YAML, TOML, JSON Cheatsheet

YAML (YAML Ain't Markup Language)

A human-friendly data serialization standard for all programming languages. Frequently used for config files (e.g., Docker, Kubernetes, CI/CD).

Key Features

- Indentation-based hierarchy (Spaces only).
- Supports comments (#).
- Key-value pairs: `key: value`.
- Lists: Start with `-`.
- Multi-line strings: `|` (preserves newlines), `>` (folds newlines).

Example

```yaml
title: YAML Example
owner:
  name: John Doe
  dob: 1979-05-27
database:
  enabled: true
  ports:
    - 8000
    - 8001
```

TOML (Tom's Obvious, Minimal Language)

A configuration file format that is easy to read due to obvious semantics. Used by Rust (Cargo), Python (Poetry), etc.

Key Features

- Key-value pairs: `key = "value"`.
- Table-based hierarchy: `[table_name]`.
- Explicit data types (Strings, Integers, Booleans, Arrays, Datetimes).
- Supports comments (#).

Example

```toml
title = "TOML Example"

[owner]
name = "John Doe"
dob = 1979-05-27T07:32:00Z

[database]
enabled = true
ports = [ 8000, 8001 ]
```

JSON (JavaScript Object Notation)

A lightweight data-interchange format based on JavaScript object syntax. The de facto standard for Web APIs.

Key Features

- Key-value pairs: `"key": "value"` (Keys MUST be double-quoted).
- Hierarchy using braces `{}` and brackets `[]`.
- Formal and strict syntax.
- Does NOT support comments (officially).

Example

```json
{
  "title": "JSON Example",
  "owner": {
    "name": "John Doe",
    "dob": "1979-05-27"
  },
  "database": {
    "enabled": true,
    "ports": [8000, 8001]
  }
}
```

Comparison Summary

Feature	YAML	TOML	JSON
Main Use	Cloud/ Infra Config	App Config (Cargo/ Poetry)	Data Exchange (API)
Readability	Very High	High	Medium
Comments	Support (#)	Support (#)	Not Supported
Hierarchy	Indentation	Table [ ]	Braces {}
Strictness	Moderate	High	Very High
Quotes on Keys	Optional	Optional	Mandatory