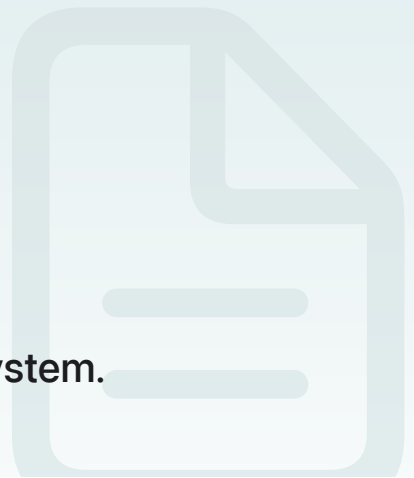


[Accelerate](#) / Compressing single files

Article

Compressing single files

Compress a single file and store the result on the file system.



Overview

In this article, you'll learn how to use `ArchiveArchive` to compress a single-source file, and write the compressed data to a file.

The code below compresses a file named `myFile.pdf` using the [Algorithm.lzfse](#) algorithm, and stores the result in a file named `myFile.pdf.lzfse`.

Create the file stream to read the source file

The [ArchiveByteStream](#) class provides static factory methods that create streams for different functions. In this case, use [fileStream\(path:mode:options:permissions:\)](#) to create a byte stream that reads the source file:

```
let sourceFilePath = FilePath(NSTemporaryDirectory() + "myFile.pdf")

guard let readFileStream = ArchiveByteStream.fileStream(
    path: sourceFilePath,
    mode: .readOnly,
    options: [ ],
    permissions: FilePermissions(rawValue: 0o644)) else {
    return
}

defer {
    try? readFileStream.close()
}
```

Create the file stream to write the compressed file

You also use `fileStream(path:mode:options:permissions:)` to create the file stream that writes the compressed file to the file system. In this case, use the `writeOnly` mode:

```
let archiveFilePath = FilePath(NSTemporaryDirectory() + "myFile.pdf.lzfse")

guard let writeFileStream = ArchiveByteStream.fileStream(
    path: archiveFilePath,
    mode: .writeOnly,
    options: [ .create ],
    permissions: FilePermissions(rawValue: 0o644)) else {
    return
}

defer {
    try? writeFileStream.close()
}
```

Create the compression stream

Create the compression stream, and specify the compression algorithm as `lzfse`. Specify the file-writing stream as the stream that receives the compressed data:

```
guard let compressStream = ArchiveByteStream.compressionStream(
    using: .lzfse,
    writingTo: writeFileStream) else {
    return
}

defer {
    try? compressStream.close()
}
```

Compress the source file

Finally, call `process(readingFrom:writingTo:)` to send the output of the file-reading stream to the compression stream. In turn, the compression stream sends its output to the file-writing stream:

```
do {
    _ = try ArchiveByteStream.process(readingFrom: readFileStream,
```

```
writingTo: compressStream)
```

```
} catch {  
    print("Handle `ArchiveByteStream.process` failed.")  
}
```

On return, `myFile.pdf.lzfse` exists in `NSTemporaryDirectory()` and contains the compressed contents of `myFile.pdf`.

See Also

Directories, Files, and Data Archives



Decompressing single files

Recreate a single file from a compressed file.



Compressing file system directories

Compress the contents of an entire directory and store the result on the file system.



Decompressing and extracting an archived directory

Recreate an entire file system directory from an archive file.



Compressing and saving a string to the file system

Compress the contents of a Unicode string and store the result on the file system.



Decompressing and Parsing an Archived String

Recreate a string from an archive file.