as %23, otherwise curl, wget or the browser will truncate the URL at the #. Example download with curl:

curl -O "https://www.hepla.de/media/Artikelclips/7262-0%23D1.png"

The downloaded file is then named 7262-0#D1.png locally — the # in the filename on the file system is not a problem.

Prerequisites: Python 3.7 or newer and the packages Pillow and numpy:

```
pip install Pillow numpy
```

Simplest call:

```
$ python corners.py 7262-0#D1.png
120,85,890,85,890,930,120,930
```

Various output formats (via --format):

Format	Output	Usage
csv (default)	120,85,890,85,890,930,120,930	CSV lists, custom scripts.
bracket	[120,85,890,85,890,930,120,930]	Directly usable in logify_remote.py calls.
json	[120, 85, 890, 85, 890, 930, 120, 930]	JSON fields in your own REST calls.
api	"ecken": [120, 85, 890, 85, 890, 930, 120, 930]	Snippet for direct insertion into the params JSON call to the Render API.

Example of direct chaining with logify_remote.py – the corner points are determined at runtime and inserted into the call:

```
python logify_remote.py 7262-0 /tmp/out \
    D1$(python corners.py 7262-0#D1.png --format=bracket)=logo.png \
    --token=YOUR_TOKEN
```

With pixel-size diagnostics (image size + bounding box in pixels go to stderr, the main result remains clean on stdout):

```
$ python corners.py 7262-0#D1.png --mit-info
Image size: 1000 × 1000 px
Bounding:   x = 120…890, y = 85…930
120,85,890,85,890,930,120,930
```

4.2.2 Custom position within the print area

You can start from the bounding box determined by corners.py and shrink or shift the logo within it. Example: logo centered on a print area of 35 mm × 7 mm, with margins of 10 % left/right and 5 % top/bottom:

```
D1[100,50,900,50,900,950,100,950]=logo.png
```

4.3 Direct HTTP call (without demo script)

If you want to call the Render API directly from your own application instead of using logify_remote.py:

```
POST https://www.hepla.de/api/v1/render/?token=YOUR_TOKEN
Content-Type: multipart/form-data

Fields:
  params           JSON with the render parameters
  aufdruck_<CODE>  Image file per print area (PNG or JPG)

JSON format of "params":
{
```

```
  "artikelnummer": "7262-0",
  "flaechen": [
    {
      "code":          "D1",
      "datei":         "aufdruck_D1",
      "ecken":         [120,85,890,85,890,930,120,930],
      "farbreduktion": 3
    }
  ],
  "masken": false
}
```

Response (JSON, Base64-encoded files):

```
{
  "ok":      true,
  "mp4":     "<Base64-encoded MP4>",
  "masken":  { "D1": "<Base64-PNG>" },   // only if masken=true
  "log":     "...",
  "warnungen": []
}
```

i *Typical processing time is 5–10 seconds per clip. In your implementation, set a generous timeout (e.g. 120 seconds) to absorb load spikes.*

5 iframe configurator

The configurator iframe is a ready-made web UI that the end customer uses directly on your website: upload logo, position it on the print area, adjust colors, check the preview. At the end, the iframe passes an order code to your page, with which you can retrieve the configuration later.

5.1 Demo file iframe_demo.html

The bundled iframe_demo.html is a standalone HTML file with everything you need to get started:

- iframe embedding with the correct URL parameters.
- JavaScript receiver for the postMessage responses.
- Example display of the order code and preview URL after submission.

Quick start:

- Open iframe_demo.html in an editor.
- Enter your API token (HAENDLER_TOKEN) at the top of the script.
- Open the file in a browser.
- Configure, click "Apply configuration" – the order code is shown below the iframe.

The file serves as a template for your own integration; it is deliberately kept small and commented.

5.2 iframe URL

The iframe is embedded with the following URL:

```
https://www.hepla.de/konfigurator/?token=YOUR_TOKEN&artikel=7262-0
```

URL parameters:

Parameter	Required	Description
token	yes	Your API token.
artikel	yes	Product number, e.g. 7262-0.
sprache	no	Display language: de (default), en, fr, es, nl, it.
ref	no	Free reseller reference (e.g. your internal order number). Passed back unchanged in the later postMessage response.
code	no	Existing order code for reloading a previous configuration. The iframe loads the stored configuration on startup and makes it available for editing.
artikelname	no	Your display name for the product if you do not want to use the standard name.
logos	no	Comma-separated list of URLs to logos of your customers, which are to be preloaded in the gallery on startup.
theme	no	Color scheme. The default uses the HEPLA look; contact us for a reseller-specific variant.

5.3 Embedding the iframe

Minimal HTML code for embedding:

```
<iframe
    src="https://www.hepla.de/konfigurator/?token=YOUR_TOKEN&artikel=7262-0"
    width="100%" height="800"
    style="border:0">
</iframe>
```

i Recommended minimum size: 800 px height, at least 600 px width. With a smaller viewport, the print mask preview and the tool panel are displayed too narrowly. On mobile devices, the configurator automatically adapts responsively.

5.4 Communication via postMessage

The iframe sends two types of messages to the embedding page. You can react to them in your JavaScript, e.g. to add the order code to your shopping cart.

5.4.1 konfigurator_ready

Sent once when the configurator is fully loaded:

```
{
  type:          'konfigurator_ready',
  artikelnummer: '7262-0'
}
```

You can use this message, for example, to hide a loading indicator.

5.4.2 konfigurator_result

Sent when the end customer clicks "Apply configuration" in the iframe:

```
{
  type:             'konfigurator_result',
  code:             '4A7B2X',                          // order code
  gueltig_bis:      '2026-06-13T14:32:00+02:00',      // ISO 8601
  artikelnummer:    '7262-0',
  haendler_ref:     '...',                             // like URL param 'ref'
  druckflaechen:    [ /* see below */ ],
  vorschau_clip_url: 'https://www.hepla.de/.../...mp4' // optional
}
```

The druckflaechen array contains one entry per configured area with:

Field	Description
code	Print code (e.g. "D1").
farben	Number of colors used (for screen / pad printing).
nachreichen	true if the customer has specified that the image will be sent in by email afterwards instead of being uploaded now.
ecken	Corner points [ul_x, ul_y, ur_x, ur_y, lr_x, lr_y, ll_x, ll_y] in per mille (0–1000).
logo_name	File name of the chosen logo.

Example receiver in JavaScript:

```
window.addEventListener('message', (event) => {
  if (event.origin !== 'https://www.hepla.de') return;
  const data = event.data;
  if (data.type === 'konfigurator_ready') {
    console.log('Configurator ready.');
  } else if (data.type === 'konfigurator_result') {
    console.log('Order code:', data.code);
    console.log('Valid until:', data.gueltig_bis);
    console.log('Preview URL:', data.vorschau_clip_url);
    // → Add to cart, navigate to next page, ...
  }
});
```

⚠ Security recommendation: always check event.origin === "https://www.hepla.de" in your postMessage receiver before using the data. This ensures that the message actually comes from the configurator and not from a third-party page.

5.5 Order code and validity

The order code is a 6-character alphanumeric identifier that HEPLA uses to uniquely associate the configuration designed by the end customer. Pass this code to the HEPLA office together with your order; we can then access the configuration without you having to forward the print files yourself.

- Format: 6 uppercase letters and digits (example: 4A7B2X).
- Validity: 30 days from submission of the configuration. The exact expiry date is in the field gueltig_bis.
- Re-editing: as long as the code is valid, you can pass it again as a URL parameter ?code=... to the iframe. The end customer then sees their previous configuration and can adjust it.

6 Order workflow

Recommended workflow for an order via the interface:

Step	On your side	On the HEPLA side
1	You embed the iframe configurator on your product page.	–
2	End customer designs logo, placement and colors in the iframe and sees the 3D preview.	Stores the configuration internally.
3	End customer clicks "Apply configuration".	Generates 6-character order code, sends postMessage back.
4	You store the code in your order system (together with quantity, delivery address and your own order number).	–
5	You send your order to HEPLA – by email to the office, with order code, quantity and optionally haendler_ref as a reference.	–
6	–	The office reads the configuration from the code, produces the order, sends order confirmation.

7 Errors and response formats

On success, the response always contains the field "ok": true. On errors:

```
{ "ok": false, "fehler": "description text" }
```

Typical errors:

HTTP	Error	Meaning
401	No API token provided.	token parameter and X-Api-

		Token header both empty.
401	Invalid or blocked API token.	Token not (or no longer) valid. Contact HEPLA.
400	Product not found.	Check product number.
400	Product not enabled for API.	Not every product is intended for the iframe configurator. Ask HEPLA.
500	Render error.	Rare, usually due to invalid corner points or a damaged image file. The demo script provides more detailed diagnostics in the server log.

8 Support and contact

For questions about integration, new token requests, or in case of problems:

- HEPLA GmbH, www.hepla.de
- Email: it@hepla.de (token management and API questions).
- Email: vertrieb@hepla.de (orders, product approvals).

For API questions, please provide if possible: your reseller ID, the product number and, where applicable, the exact call or error message – this allows us to help you faster.