Grapharch contributors

Practical code examples

Grapharch Documentation Preview 31 October 2026

This version:
https://ekgf.github.io/grapharch/showcases/artifacts/code-highlighting/specification/

Latest published version:
https://ekgf.github.io/grapharch/showcases/artifacts/code-highlighting/specification/

Grapharch contributors.

Document license

Contents

Abstract . . . 3
Status of This Document . . . 3
1 Code examples . . . 4
1.1 Python . . . 4
1.2 Bash . . . 4
1.3 JavaScript and Node.js . . . 4
1.4 TypeScript . . . 4
1.5 JSON . . . 5
1.6 JSON-LD . . . 5
1.7 SPARQL . . . 5
1.8 Turtle . . . 5
A Document provenance . . . 7

Abstract

Python, shell, JavaScript, TypeScript, JSON, JSON-LD, Turtle and SPARQL examples share the same syntax colours in HTML and PDF. Code is displayed as source and is never executed.

Status of This Document

This executable showcase verifies offline publication from a restored documentation graph. All names and endpoints are illustrative.

1 Code examples

Each fenced block retains its language and source text. The optional `example` marker adds a numbered caption. Syntax colours are generated during publication.

1.1 Python

Example 1

```python
import json

# Unicode and multiline strings remain literal source.
def describe(name: str) -> str:
    message = f"""Hello {name}
A second line stays inside this string."""
    return json.dumps({"name": name, "message": message}, ensure_ascii=False)

print(describe("Bønds"))
```

1.2 Bash

Example 2

```bash
#!/bin/bash
# Environment references and quoted backticks are displayed, never run.
name="Bønds"
printf '%s\n' "$name" '$HOME' '``literal``'
curl --header 'Accept: application/ld+json' 'https://example.org/data'
```

1.3 JavaScript and Node.js

Example 3

```javascript
const name = "Bønds";
const description = `Hello ${name}`;
const record = { name, active: true, count: 42 };
console.log(JSON.stringify(record));
```

1.4 TypeScript

Example 4

```typescript
interface DataProduct {
  name: string;
  active: boolean;
}
const product: DataProduct = { name: "UK Bonds", active: true };
console.log(product.name);
```

1.5 JSON

Example 5

```
{
  "name": "UK Bonds",
  "active": true,
  "count": 42,
  "markup": "<script>& </code>",
  "missing": null
}
```

1.6 JSON-LD

Example 6

```
{
  "@context": {
    "@vocab": "https://schema.org/",
    "name": { "@id": "name", "@language": "en" }
  },
  "@id": "https://example.org/products/bonds",
  "@type": "Dataset",
  "name": "UK Bonds"
}
```

1.7 SPARQL

Example 7

```
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX dcat: <http://www.w3.org/ns/dcat#>

# Both SPARQL variable spellings are supported.
SELECT ?dataset $title WHERE {
  ?dataset a dcat:Dataset ;
           dct:title $title .
  FILTER(LANG($title) = "en")
}
ORDER BY $title
LIMIT 20
```

1.8 Turtle

Example 8

```
@prefix ex: <https://example.org/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
```

```
# A labelled dataset with a numeric count.
ex:bonds a ex:Dataset ;
    ex:label "UK Bonds"@en ;
    ex:count 42 ;
    ex:active true .
```

Note
These examples are documentation, not executable build steps. Publication needs no browser script, syntax service or additional Typst package download.

A Document provenance

This record identifies the generator and source revisions collected for this publication.

Generator	GraphArch 0.2.29 (build 412)
Generator commit	`8908a9610d77`

No Git repository revision was recorded for these sources.