

JSON a Colleague Can Validate

Agree on fields, types and missing values before exchanging records

01

02

03

How to use this guide

People handing structured records between tools, developers or small operational systems.

A practical workbook using an original inventory handoff, a reviewed schema, valid and invalid cases, exact identifiers and JSON Pointer exercises.

- Distinguish valid JSON from a valid business record.
- Specify required fields, absent and null values, identifiers and extra-field policy.
- Reproduce positive and negative contract checks before a handoff.

Scope and limitations

- The inventory and organizations are fictional teaching examples.
- The included checker validates this small exercise contract; it is not a general JSON Schema implementation.
- No production API, database migration or real inventory update is performed.
- Schema inference cannot discover business meaning or prove compatibility with an undocumented consumer.

Published by KitForma. Edition 1.0. Sources checked 1 October 2026. People and businesses in worked examples are fictional. This is an independent publication, not endorsed by the platforms mentioned.

Copyright 2026 KitForma. Usage, support and refund terms are included in LICENSE.txt in the purchased package.

Contents

How to use this guide	2
Contents	3
Define the handoff before its file format	4
Sources and tools	6

01

Define the handoff before its file format

Two systems can read the same JSON file and still disagree about what it means. One may treat quantity as available stock while the other assumes it includes reserved items. One may replace a missing note with an empty string while the other preserves the previous note. A data contract makes those decisions explicit before the record crosses the boundary. Start with the job the recipient must perform and the meaning of each value, then choose the representation that supports that agreement.

The original exercise is a small inventory snapshot from the fictional North store room to a reporting application. It contains three item records and does not update real stock. Quantity means the counted units in this snapshot, with no reservation or future-delivery calculation. The recipient's task is to display the items accurately and identify unresolved restock notes. It must not infer a purchase order, change a supplier record or merge the snapshot into a live database merely because the file parses successfully.

Write a short contract brief naming the producer, consumer, operation and failure behavior. For this example, an invalid snapshot is rejected as a whole and returned for review. A partial-import policy would require a separate record of accepted and rejected items; it is not silently assumed. The operation is a snapshot, not a patch, so omission of an item does not authorize deletion elsewhere. That single sentence prevents a serious misunderstanding that no indentation style or field-name convention could resolve by itself.

Keep an original sample, but do not treat it as the whole agreement. A sample shows what happened to be present once. It may contain no

empty values, no unusual identifiers and no newly introduced field. The workbook includes deliberately varied and invalid records so the boundary can be tested. The goal is a small contract that both sides can explain, together with evidence that the receiving implementation accepts intended cases and rejects known mistakes without changing their meaning.

Sources and tools

[S1] RFC 8259: JSON data interchange

<https://www.rfc-editor.org/rfc/rfc8259>

Checked: 2026-10-01

[S2] JSON Schema: Object reference

<https://json-schema.org/understanding-json-schema/reference/object>

Checked: 2026-10-01

[S3] RFC 6901: JSON Pointer

<https://www.rfc-editor.org/rfc/rfc6901>

Checked: 2026-10-01

JSON formatter

Inspect syntax and layout; formatting does not establish the contract.

<https://kitforma.com/en/json-formatter/>

Infer JSON Schema

Produce a sample-derived starting point that must be reviewed for business rules and optional fields.

<https://kitforma.com/en/json-schema-inference/>

JSON Pointer query

Retrieve an explicit existing path without evaluating code.

<https://kitforma.com/en/json-pointer-query/>

Compare JSON values

Describe changes independently of object key order; arrays compare by index, not identity.

<https://kitforma.com/en/json-structure-diff/>

Inside the complete guide

Separate syntax, structure and business meaning

Choose identifiers that remain exact

Make absence and null mean different things

Turn a sample into a reviewed schema

Use exact paths when discussing a failure

Build examples that can prove the boundary wrong

Plan changes before sending a new version

Capstone: deliver the North inventory contract

Keep the agreement useful after the first exchange