gitprotocol-common

gitprotocol-common(5) ==========

NAME ---- gitprotocol-common - Things common to various protocols

SYNOPSIS --------

> <over-the-wire-protocol>

DESCRIPTION -----------

This document defines things common to various over-the-wire protocols and file formats used in Git.

ABNF Notation -------------

ABNF notation as described by RFC 5234 is used within the protocol documents, except the following replacement core rules are used:

```
   HEXDIG    =  DIGIT / "a" / "b" / "c" / "d" / "e" / "f"
```

We also define the following common rules:

```
   NUL       =  %x00
   zero-id   =  40*"0"
   obj-id    =  40*(HEXDIGIT)

   refname  =  "HEAD"
   refname /=  "refs/" <see discussion below>
```

A refname is a hierarchical octet string beginning with "refs/" and not violating the ’git-check-ref-format’ command’s validation rules. More specifically, they:

1. They can include slash `/` for hierarchical (directory) grouping, but no slash-separated component can begin with a dot `.`.

1. They must contain at least one `/`. This enforces the presence of a category like `heads/`, `tags/` etc. but the actual names are not restricted.

1. They cannot have two consecutive dots `..` anywhere.

1. They cannot have ASCII control characters (i.e. bytes whose values are lower than \040, or \177 `DEL`), space, tilde `~`, caret `^`, colon `:`, question-mark `?`, asterisk `*`, or open bracket `[` anywhere.

1. They cannot end with a slash `/` or a dot `.`.

1. They cannot end with the sequence `.lock`.

1. They cannot contain a sequence `@{`.

1. They cannot contain a `\`.

pkt-line Format ----------

Much (but not all) of the payload is described around pkt-lines.

A pkt-line is a variable length binary string. The first four bytes of the line, the pkt-len, indicates the total length of the line, in hexadecimal. The pkt-len includes the 4 bytes used to contain the length’s hexadecimal representation.

A pkt-line MAY contain binary data, so implementors MUST ensure pkt-line parsing/formatting routines are 8-bit clean.

A non-binary line SHOULD BE terminated by an LF, which if present MUST be included in the total length. Receivers MUST treat pkt-lines with non-binary data the same whether or not they contain the trailing LF (stripping the LF if present, and not complaining when it is missing).

The maximum length of a pkt-line’s data component is 65516 bytes. Implementations MUST NOT send pkt-line whose length exceeds 65520 (65516 bytes of payload + 4 bytes of length data).

Implementations SHOULD NOT send an empty pkt-line ("0004").

A pkt-line with a length field of 0 ("0000"), called a flush-pkt, is a special case and MUST be handled differently than an empty pkt-line ("0004").

```
   pkt-line     =  data-pkt / flush-pkt

   data-pkt     =  pkt-len pkt-payload
   pkt-len      =  4*(HEXDIG)
   pkt-payload  =  (pkt-len - 4)*(OCTET)

   flush-pkt    = "0000"
```

Examples (as C-style strings):

```
pkt-line          actual value
---------------------------------
"0006a\n"         "a\n"
"0005a"           "a"
"000bfoobar\n"    "foobar\n"
"0004"            ""
```

GIT --- Part of the linkgit:git[1] suite

□□□ #1
□ □□□ □□□ 2026-10-31 20:14:19 UTC
□ □□□ □□□ 2026-10-31 20:14:19 UTC