

# Package: mime (via r-universe)

July 24, 2024

**Type** Package

**Title** Map Filenames to MIME Types

**Version** 0.12.1

**Description** Guesses the MIME type from a filename extension using the data derived from /etc/mime.types in UNIX-type systems.

**Imports** tools

**License** GPL

**URL** <https://github.com/yihui/mime>

**BugReports** <https://github.com/yihui/mime/issues>

**RoxygenNote** 7.1.1

**Encoding** UTF-8

**Repository** <https://yihui.r-universe.dev>

**RemoteUrl** <https://github.com/yihui/mime>

**RemoteRef** HEAD

**RemoteSha** 1763e0dcb72fb58d97bab97bb834fc71f1e012bc

## Contents

guess_type . . . . .	2
mimemap . . . . .	3
parse_multipart . . . . .	3

<b>Index</b>	<b>5</b>
--------------	----------

---

 guess\_type

*Guess the MIME types from filenames*


---

## Description

Look up in the [mimemap](#) table for the MIME types based on the extensions of the given filenames.

## Usage

```
guess_type(
  file,
  unknown = "application/octet-stream",
  empty = "text/plain",
  mime_extra = mimeextra,
  subtype = ""
)
```

## Arguments

file	a character vector of filenames, or filename extensions
unknown	the MIME type to return when the file extension was not found in the table
empty	the MIME type for files that do not have extensions
mime_extra	a named character vector of the form <code>c(extension = type)</code> providing extra MIME types (by default, <a href="#">mimeextra</a> ); note this MIME table takes precedence over the standard table <a href="#">mimemap</a>
subtype	a character vector of MIME subtypes, which should be of the same length as file if provided (use an empty character string for a file if we do not want a subtype for it)

## Examples

```
library(mime)
# well-known file types
guess_type(c("a/b/c.html", "d.pdf", "e.odt", "foo.docx", "tex"))
# not in the standard table, but in mimeextra
guess_type(c("a.md", "b.R"), mime_extra = NULL)
guess_type(c("a.md", "b.R"))

# override the standard MIME table (tex is text/x-tex by default)
guess_type("tex", mime_extra = c(tex = "text/plain"))
# unknown extension 'bar'
guess_type("foo.bar")
# force unknown types to be plain text
guess_type("foo.bar", unknown = "text/plain")

# empty file extension
guess_type("Makefile")
```

```
# we know it is a plain text file
guess_type("Makefile", empty = "text/plain")

# subtypes
guess_type(c("abc.html", "def.htm"), subtype = c("charset=UTF-8", ""))
```

---

mimemap

*Tables for mapping filename extensions to MIME types*


---

### Description

The data `mimemap` is a named character vector that stores the filename extensions and the corresponding MIME types, e.g. `c(html = 'text/html', pdf = 'application/pdf', ...)`. The character vector `mimeextra` stores some additional types that we know, such as Markdown files (``.md``), or R scripts (``.R``).

### Source

The file `‘/etc/mime.types’` on Debian.

### Examples

```
str(as.list(mimemap))
mimemap["pdf"]
mimemap[c("html", "js", "css")]
# additional MIME types (not exported)
mime::mimeextra
```

---

parse\_multipart

*Parse multipart form data*


---

### Description

This function parses the HTML form data from a Rook environment (an HTTP POST request).

### Usage

```
parse_multipart(env)
```

### Arguments

`env`                    the HTTP request environment

### Value

A named list containing the values of the form data, and the files uploaded are saved to temporary files (the temporary filenames are returned). It may also be `NULL` if there is anything unexpected in the form data, or the form is empty.

**References**

This function was borrowed from <https://github.com/jeffreyhorner/Rook/> with slight modifications.

# Index

## \* datasets

mimemap, 3

guess\_type, 2

mimeextra, 2

mimeextra (mimemap), 3

mimemap, 2, 3

parse\_multipart, 3