RSW-0001 — Cartridge Format

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-0001 — Cartridge Format

Revision 0.01 | Format 1 | Experimental proposal

November 23, 2026

Developed by Pablo Murad

pablo@pablomurad.com

01. Scope and Normative Language

This specification defines the Really Small Web cartridge binary container, proposal revision 0.01, for the MVP described in the conceptual document v0.02. The format version is 1; it is not the network protocol version nor the software version. Status: experimental normative proposal, not yet ratified or officially implemented.

MUST and MUST NOT indicate mandatory requirements; SHOULD indicates a recommendation with justifiable exceptions; MAY indicates an option. A producer writes cartridges; a reader validates and reads them; a validator may operate without rendering. Structural conformance with this document is not equivalent to full conformance with RSW.

A cartridge is a static cover PNG with a single private chunk containing a ZIP. The PNG remains readable as an ordinary image. The recommended extension is .rsw.png; the extension and the Content-Type are not proof of the format. On the web, image/png is used. No new media type is registered by this proposal.

Out of scope: full manifest schema (RSW-0002), signature and trust (RSW-0003), URI and resolution (RSW-0004/0005), hosting (RSW-0006), and the rendering policy (RSW-0011). No execution or network request results from importing.

02. PNG Envelope and the rsWT Chunk

The file MUST begin with the eight-byte PNG signature: 89 50 4E 47 0D 0A 1A 0A. The v1 cover profile is a deliberately narrow subset of PNG: IHDR, one or more consecutive IDATs, one rsWT, and IEND, in that order. No other chunk is allowed in this profile. The IHDR has 13 bytes, bit depth 8, color type 2 (RGB) or 6 (RGBA), compression 0, filter 0, and interlace 0. The image MUST have positive dimensions within the limits of section 04. The producer converts covers of other types, removing metadata, before packaging; if already compatible, it MAY preserve the bytes. There is no APNG, palette, ICC, or EXIF in the v1 cover.

Each chunk has a uint32 big-endian length, a four-byte ASCII type, data, and a uint32 big-endian CRC-32. The CRC uses the PNG algorithm (ISO-HDLC), over type and data, not over the length. All CRCs MUST be valid. The length MUST fit in the remaining bytes before reading or allocating. IEND has zero length and ends exactly at EOF; additional bytes are forbidden.

The rsWT type is 72 73 57 54 in hexadecimal: ancillary, private, valid reserved bit, and not safe to copy when the critical image changes. The producer writes exactly one rsWT immediately before IEND. Its data field is the complete raw ZIP, without Base64, additional header, padding, encryption, or external compression. An empty, repeated, or out-of-position rsWT is invalid. The reader does not search for ZIP signatures by arbitrary scanning.

The IDATs, concatenated, MUST form a single valid PNG zlib stream, with correct checksum, no residual bytes, and expanding exactly to height x (1 + width x channels) bytes of filtered scanlines. The filters MUST be in the range 0..4. A limited PNG decoder can validate this, complemented by full-consumption checks. An ordinary PNG without rsWT is not a cartridge; if a malformed rsWT is present, do not treat it as a valid cartridge.

03. ZIP Profile and Internal Paths

The payload MUST be a single-disk ZIP (APPNOTE), without prefix or suffix, without ZIP64, encryption, volumes, comments, extra fields, directory entries, symbolic links, hardlinks, or special files. Allowed methods: 0 (Store) and 8 (Deflate, RFC 1951 stream without zlib). Allowed general flags: 0x0000 or 0x0800; data descriptors (bit 3) are not allowed. The producer knows sizes and CRCs before writing. For Go, pre-compressing and using archive/zip.CreateRaw is an option; the default writer may produce incompatible descriptors.

The first local record starts at offset zero. The local records, their names, and compressed data occupy contiguous ranges without gaps or overlaps up to the central directory; this contains exactly one record per file, and ends at the classic 22-byte EOCD, without comment. The EOCD MUST indicate disk 0, directory on disk 0, a coherent record count, and exact offsets/sizes. No other records are admitted. Local and central headers MUST agree on name, method, flags, sizes, CRC, and version/time fields; local offsets MUST be unique and valid. All extra fields and comments have zero length. Stored files have equal compressed and uncompressed sizes. The sizes and CRC-32 of each file MUST match the data actually read; Deflate MUST consume exactly its range and end in a valid stream. Required-version values above 20 do not belong to the v1 profile. Attributes indicating a directory or a non-regular Unix type are rejected; attributes are not restored on the filesystem.

Names are lowercase ASCII, without leading or trailing slashes, with / as the separator. Each segment follows [a-z0-9][a-z0-9._-]*, does not end with a period, and is up to 64 bytes. Full path: up to 240 bytes and 8 segments. Segments whose name before the first period is con, prn, aux, nul, com1..com9, or lpt1..lpt9 are forbidden. Forbid duplicates, a file prefix used as a directory, backslashes, absolute names, uppercase letters, control characters, NUL, and any implicit normalizations. Do not decode percent-encoding in ZIP names. URLs are the subject of another specification.

The ZIP contains only virtual regular files. Directories are implicit. The reader SHOULD access entries in memory or via limited reads, not extract to arbitrary folders. When exporting, use a new user-controlled directory, without following links, without overwriting files, and without restoring executable permissions.

04. Mandatory Budget and Limits

All limits below are inclusive and independent. KiB = 1024 bytes; MiB = 1048576 bytes. The producer MUST NOT generate, and the reader MUST NOT accept, cartridges above the v1 profile limits. More restrictive local limits are allowed by policy, but must be reported as a local refusal, not as universal invalidity.

```
Complete PNG: 2 MiB (2,097,152 bytes), including the entire cover and ZIP.
PNG chunks, including IHDR and IEND: 1024.
Cover width and height: 1..2048 pixels each.
Maximum cover area: 4,194,304 pixels.
Expanded ZIP: 32 MiB, sum of all files, including manifest and signature.
Expanded internal file: 8 MiB.
ZIP entries: 1..512 (all regular; manifest included).
manifest.json: 1..16 KiB.
signature.sig, if present: 1..1024 bytes in this structural profile; RSW-0003 will fix the exact rep
resentation.
Path: 240 bytes; segment: 64 bytes; depth: 8 segments.
```

Tiny: complete PNG up to 64 KiB; recommended: up to 256 KiB. These are size classifications, not different formats. The 10 MiB mentioned in the conception is a future possibility, not a v1 option.

The declared size sum does not replace real counting during decompression. The reader MUST stop when exceeding the declared or absolute limit. Use checked arithmetic and streaming limits. No additional

compression-ratio limit is required for conformance if all absolute and working limits are applied. An implementation MUST allow cancellation, limit concurrent operations, and maintain a finite budget of CPU, memory, and disk. Timeouts vary by equipment; they do not define the validity of the bytes. Internal resources, including images, need their own decoding limits in RSW-0011; do not open them during structural validation.

05. Minimal Manifest and Byte Identity

The ZIP MUST contain exactly one manifest.json at the root. It must be strict UTF-8 JSON, without BOM, with an object at the top level, without duplicate member names at any level, without content after the object other than JSON whitespace. Strings must represent valid Unicode, without isolated surrogates. The reader MUST limit nesting to 32 levels of objects/arrays (root = 1). This document establishes only the framing fields:

```
format: JSON literal integer 1. Another integer version is UNSUPPORTED_FORMAT; absent, wrong type, o
r non-integer representation is INVALID_MANIFEST.
entrypoint: string with a valid path pointing to an existing regular file, other than manifest.json
and signature.sig. For the MVP, extension .md or .html; other types require a future profile extensi
on.
```

RSW-0002 will define the remaining mandatory and optional fields (protocol, name, profile, version, author, files, etc.), evolution rules, and compatibility. Additional fields do not by themselves change the result of structural validation. A minimal manifest.json does not necessarily represent a fully valid publication. signature.sig is reserved at the root; its presence does not mean a valid signature. A reader MUST NOT classify a cartridge as VERIFIED unless it implements RSW-0003 and validates the corresponding requirements.

Three distinct hashes are defined, all SHA-256 over exact bytes, presented as 64 lowercase hex digits with a sha256: prefix when textual. H_png: the complete file from the first byte of the signature to the end of IEND. H_zip: the rsWT data field without its header or CRC. H_cover: the original PNG reconstructed by removing exactly the whole rsWT chunk (length, type, data, CRC) and preserving all other bytes in order. H_cover is the proposed basis for the signed cover commitment of RSW-0003. H_png is never inserted as a hash of itself. Repacking, changing compression, or altering the cover may change H_png without changing the internal files.

The authorship signature and the binding of the manifest to H_cover are not defined here. CRCs and hashes without an authenticated reference detect errors or allow identification, but do not prove authorship. The rsw://sha256/ address will use H_png, subject to the RSW-0004 grammar.

06. Reading and Production Algorithm

Staged reading, always treating the input as untrusted:

1. Read up to 2 MiB + 1; reject excess. Validate the signature, lengths, order, and CRCs of all chunks. Check IHDR and budget before decoding the cover. For a PNG without rsWT, return NOT_RSW after limited structural reading of the chunks; the v1 cover restriction applies only to candidates with rsWT. If the PNG is malformed or above the limit, report the corresponding error. Distinguish an ordinary PNG from an invalid RSW.
2. Verify the static PNG image, the IDAT stream, EOF, and the unique presence of rsWT. Compute H_png and H_cover over the original bytes, without re-encoding.
3. Delimit rsWT and validate the EOCD, central directory, and local headers. Check counts, sizes, offsets, flags, paths, and types before expanding. Compute H_zip.
4. Decompress each entry in a limited stream; validate the real size, CRC, and complete consumption. Do not publish partially validated content.
5. Read manifest.json and check the minimal rules and the entrypoint. If the format is unknown, stop without trying to interpret content. Forward to the RSW-0002/0003 validator, when available.

6. Only after applicable validation, offer import into the local library. The MVP does not execute scripts; rendering only in the isolated environment defined by RSW-0011. Check remote references and separate resources independently, never automatically at this stage.

Production: choose the cover; normalize it to the v1 profile; gather regular files; validate names and limits; create the manifest; if signed, apply the future RSW-0003; write the ZIP; compute the rsWT CRC-32; insert before IEND; validate the result using the same parser as the reader; only then publish or write atomically. Preserve the original file in the library. The packer must not accept a PNG with a pre-existing rsWT without an explicit repacking operation.

Minimum errors: NOT_PNG; NOT_RSW; INVALID_PNG; INVALID_CHUNK; LIMIT_EXCEEDED; INVALID_ZIP; INVALID_PATH; INVALID_MANIFEST; UNSUPPORTED_FORMAT; UNSUPPORTED_PROFILE; IO_ERROR. Codes are categories, not HTTP statuses. More than one failure may occur: identical diagnostic priority is not required. Do not render, extract, or record as valid in case of failure. A refusal by local policy must be distinguishable from a format error.

07. Conformance, Tests, and Remaining Work

Minimum interoperability criteria: the same file produces the same extracted bytes, H_png, H_zip, H_cover, and structural validation result in two independent implementations. The PNG must open as an image in a conventional viewer. The reader must reject variants outside the profile even when a generic PNG or ZIP library accepts them. Do not mistake a library's permissiveness for RSW conformance.

Vectors to publish: minimal Store; minimal Deflate; RGB and RGBA; absence of rsWT; incorrect PNG CRC; duplicated, empty, or out-of-order rsWT; data after IEND; PNG above 2 MiB; cover above the limit; truncated or excess IDAT; ZIP64; descriptor flag; entries with conflicting offsets; unsafe path; duplicate entry; file/directory collision; incorrect ZIP CRC; manifest absent, duplicated, invalid UTF-8, or with a future version; entrypoint missing; expansion at the exact limit and above. Add PNG/ZIP/JSON fuzzing, memory-consumption tests, and independent interoperability tests.

The reference implementation in Go should use encoding/binary and hash/crc32 for chunks, image/png for additional image validation, archive/zip and compress/flate for ZIP, and crypto/sha256 for hashes. The strict subset requires extra checks of headers, offsets, and limits; archive/zip and image/png alone do not guarantee conformance. Using a small dependency is acceptable when necessary for security.

Still missing: fully specifying the manifest and signatures; closing URIs and resolution; defining safe rendering; implementing the CLI and library; generating negative fixtures and complete reference tests; validating in at least two implementations; measuring performance and ratifying this proposal. RSW-0001 does not declare the MVP complete. Outside this profile: LSB, GIF, animated cartridges, multiple payloads, ZIP64, package encryption, JavaScript, social APIs, relays, P2P, and messaging. A future extension cannot silently change v1 requirements.

External references: W3C PNG Third Edition, https://www.w3.org/TR/png-3/ ; PKWARE ZIP APPNOTE, https://support.pkware.com/pkzip/appnote ; RFC 1951 (Deflate), https://www.rfc-editor.org/rfc/rfc1951 ; RFC 8259 (JSON), https://www.rfc-editor.org/rfc/rfc8259 ; FIPS 180-4 (SHA-256), https://csrc.nist.gov/pubs/fips/180-4/upd1/final .

Internal reference: Really Small Web, conception document v0.02, 23/11/2026. License: CC0 1.0 (public domain dedication; see CC0-1.0.txt in the downloads area). The document remains an experimental specification; the license is ratified. Developed by Pablo Murad.

A. Informative Minimal Vector

Example file: rsw-0001-exemplo.rsw.png, delivered separately. A white 1 x 1 pixel RGB cover; a Store ZIP with two entries, unsigned. The presence of a credit in the content does not constitute cryptographic proof of authorship. The example satisfies the RSW-0001 framing; it does not pretend to validate a future RSW-0002 schema or an RSW-0003 signature.

```
manifest.json (UTF-8, no trailing newline):
{"format":1,"entrypoint":"index.md"}

index.md (UTF-8, LF, with trailing newline):
# really small web.

Desenvolvido por Pablo Murad.

Complete PNG: 384 bytes. ZIP: 303 bytes. Cover: 69 bytes.
H_png: 8eeefa77318c84b0c3077a121f149d57166d8e8f89aec772a7213987020f95ca
H_zip: e1cc2b74bba7d645fa3a632fb43165c3e0ae33d2ba0aee2bee05f480cdb7bd3c
H_cover: e878950f8091ec010cf5cc723bdea027a8539cf7147cfea199c2f666232dcd4e
rsWT CRC: d4d9fa1d
```

The example was generated and opened as a PNG, and the ZIP was tested with the standard library. This is not equivalent to a complete test of the reference parser, which has not yet been implemented.

Author's Address

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

Small files. Independent worlds.