What is Rune?

- Rune is a dynamic content preprocessor that unifies content from multiple sources (YAML, HTML, Markdown, SVG, PNG, ZIP, JPG, and JSON files) into a single JSON structure.

- Built for the entire team — developers, designers, and content creators —Rune simplifies collaboration by providing input formats that are familiar and easy to use.

Key Features

1. Dynamic Content Processing: Merges YAML and HTML files, embeds assets as Base64.
2. Multilingual Support: Integrates translations in a flat, i18n-compatible structure.
3. Embedded Assets: Supports images (SVG, PNG, JPG) and downloadable files (ZIP, etc).
4. Portability: Produces a self-contained JSON
5. Extensibility: Supports reusable components and custom structures.

Rune and the Basket of Fruits

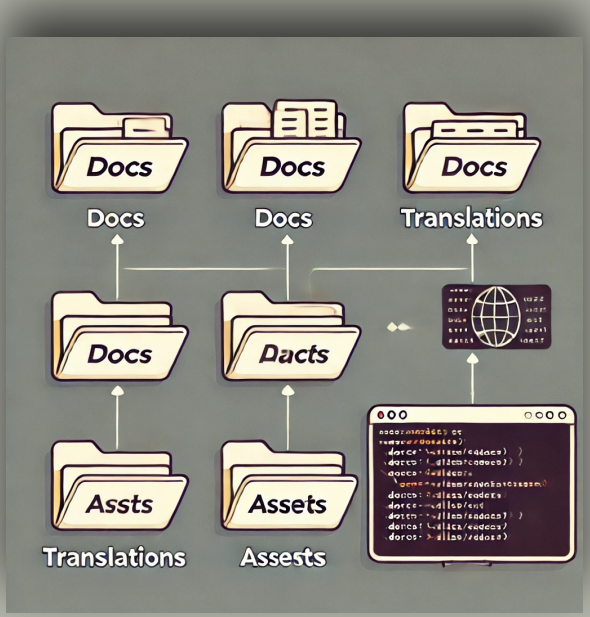
- Rune is like a master basket weaver:
  - HTML files are apples: structured and familiar to frontend developers.
  - YAML files are oranges: layered and segmented for backend developers.
  - JSON translations are bananas: easy to peel and ideal for localization.
  - Markdown files are grapes: simple and versatile for documentation.
  - Assets are cherries: vibrant and ready for inclusion.

How It Works

1. Prepare a directory with YAML, HTML, and assets.
2. Add translations for multilingual support.
3. Run Rune to consolidate all content into a single JSON file.
4. Use the output JSON in your application or provide it through an API.

Example Workflow

1. Add a HTML view:
   - docs/HelloWorld.html
2. Add a translation file:
   - docs/translations/HelloWorld.en.json
3. Run Rune:
   - `rune docs json`
4. The output is an array in JSON which contains all views, translations, and embedded assets.
5. Output can be extended easily by adding more items later (e.g. downloadable attachments in the API backend)



Advanced Features

- Reusable components with parameters and nested children.
- Automatic embedding of images and assets as Base64 data urls.
- Extensibility for new file types and custom configurations.

Functional Source License

- License is Functional Source License, Version 1.1, MIT Future License
- Two years after release it’s standard MIT, but probably we’ll republish it as open source MIT or AGPL before that.