

Package ‘codetools’

April 17, 2009

Version 0.2-2

Priority recommended

Author Luke Tierney <luke@stat.uiowa.edu>

Description Code analysis tools for R

Title Code Analysis Tools for R

Depends R (>= 2.1)

Maintainer Luke Tierney <luke@stat.uiowa.edu>

License GPL

Repository CRAN

Date/Publication 2009-03-08 16:10:19

R topics documented:

checkUsage	1
codetools	3
findGlobals	4
showTree	5
Index	6

checkUsage

*Check R Code for Possible Problems***Description**

Check R code for possible problems.

Usage

```
checkUsage(fun, name = "<anonymous>", report = cat, all = FALSE,
           suppressLocal = FALSE, suppressParamAssigns = !all,
           suppressParamUnused = !all, suppressFundefMismatch = FALSE,
           suppressLocalUnused = FALSE, suppressNoLocalFun = !all,
           skipWith = FALSE, suppressUndefined = dfltSuppressUndefined)
checkUsageEnv(env, ...)
checkUsagePackage(pack, ...)
```

Arguments

fun	closure.
name	character; name of closure.
env	environment containing closures to check.
pack	character naming package to check.
...	options to be passed to checkUsage.
report	function to use to report possible problems.
all	logical; report all possible problems if TRUE.
suppressLocal	suppress all local variable warnings.
suppressParamAssigns	suppress warnings about assignments to formal parameters.
suppressParamUnused	suppress warnings about unused formal parameters.
suppressFundefMismatch	suppress warnings about multiple local function definitions with different formal argument lists
suppressLocalUnused	suppress warnings about unused local variables
suppressNoLocalFun	suppress warnings about using local variables as functions with no apparent local function definition
skipWith	logical; if true, do not examine code portion of with expressions.
suppressUndefined	suppress warnings about undefined global functions and variables.

Details

`checkUsage` checks a single R closure. Options control which possible problems to report. The default settings are moderately verbose. A first pass might use `suppressLocal=TRUE` to suppress all information related to local variable usage. The `suppressXYZ` values can either be scalar logicals or character vectors; then they are character vectors they only suppress problem reports for the variables with names in the vector.

`checkUsageEnv` and `checkUsagePackage` are convenience functions that apply `checkUsage` to all closures in an environment or a package. `checkUsagePackage` requires that the package be loaded. If the package has a name space then the internal name space frame is checked.

Author(s)

Luke Tierney

Examples

```
checkUsage(checkUsage)
checkUsagePackage("codetools", all=TRUE)
## Not run: checkUsagePackage("base", suppressLocal=TRUE)
```

codetools

Low Level Code Analysis Tools for R

Description

These functions provide some tools for analysing R code. Mainly indented to support the other tools in this package and byte code compilation.

Usage

```
collectLocals(e, collect)
collectUsage(fun, name = "<anonymous>", ...)
constantFold(e, env = NULL, fail = NULL)
findFuncLocals(formals, body)
findLocals(e, envir = .BaseEnv)
findLocalsList(elist, envir = .BaseEnv)
flattenAssignment(e)
getAssignedVar(e)
isConstantValue(v, w)
makeCodeWalker(..., handler, call, leaf)
makeLocalsCollector(..., leaf, handler, isLocal, exit, collect)
makeUsageCollector(fun, ..., name, enterLocal, enterGlobal, enterInternal,
                   startCollectLocals, finishCollectLocals, warn,
                   signal)
walkCode(e, w = makeCodeWalker())
```

Arguments

<code>e</code>	R expression.
<code>elist</code>	list of R expressions.
<code>v</code>	R object.
<code>fun</code>	closure.
<code>formals</code>	formal arguments of a closure.
<code>body</code>	body of a closure.
<code>name</code>	character.
<code>env</code>	character.
<code>envir</code>	environment.
<code>w</code>	code walker.
<code>...</code>	extra elements for code walker.
<code>collect</code>	function.
<code>fail</code>	function.
<code>handler</code>	function.
<code>call</code>	function.
<code>leaf</code>	function.
<code>isLocal</code>	function.
<code>exit</code>	function.
<code>enterLocal</code>	function.
<code>enterGlobal</code>	function.
<code>enterInternal</code>	function.
<code>startCollectLocals</code>	function.
<code>finishCollectLocals</code>	function.
<code>warn</code>	function.
<code>signal</code>	function.

Author(s)

Luke Tierney

`findGlobals`*Find Global Functions and Variables Used by a Closure*

Description

Finds global functions and variables used by a closure.

Usage

```
findGlobals(fun, merge = TRUE)
```

Arguments

<code>fun</code>	closure.
<code>merge</code>	logical

Details

The result is an approximation. R semantics only allow variables that might be local to be identified (and event that assumes no use of `assign` and `rm`).

Value

Character vector if `merge` is true; otherwise, a list with `functions` and `variables` components.

Author(s)

Luke Tierney

Examples

```
findGlobals(findGlobals)
findGlobals(findGlobals, merge = FALSE)
```

`showTree`*Print Lisp-Style Representation of R Expression*

Description

Prints a Lisp-style representation of R expression. This can be useful for understanding how some things are parsed.

Usage

```
showTree(e, write = cat)
```

Arguments

`e` R expression.
`write` function of one argument to write the result.

Author(s)

Luke Tierney

Examples

```
showTree(quote(-3))  
showTree(quote("x"<-1))  
showTree(quote("f"(x)))
```

Index

*Topic **programming**

- checkUsage, 1
- codetools, 3
- findGlobals, 4
- showTree, 5

- checkUsage, 1
- checkUsageEnv (*checkUsage*), 1
- checkUsagePackage (*checkUsage*), 1
- codetools, 3
- collectLocals (*codetools*), 3
- collectUsage (*codetools*), 3
- constantFold (*codetools*), 3

- findFuncLocals (*codetools*), 3
- findGlobals, 4
- findLocals (*codetools*), 3
- findLocalsList (*codetools*), 3
- flattenAssignment (*codetools*), 3

- getAssignedVar (*codetools*), 3

- isConstantValue (*codetools*), 3

- makeCodeWalker (*codetools*), 3
- makeConstantFolder (*codetools*), 3
- makeLocalsCollector (*codetools*), 3
- makeUsageCollector (*codetools*), 3

- showTree, 5

- walkCode (*codetools*), 3