Package: awscrt (via r-universe)

October 28, 2026

Title Hardware-Accelerated 'CRC Checksums' from the AWS Common Runtime

Version 0.1.1

Description Computes 'CRC32', 'CRC32C' and 'CRC64NVME' 'checksums' using the 'aws-checksums' library from the 'AWS' Common Runtime, with runtime dispatch to hardware instructions ('SSE4.2', 'PCLMULQDQ', 'AVX-512', 'ARMv8' 'CRC' and 'PMULL') and a portable software fallback.

License Apache License (>= 2)

URL https://github.com/paws-r/aws-crt-r, https://paws-r.r-universe.dev/awscrt

BugReports https://github.com/paws-r/aws-crt-r/issues

Encoding UTF-8

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

SystemRequirements GNU make, C compiler

Config/roxygen2/version 8.1.0

NeedsCompilation yes

Author Dyfan Jones [aut, cre], Amazon.com, Inc. or its affiliates [cph] (Author of the bundled aws-checksums and aws-c-common libraries)

Maintainer Dyfan Jones <dyfan.r.jones@gmail.com>

Config/pak/sysreqs make

Repository https://cran.r-universe.dev

Date/Publication 2026-10-28 11:40:16 UTC

RemoteUrl https://github.com/cran/awscrt

RemoteRef HEAD

RemoteSha 2c39b9909ad66360a22afd2011a24b4aba205b15

Contents

checksums 2
checksums-file 3
cpu_features 4

Index **5**

`checksums` *CRC checksums*

Description

Compute CRC32, CRC32C (Castagnoli) and CRC64NVME checksums with the `aws-checksums` library from the AWS Common Runtime.

Usage

```
crc32(x, previous = NULL, raw = FALSE)

crc32c(x, previous = NULL, raw = FALSE)

crc64nvme(x, previous = NULL, raw = FALSE)
```

Arguments

`x`	A raw vector, or a single string (converted to UTF-8 bytes).
`previous`	`NULL` (start a new checksum) or the raw checksum returned by a previous call with `raw = TRUE`.
`raw`	If `FALSE` (the default), return the checksum as a hexadecimal string. If `TRUE`, return it as a big-endian raw vector.

Details

Hardware instructions are used when the CPU supports them (SSE4.2, PCLMULQDQ and AVX-512 on x86-64, CRC and PMULL on arm64), otherwise a portable software implementation is used.

Value

A hexadecimal string, or with `raw = TRUE` a big-endian raw vector (length 4 or 8).

Examples

```
crc32c("123456789")
crc64nvme("123456789")
crc32c("123456789", raw = TRUE)

# Incremental
part1 <- crc32(charToRaw("1234"), raw = TRUE)
identical(crc32(charToRaw("56789"), part1), crc32("123456789"))
```

`checksums-file` *CRC checksums of a file*

Description

Like `crc32()`, `crc32c()` and `crc64nvme()` but reading the file in chunks, so files larger than memory can be checksummed.

Usage

```
crc32_file(path, previous = NULL, raw = FALSE, chunk_size = 1048576L)

crc32c_file(path, previous = NULL, raw = FALSE, chunk_size = 1048576L)

crc64nvme_file(path, previous = NULL, raw = FALSE, chunk_size = 1048576L)
```

Arguments

`path`	Path to a file.
`previous`	`NULL` (start a new checksum) or the raw checksum returned by a previous call with `raw = TRUE`.
`raw`	If `FALSE` (the default), return the checksum as a hexadecimal string. If `TRUE`, return it as a big-endian raw vector.
`chunk_size`	Number of bytes to read at a time. Default chunk size (1 MiB)

Value

A hexadecimal string, or with `raw = TRUE` a big-endian raw vector, see checksums.

Examples

```
f <- tempfile()
writeLines("hello", f)
crc32c_file(f)
unlink(f)
```

cpu_features *CPU features used for hardware acceleration*

Description

CPU features used for hardware acceleration

Usage

```
cpu_features()
```

Value

A named logical vector of the CPU features detected at runtime by the AWS Common Runtime. Features that do not apply to the current architecture are FALSE.

Examples

```
cpu_features()
```

Index

checksums, 2, *3*
checksums-file, 3
cpu_features, 4
crc32 *(*checksums*)*, 2
crc32(), *3*
crc32_file *(*checksums-file*)*, 3
crc32c *(*checksums*)*, 2
crc32c(), *3*
crc32c_file *(*checksums-file*)*, 3
crc64nvme *(*checksums*)*, 2
crc64nvme(), *3*
crc64nvme_file *(*checksums-file*)*, 3