EcomLabs

STEP-BY-STEP GUIDE

Product structured data

A Product JSON-LD template ready to adapt, how to print it from a Shopify theme and a checklist for validating it with the Rich Results Test and Search Console before you wait for results.

FOR	Shopify stores or any platform
TESTED WITH	Node 22 (JSON.parse) · Shopify CLI 3.88.1 (theme check)
UPDATED	October 30, 2026
DOCS	developers.google.com/search/docs/appearance/structured-data/merchant-listing
ARTICLE	ecomlabs.dev/g/seo-en
AUTHOR	Luis Morales · @luismoralesdev

Contents

Step 01	See what your store already prints
Step 02	The Product JSON-LD template
Step 03	Print it from Shopify
Step 04	Validate with the Rich Results Test
Step 05	Publish and watch Search Console
Final	Validation checklist

STEP 01

See what your store already prints

You need access to your theme code, a published product page and Search Console. Start with what is already there:

1. Open a product page and choose View source.
2. Search for `application/ld+json`.
3. Count the blocks with `"@type": "Product"` or `"ProductGroup"`.

WHAT YOU FIND	WHAT TO DO
One block with name, image and offer	Go to step 04 and validate it.
None	Use the filter or the snippet in step 03.
Two blocks (theme and app)	Compare price and currency. Keep one.

STEP 02

The Product JSON-LD template

This template has the required merchant listing properties and the most useful recommended ones. The data is fictional and illustrative; we parsed it with `JSON.parse` and checked it against Google's documentation.

```
<script type="application/ld+json">
{
  "@context": "https://schema.org",
  "@type": "Product",
  "name": "Handmade ceramic mug, 12 oz",
  "description": "Hand-glazed stoneware mug. Dishwasher and microwave safe.",
  "image": [
    "https://example.com/photos/1x1/blue-mug.jpg",
    "https://example.com/photos/4x3/blue-mug.jpg",
    "https://example.com/photos/16x9/blue-mug.jpg"
  ],
  "sku": "MUG-BLUE-12",
  "brand": { "@type": "Brand", "name": "Example Ceramics" },
  "aggregateRating": {
    "@type": "AggregateRating",
    "ratingValue": 4.6,
    "reviewCount": 38
  },
  "offers": {
    "@type": "Offer",
    "url": "https://example.com/products/blue-mug-12oz",
    "price": 24.00,
    "priceCurrency": "USD",
```

```
    "itemCondition": "https://schema.org/NewCondition",
    "availability": "https://schema.org/InStock",
    "shippingDetails": {
      "@type": "OfferShippingDetails",
      "shippingRate": { "@type": "MonetaryAmount", "value": 4.99, "currency": "USD" },
      "shippingDestination": { "@type": "DefinedRegion", "addressCountry": "US" },
      "deliveryTime": {
        "@type": "ShippingDeliveryTime",
        "handlingTime": { "@type": "QuantitativeValue", "minValue": 0, "maxValue": 1, "unitCode": "DAY" },
        "transitTime": { "@type": "QuantitativeValue", "minValue": 2, "maxValue": 5, "unitCode": "DAY" }
      }
    },
    "hasMerchantReturnPolicy": {
      "@type": "MerchantReturnPolicy",
      "applicableCountry": "US",
      "returnPolicyCategory": "https://schema.org/MerchantReturnFiniteReturnWindow",
      "merchantReturnDays": 30,
      "returnMethod": "https://schema.org/ReturnByMail",
      "returnFees": "https://schema.org/FreeReturn"
    }
  }
}
</script>
```

FIELD	REPLACE IT WITH
`name`, `description`	The page's own, matching the H1 and the visible copy.
`image`	Public URLs of your photos. Ideally several, in 1x1, 4x3 and 16x9.
`price`	A number with a point, no symbol, above zero.
`priceCurrency`	An ISO 4217 code: `USD`, `CAD`, `EUR`.
`availability`	`InStock`, `OutOfStock`, `PreOrder` or another schema.org value.
`aggregateRating`	Your real reviews; delete it if they are not on the page.
`shippingDetails`, `hasMerchantReturnPolicy`	Your policies, or delete them if you set them in Merchant Center or Search Console.

STEP 03

Print it from Shopify

The short route is the `structured_data` filter, which Dawn uses: it returns a `Product` or a `ProductGroup` with name, image, brand and offer.

```
<script type="application/ld+json">
  {{ product | structured_data }}
</script>
```

If you need control over every property, use this snippet (`snippets/product-json-ld.liquid`) instead, not as well:

```
{% doc %}
  Prints Product JSON-LD for the selected or first available variant.
  @param {product} product - The product on the page
  @example
  {% render 'product-json-ld', product: product %}
{% enddoc %}
{%- assign variant = product.selected_or_first_available_variant -%}
<script type="application/ld+json">
  {
    "@context": "https://schema.org",
    "@type": "Product",
    "name": {{ product.title | json }},
    "description": {{ product.description | strip_html | strip | json }},
    {%- if product.featured_image %}
    "image": {{ product.featured_image | image_url: width: 1200 | prepend: 'https:' | json }},
    {%- endif %}
    {%- if variant.sku != blank %}
    "sku": {{ variant.sku | json }},
    {%- endif %}
    "brand": { "@type": "Brand", "name": {{ product.vendor | json }} },
    "offers": {
      "@type": "Offer",
      "url": {{ variant.url | prepend: request.origin | json }},
      "price": {{ variant.price | divided_by: 100.0 }},
      "priceCurrency": {{ cart.currency.iso_code | json }},
      "itemCondition": "https://schema.org/NewCondition",
      "availability": "https://schema.org/{% if variant.available %}InStock{% else %}OutOfStock{% endif %}"
    }
  }
</script>
```

> **CAUTION**
> Do not use `money_without_currency` for the `price`: with a decimal-comma currency format the value becomes invalid. And write every string with `| json`: a title with quotes breaks the JSON if you wrap it in quotes by hand.

The snippet passed `shopify theme check` in a minimal theme and its JSON parses with and without SKU, image and stock. It was not tested on a live store: try it on a development theme.

STEP 04

Validate with the Rich Results Test

1. Go to `search.google.com/test/rich-results`.
2. Test a product's public URL or, before publishing, choose Code and paste the block.
3. Leave the user agent on smartphone, the default.
4. Open the product items detected: valid, valid with warnings or invalid.
5. Fix every critical error and test again. Warnings improve your data but are not required for eligibility.

STEP 05

Publish and watch Search Console

1. Publish a few pages and check them with URL Inspection: nothing blocking them through `robots.txt`, a `noindex` or a login.
2. In Search Console, under Shopping, open the Product snippets and Merchant listings reports.
3. If an issue shows up, fix it on every page and click Validate fix. Google checks a sample first; it typically takes up to about two weeks and notifies you when it ends.

> **CAUTION**
> Passing the test does not guarantee the rich result: Google decides whether to show it. And do not mark up anything that is not on the page; marking up reviews or prices visitors cannot see can end in a manual action.

FINAL

Validation checklist

- [ ] Each product page has exactly one `Product` or `ProductGroup` block.
- [ ] It has `name`, `image` and `offers` with a `price` above zero and a `priceCurrency`.
- [ ] The price is a number with a point and matches the visible one.
- [ ] `availability` matches the variant's real stock.
- [ ] Marked-up reviews are visible on the page.
- [ ] Shipping and returns are in the markup or in Merchant Center or Search Console.
- [ ] The Rich Results Test shows no critical errors.
- [ ] You checked the Product snippets and Merchant listings reports.