RSW-0002 — Manifest Specification

Revision 0.01 | Protocol rsw/1 | Format 1 | Experimental proposal

Developed by Pablo Murad — November 23, 2026 — pablo@pablomurad.com — https://rsw.works

Experimental Project Specification. This document is an experimental specification of the Really Small Web project. It is a work in progress and has not been published as an RFC or approved as an Internet Standard.

License: specifications CC0 1.0 (public domain); future reference implementation and official tools MIT.

RSW-0002 — Manifest Specification

Revision 0.01 | Protocol rsw/1 (proposed) | Format 1

Experimental proposal

Developed by Pablo Murad

Pablo Murad — November 23, 2026

pablo@pablomurad.com

01. Scope and Terminology

RSW-0002 specifies the manifest.json of a Really Small Web cartridge. It complements RSW-0001 (Cartridge Format, format 1) and the conceptual document v0.02. It does not replace the binary PNG/ZIP validation, does not define the hosting index, and does not grant execution permissions.

Status: experimental normative proposal, revision 0.01. The new choices in this specification must be ratified together with RSW-0003, RSW-0004, RSW-0007, RSW-0011, RSW-0013, and RSW-0016. There is no official implementation nor any claim of complete interoperability.

MUST and MUST NOT are requirements; SHOULD is a recommendation; MAY is an option. A producer creates manifests; a reader interprets cartridges; a validator verifies rules without rendering content. All treat inputs as untrusted.

Distinguish four concepts: container format (format), core protocol version (protocol), specification revision (0.01), and publication version (version). The conception document v0.02 is not the protocol version.

Proposal: format = 1, protocol = "rsw/1". The rsw/0.01 identifier used in the conception examples was provisional. Version 1 indicates the experimental generation of the interpretation rules, not a promise of stability nor the 1.00 release of the software.

The manifest identifies the cartridge, discovers its reading entry, verifies file and cover integrity, describes authorship and version, and declares references. The digital signature will be specified in RSW-0003. Discovery or rendering is never triggered simply by the presence of a field.

02. Serialization and Limits

The manifest MUST exist exactly once at the root of the ZIP, named manifest.json. All RSW-0001 requirements apply: valid UTF-8 without BOM, JSON object at the root, no duplicate members at any level, no isolated Unicode surrogates, maximum depth 32 (root = 1), size of 1 to 16,384 bytes inclusive. Only JSON whitespace may precede or follow the object. Do not allow comments, trailing commas, NaN, or Infinity.

Field names are case-sensitive. Mandatory fields do not admit null. Absent optional fields MUST NOT receive implicit values before signature verification. No Unicode normalization, case conversion, trimming, or type coercion modifies received values. Visual presentation may escape characters without altering the original.

All numeric quantities in the manifest, including inside extensions, MUST be JSON integers in decimal notation, without fraction, exponent, or -0, within the range -9007199254740991 to 9007199254740991. This rule avoids precision loss in JavaScript readers and prepares interoperability with JCS (RFC 8785). Do not use float64 to validate the lexical representation before checking the token.

String limits in this document are in UTF-8 bytes after JSON decoding, not in visual characters nor in escaped representation bytes. Unless a stricter rule applies, known textual fields cannot contain controls U+0000 to U+001F or U+007F to U+009F. Strings intended for the interface must be displayed as text, never as HTML.

The producer SHOULD emit compact or readable, stable JSON. Member order has no meaning; arrays keep their order. The bytes to be signed will be defined by RSW-0003; do not sign an arbitrary reserialization nor assume that valid JSON is already canonicalized.

03. Mandatory Fields

format — literal integer 1; identifies the container profile of RSW-0001. Another valid integer indicates UNSUPPORTED_FORMAT.

protocol — literal string "rsw/1"; an unknown, well-formed generation (rsw/ followed by a positive decimal integer without leading zero) indicates UNSUPPORTED_PROTOCOL. An absent value or invalid syntax indicates INVALID_MANIFEST.

name — stable publication identifier, 1 to 64 ASCII bytes. Grammar: [a-z0-9]+(?:-[a-z0-9]+)*. Do not change between versions. Renaming creates another publication identity. It is not a file name, domain, or proof of authorship.

title — human title of 1 to 120 UTF-8 bytes, with at least one non-whitespace character. May vary between versions. Do not use as an identifier.

profile — enum text or static. Determines the reading profile, not the privileges. Well-formed future values (same grammar as name) indicate UNSUPPORTED_PROFILE. The reader may not implement static, but must report the limitation.

type — semantic string of 1 to 32 bytes, same grammar as name. Conventional values: site, page, profile, journal, zine, gallery, book, directory, webring, feed, data, app, game, archive. Unknown values are treated as generic. app and game do not enable scripts.

version — integer from 1 to 9007199254740991. Identifies the edition number, not the software version. First edition: 1; later editions require previous_release. Update acceptance rules belong to RSW-0013.

entrypoint — exact path of an existing internal file, per RSW-0001. For text, ends in .md; for static, ends in .html. It must appear in files. It does not admit a URL, query, fragment, or reserved path.

cover_hash — H_cover from RSW-0001, textual sha256:<64 lowercase hex>. It must match the PNG reconstructed by removing the whole rsWT chunk, without re-encoding the image.

files — object of paths and hashes, per section 05. At least one entry; includes the entrypoint and all non-reserved resources.

04. Optional Fields and Dependencies

description — description of 1 to 512 UTF-8 bytes. Plain text, not HTML.

author — public authorship name, 1 to 120 UTF-8 bytes. May be a pseudonym. It does not verifiably identify a person. It is not required for local cartridges.

public_key — when present, string with the prefix ed25519: followed by 52 uppercase RFC 4648 Base32 characters, without padding, encoding exactly 32 bytes. Unused trailing bits must be zero (re-encode and compare). This encoding is a proposal for RSW-0003; it does not imply cryptographic validity of the key. The presence of public_key requires signature.sig at the root, and vice versa. Absence of both is allowed only for unsigned cartridges. Do not publish as authenticated without RSW-0003.

created_at — optional edition date, strictly YYYY-MM-DDTHH:MM:SSZ (20 ASCII bytes). Valid Gregorian calendar, year 0001 to 9999, hours 00 to 23, seconds 00 to 59. Do not accept offsets, fractions, or leap seconds in this profile. It is not proof of time nor a sufficient criterion for ordering updates.

license — string of 1 to 128 UTF-8 bytes. SHOULD be an SPDX identifier or expression, when applicable, or LicenseRef-... for a custom license documented in an internal file. Absence does not authorize redistribution. The reader does not need to interpret the license legally.

previous_release — textual H_png hash of the declared preceding edition. Mandatory if version > 1, forbidden if version = 1. Contains no location. The reader does not need to own the previous edition to import the cartridge, but must not claim to have verified its chain. RSW-0013 defines conflicts, sequence, updates, and regressions.

tags — optional array of up to 16 distinct strings, each of 1 to 32 bytes in the name grammar. Order is informative.

peers, capabilities, permissions, and extensions are defined in section 06. No other top-level fields are allowed in this generation; extensions use the dedicated object. Optional fields present with the wrong type or null are invalid.

05. Inventory and Integrity

files is a JSON object mapping each non-reserved internal path to exactly one SHA-256 hash of its uncompressed bytes. The member name is the exact ASCII path per RSW-0001. The value is the string "sha256:" followed by 64 lowercase hexadecimal digits. The hash of an empty file is allowed.

The key set of files MUST match exactly the set of regular ZIP files, except manifest.json and signature.sig at the root. No resource can be left out; no nonexistent resource can be listed; these two reserved files cannot appear in the inventory. Do not accept wildcards, aliases, partial lists, or normalized names. The manifest's maximum size of 16 KiB is independent of the ZIP's maximum of 512 files: not every combination of 512 paths and hashes fits in the manifest. The producer must reduce the number of resources or their paths. Do not truncate the inventory.

The reader MUST verify all hashes after structural validation and before declaring the manifest intact. Hashes must use the real bytes, without normalizing line endings, converting encoding, reformatting HTML, or recompressing the file. Compare cover_hash with H_cover. Mismatches are INTEGRITY_MISMATCH. Extra or missing files are INVALID_INVENTORY.

H_png identifies the complete PNG; H_zip identifies the payload bytes; H_cover identifies the cover without rsWT. H_png and H_zip are not manifest members: including them in the ZIP itself would create self-reference. previous_release references another PNG. The cover commitment does not include rsWT itself, avoiding a cycle.

Future signature: RSW-0003 will bind the whole manifest, including files, cover_hash, public_key, version, and extensions, with domain separation and unambiguous serialization. The commitment authenticates the cover and the files when verified; it does not intend to authenticate a specific ZIP encoding. Changing compression may change H_png without changing the committed content.

A hash without an authenticated signature does not prove authorship. An adversary can replace files and recalculate their hashes. No manifest alone can produce a VERIFIED state.

06. References and Extensions

peers — optional array of up to 32 unique strings, each of 1 to 512 UTF-8 bytes. References MUST begin literally with rsw://, have content after the prefix, and contain no ASCII whitespace, controls, or backslashes. Full grammar validation belongs to RSW-0004; discovery semantics belong to RSW-0007. A reference failing URI validation is not navigable and must be flagged. The manifest does not authorize automatic fetch, import, or download. Ordering is preserved; do not deduplicate URIs by normalization at this stage.

capabilities — optional array of up to 16 unique identifiers of 1 to 32 bytes in the name grammar. Initial informational values: discovery and updates. Unknown values may be shown as unsupported; they do not trigger behavior. No capability is mandatory to read the cartridge.

permissions — if present in the MVP, MUST be an empty array. Absence is equivalent to no permission requested for the execution policy, but not for the signed bytes. A non-empty array requests a resource outside the initial profile: return UNSUPPORTED_FEATURE, without granting access or trying to interpret its content. Cartridge JavaScript remains disabled.

extensions — optional object with up to 16 members. Each name follows an ASCII namespace of 3 to 80 bytes, in lowercase, composed of at least two dot-separated segments, each following the name grammar (e.g., com.example.notes). Values are JSON within the general size, depth, and number rules. An unknown extension must be preserved when the manifest is copied, and ignored for interpretation. It cannot alter core fields, require privileges, declare new mandatory rules, or make an invalid publication valid.

Unknown fields at the root are INVALID_MANIFEST. New interpretation requirements that cannot be ignored require formal protocol or profile evolution, not a silent extension. Compatibility handling will be refined in RSW-0016. A reader must never automatically rewrite a signed manifest.

07. Validation and States

1. Run RSW-0001, including limits, container verification, and file names. Stop on structural failure.
1. Read manifest.json with a limited, duplicate-sensitive parser. Check the lexical representation, Unicode, depth, and size. Detect format and protocol versions before interpreting fields specific to this generation.
1. Validate mandatory fields, types, grammars, limits, and conditional relations. Verify that profile and entrypoint are compatible. Do not treat the author's name as identity proof.
1. Compare the inventory with the ZIP and compute the hashes of all files and the cover. Any discrepancy prevents importing as a valid cartridge.
1. Check public_key/signature.sig coherence. If both are absent, classify as UNSIGNED and allow only local import with a warning. If present, manifest validation does not conclude authentication: mark SIGNATURE_PENDING until RSW-0003 is executed. If the cryptographic verification fails, reject as INVALID. If successful, local policy may distinguish VERIFIED from TRUSTED.
1. Apply the profile, rendering, local resource, and URI policies of the corresponding specifications. The result MANIFEST_VALID does not mean rendering is safe nor that the edition is the most recent. No script, remote access, or privileged operation is an automatic consequence of validation.

Proposed additional error categories: INVALID_MANIFEST (syntax/fields), UNSUPPORTED_PROTOCOL (generation), UNSUPPORTED_PROFILE (profile), UNSUPPORTED_FEATURE (permissions), INVALID_INVENTORY (divergent inventory), INTEGRITY_MISMATCH (hash mismatch). RSW-0001 structural errors remain applicable. A more restrictive local policy does not make the file universally invalid.

In Go, encoding/json alone does not sufficiently detect all cases of duplicates, isolated surrogates, and lexical number representation. Implement or adopt strict validation before decoding into structs. Use preserved tokens/numbers, crypto/sha256, and limit tests. Do not create heavy dependencies just to validate the manifest.

08. Real and Reproducible Example

The separately distributed vector, RSW-0002-exemplo.rsw.png, contains a white 1 x 1 pixel RGB PNG, a Store ZIP with manifest.json and index.md, unsigned. Textual attribution does not prove cryptographic authorship. The manifest below is in the example JSON file; the hashes are real, not placeholders.


```
{
  "format": 1,
  "protocol": "rsw/1",
  "name": "hello",
  "title": "really small web.",
  "profile": "text",
  "type": "page",
  "version": 1,
  "author": "Pablo Murad",
```

```
  "entrypoint": "index.md",
  "cover_hash": "sha256:e878950f8091ec010cf5cc723bdea027a8539cf7147cfea199c2f666232dcd4e",
  "files": {
    "index.md": "sha256:786bdee1c1944ede3eca85eeed8b2147046a86088598e11e3a36f951be700df4"
  }
}
```

index.md (UTF-8, LF, with trailing newline):

```
# really small web.

Desenvolvido por Pablo Murad.

PNG: 749 bytes; ZIP: 668 bytes; manifest: 401 bytes.
H_png: sha256:e3a9d3d96b055f8092b3a5e0e88000fc17356f136fa15a15e8c3ccad1f29457c
H_zip: sha256:f7b2bd18654da4017cb968fd71634c4a88f93a310cb093d41285dbf2e5f07ade
H_cover: sha256:e878950f8091ec010cf5cc723bdea027a8539cf7147cfea199c2f666232dcd4e
```

The image was decoded, the ZIP tested, and the hashes and inventory checked by script. The vector was produced according to the proposed rules; it does not constitute certification by an official parser, which does not yet exist. The previous minimal RSW-0001 vector does not need to change: it remains structurally valid, but it does not meet the full RSW-0002 manifest.

09. Interoperability and Remaining Work

Positive vectors to publish: complete minimal manifest without signature; static manifest; absent and present optional fields; Unicode title; unknown extension; version 2 with previous_release; valid key and signature after RSW-0003; exact byte and count limits, also respecting the total size.

Negative vectors: BOM; duplicate member at the root or in an inner object; invalid UTF-8; isolated surrogate; root array; exponential, fractional, -0, or out-of-range integer; missing mandatory field; future protocol; unknown profile; unknown root field; invalid path; entrypoint not listed or incompatible; uppercase or tampered hash; missing or extra file; modified cover; version > 1 without previous_release; version 1 with previous_release; public_key without signature.sig or vice versa; Base32 with non-zero trailing bits; non-empty permissions; peer with whitespace; repeated tag; extension with invalid namespace; manifest above 16 KiB.

Interoperability criterion: two independent implementations must produce the same H_png/H_zip/H_cover values and per-entry hashes, the same manifest acceptance states, and the same field interpretation. Identical diagnostic order is not required for inputs with multiple failures.

Pending before ratification: define signed bytes, signature representation, key validation, and cryptographic tests (RSW-0003); complete URIs and peers (RSW-0004/0007); HTML/CSS/Markdown rules and isolation (RSW-0011); publication evolution and conflicts (RSW-0013); formal compatibility (RSW-0016). Implement a strict parser, validator, CLI, negative fixtures, and fuzzing. Reassess decisions jointly without silently changing published formats.

Out of scope: hosting indexes, automatic discovery, relays, chat, comments, private messages, JavaScript runtime, interactive permissions, P2P, payload encryption, key rotation and recovery. No new infrastructure requirement was introduced.

References: RSW v0.02, 23/11/2026; RSW-0001, revision 0.01; RFC 8259 (JSON), https://www.rfc-editor.org/rfc/rfc8259 ; RFC 8785 (JCS), https://www.rfc-editor.org/rfc/rfc8785 ; RFC 4648 (Base32), https://www.rfc-editor.org/rfc/rfc4648 ; RFC 8032 (Ed25519), https://www.rfc-editor.org/rfc/rfc8032 . License: CC0 1.0 (public domain dedication; see CC0-1.0.txt in the downloads area).

Next specification: RSW-0003 — Identity and Signatures.

Developed by Pablo Murad | pablo@pablomurad.com.

END — RSW-0002

Author's Address

Pablo Murad
Independent
pablo@pablomurad.com
https://pablomurad.com
Official project website: https://rsw.works

Small files. Independent worlds.