### Package ‘beepr’

June 4, 2018

**Type**  Package  

**Title**  Easily Play Notification Sounds on any Platform  

**Version**  1.3  

**Encoding**  UTF-8  

**Date**  2018-06-01  

**Description**  The main function of this package is beep(), with the purpose to make it easy to play notification sounds on whatever platform you are on. It is intended to be useful, for example, if you are running a long analysis in the background and want to know when it is ready.

**License**  GPL-3  

**Imports**  stringr (>= 1.0.0), audio  

**RoxygenNote**  6.0.1.9000  

**NeedsCompilation**  no  

**Author**  Rasmus Bååth [aut, cre], Amanda Dobbyn [ctb]  

**Maintainer**  Rasmus Bååth <rasmus.baath@gmail.com>  

**Repository**  CRAN  

**Date/Publication**  2018-06-04 10:36:35 UTC  

### R topics documented:

- beep .......................................................... 2  
- beepr .......................................................... 3  
- beep_on_error .............................................. 4  

**Index**  

1
**Description**

`beep` plays a short sound which is useful if you want to get notified, for example, when a script has finished. As an added bonus there are a number of different sounds to choose from.

**Usage**

```r
beep(sound = 1L, expr = NULL)
```

**Arguments**

- `sound` character string or number specifying what sound to be played by either specifying one of the built in sounds, specifying the path to a wav file or specifying an url. The default is 1. Possible sounds are:
  1. "ping"
  2. "coin"
  3. "fanfare"
  4. "complete"
  5. "treasure"
  6. "ready"
  7. "shotgun"
  8. "mario"
  9. "wilhelm"
  10. "facebook"
  11. "sword"

  If `sound` does not match any of the sounds above, or is a valid path or url, a random sound will be played. Currently `beep` can only handle http urls, https is not supported.

- `expr` An optional expression to be executed before the sound.

**Details**

If `beep` is not able to play the sound a warning is issued rather than an error. This is in order to not risk aborting or stopping the process that you wanted to get notified about.

**Examples**

```r
# Play a "ping" sound
beep()

## Not run:
# Play a fanfare instead of a "ping".
```
beepr

beep("fanfare")
# or
beep(3)

# Play a random sound
beep(0)

# Update all packages and "ping" when it's ready
update.packages(ask=FALSE); beep()

## End(Not run)

---

**beep**

*Easily Play Notification Sounds on any Platform*

---

**Description**

This package contains one function, beep(), with one purpose: To make it easy to play notification sounds on whatever platform you are on. It is intended to be useful, for example, if you are running a long analysis in the background and want to know when it is ready.

**Details**

The package contains one main function **beep**, check it out to see what it does. For sound on Windows and MacOS **beep** depends on the **audio** package. For sound on Linux **beep** depends on that either the paplay utility from the Pulse Audio system, the aplay utility from the ALSA system, or VLC media player (http://www.videolan.org/vlc/index.html) is installed and on the PATH. Chances are that you already have one of these.

**Author(s)**

Rasmus Bååth < rasmus.baath@gmail.com >

**Examples**

# Play a "ping" sound
beep()
beep_on_error  

*Description*

`beep_on_error` wraps an expression and plays a short sound only if an error occurs.

*Usage*

```r
beep_on_error(expr, sound = 1)
```

*Arguments*

- `expr`: An expression to be evaluated for errors.
- `sound`: character string or number specifying what sound to be played by either specifying one of the built-in sounds, specifying the path to a wav file or specifying an url. The default is 1. Possible sounds are:
  1. "ping"
  2. "coin"
  3. "fanfare"
  4. "complete"
  5. "treasure"
  6. "ready"
  7. "shotgun"
  8. "mario"
  9. "wilhelm"
  10. "facebook"
  11. "sword"

If `sound` does not match any of the sounds above, or is a valid path or url, a random sound will be played. Currently `beep` can only handle http urls, https is not supported.

*Details*

If `beep` is not able to play the sound a warning is issued rather than an error. This is in order to not risk aborting or stopping the process that you wanted to get notified about.

*Examples*

```r
# Play a "ping" sound if \code{expr} produces an error
beep_on_error(log("foo"))

# Stay silent if \code{expr} does not produce an error
beep_on_error(log(1))
```
## not run:
# Play the Wilhelm scream instead of a ping on error.
beep_on_error(runif("bar"), "wilhelm")

## End(Not run)
Index

beep, 2, 3
beep_on_error, 4
beep, 3
beep-package (beepr), 3