WARC reader process

Describes the steps taken to read and validate a WARC record.

Steps to parsing a WARC record:

The following steps are taken when parsing a WARC record:

1. Parse and possibly skip lines until a valid "WARC/x.x" version line is identified. (Generates warnings if empty or unknown lines precede a version line)
2. Parse and identify valid header lines until an empty line is encountered. (Generates warnings if a WARC header value is empty or invalid)
3. Validation of required and misplaced WARC headers based on the WARC-Type, if present. (Generates warnings if values are not present/absent as per the WARC specs)
4. The payload is wrapped up in a InputStream and made accessible through WARC record methods for payload processing. In case of http(s) payload content the http headers are parsed and made accessible along with the actually http response data. The record and the payload are digested if requested.
5. When payload processing is done and the WARC record is closed the final step is to look for the mandatory trailing two linefeeds. (Generates warnings in case more or less than two lines are parsed)

Step 1:

Before a record can be parsed it must first be identified. So the first step is to look for the WARC version line in the stream.

One line is read at a time and compared to a valid WARC version line. The parser is faily strict and will accept "WARC/" and an invalid version string.

Accepted version strings are in the format of "x.x[x.x]" even though this will most likely never happen.

Warnings/Errors range from leading garbage before a valid WARC identifier line to invalid version information and missing CR-LF pairs.

Step 2:

This part of the record parser looks for valid header lines. This process is only terminated when an empty line is encountered.

Each possible header line is analyzed for correctness. Basic correctness is the presence of a ":" delimiter between the header-name and header-value.

Furthermore header-names can only contain US-ASCII character excluding control characters, white spaces, etc.

Header-values are valid in the presence of either US-ASCII characters, UTF-8, quoted strings or encoded words. All the encodings can be used in sequence but no simultaniously.

Header-values can span multiple lines using FWS (Feeding White Space).

Warnings/Errors range from invalid headers, missing or empty values, incorrect encoding to invalid uri/date/numeric/ip/content-type/digest formats.

Step 3:

This step is central in validating the WARC record header. Depending on the "WarcType-Id" the headers present are examined according to the profile for that type.

Warnings/Errors range from missing required fields to the presence of unwanted fields.

Step 4:

Parsing of the record header is now done and the payload can now be processed. Payload processing is only possible when a valid "Content-Length" has been parsed.

The payload is made available through a fixed length input stream which is problematic without a valid length.

If requested the record digest is computed. The payload digest is computed if requested but only in case the record has a defined payload with a leading header. (Currently only http response header)

Warnings/Errors on the payload stream are non existing and at the discretion of post processing parsing.

Step 5:

If WARC digest headers are present in the record and digests have been computed while reading the payload they will be compared.

In accordance with the WARC specifications two trailing linefeeds are required after a record.

Warnings/Errors reported are restricted to the presence of more or less that two linefeeds.