DESCRIPTION

DESCRIPTION

The description file explains the package, licencing, contact information and dependencies.

DESCRIPTION

DESCRIPTION

The description file explains the package, licencing, contact information and dependencies.

DESCRIPTION

DESCRIPTION

The description file explains the package, licencing, contact information and dependencies.

DESCRIPTION

DESCRIPTION

The description file explains the package, licencing, contact information and dependencies.

NAMESPACE

NAMESPACE

The namespace lists the functions exported by the package, and imports from dependencies.

NAMESPACE

NAMESPACE

The namespace lists the functions exported by the package, and imports from dependencies.

NAMESPACE

NAMESPACE

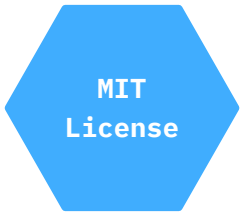
The namespace lists the functions exported by the package, and imports from dependencies.

NAMESPACE

NAMESPACE

The namespace lists the functions exported by the package, and imports from dependencies.

LICENSE (MIT)



Licenses set out how others can use the package, the MIT Licence is a ”permissive” licence.

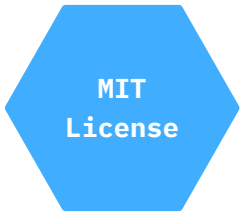
LICENSE (GPL)

Licenses set out how others can use the package, the GPL is a ”copyleft” licence.

LICENSE (CCO)

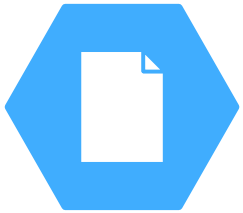
Licenses set out how others can use the package, the CC0 is a ”public domain” licence.

LICENSE (MIT)



Licenses set out how others can use the package, the MIT Licence is a ”permissive” licence.

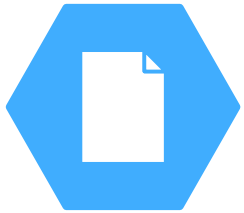
function.R



An R script contains your package code, for example a single function.

ACTION: Discard 10 units of code from your package

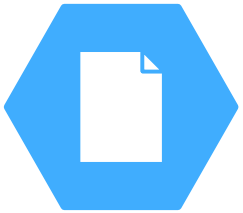
function.R



An R script contains your package code, for example a single function.

ACTION: Discard 10 units of code from your package

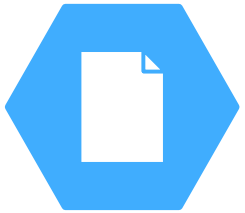
function.R



An R script contains your package code, for example a single function.

ACTION: Discard 10 units of code from your package

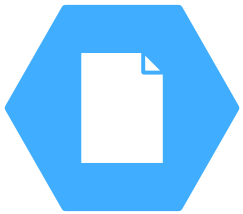
function.R



An R script contains your package code, for example a single function.

ACTION: Discard 10 units of code from your package

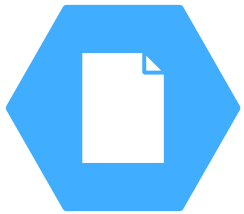
family.R



An R script contains your package code, for example a family of related functions.

ACTION: Discard 10 units of code from your package

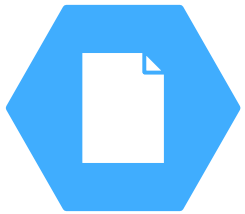
family.R



An R script contains your package code, for example a family of related functions.

ACTION: Discard 10 units of code from your package

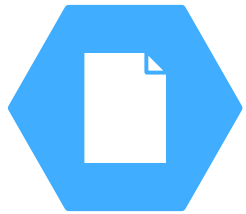
family.R



An R script contains your package code, for example a family of related functions.

ACTION: Discard 10 units of code from your package

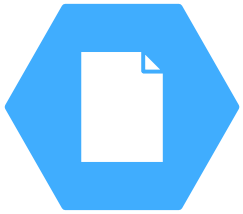
family.R



An R script contains your package code, for example a family of related functions.

ACTION: Discard 10 units of code from your package

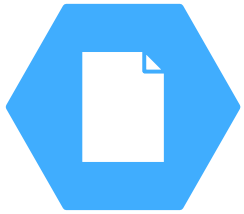
utils.R



An R script contains your package code, for example ”internal” utility functions.

ACTION: Discard 10 units of code from your package

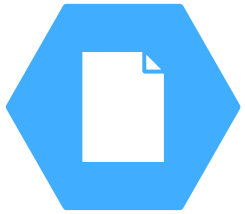
utils.R



An R script contains your package code, for example ”internal” utility functions.

ACTION: Discard 10 units of code from your package

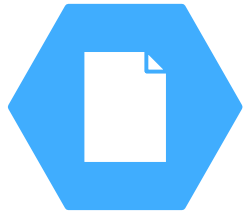
utils.R



An R script contains your package code, for example ”internal” utility functions.

ACTION: Discard 10 units of code from your package

utils.R



An R script contains your package code, for example ”internal” utility functions.

ACTION: Discard 10 units of code from your package

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

1 UNIT OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

2 UNITS OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

2 UNITS OF CODE

R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

2 UNITS OF CODE

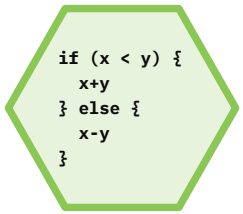
R function

```
if (x < y) {
  x+y
} else {
  x-y
}
```

Functions are the building blocks of R packages.

2 UNITS OF CODE

R function



Functions are the building blocks of R packages.

2 UNITS OF CODE

R function

if (x < y) {
x+y
} else {
x-y
}

Functions are the building blocks of R packages.

2 UNITS OF CODE

R function

if (x < y) {
x+y
} else {
x-y
}

Functions are the building blocks of R packages.

2 UNITS OF CODE

R function

if (x < y) {
x+y
} else {
x-y
}

Functions are the building blocks of R packages.

2 UNITS OF CODE

R function

if (x < y) {
x+y
} else {
x-y
}

Functions are the building blocks of R packages.

3 UNITS OF CODE

R function

if (x < y) {
x+y
} else {
x-y
}

Functions are the building blocks of R packages.

3 UNITS OF CODE

R function

if (x < y) {
x+y
} else {
x-y
}

Functions are the building blocks of R packages.

3 UNITS OF CODE

R function

if (x < y) {
x+y
} else {
x-y
}

Functions are the building blocks of R packages.

3 UNITS OF CODE

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Minor error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 1

Major error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 3

Major error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 3

Major error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 3

Major error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 3

Major error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 3

Major error

Oh dear! Your code doesn't work as expected

ERROR VALUE: 3

Fatal error

Oh dear! Your code doesn't work as expected

ACTION: Discard 1 code card each turn until this error is fixed

ERROR VALUE: 5

Fatal error

Oh dear! Your code doesn't work as expected

ACTION: Discard 1 code card each turn until this error is fixed

ERROR VALUE: 5

Fatal error

Oh dear! Your code doesn't work as expected

ACTION: Discard 1 code card each turn until this error is fixed

ERROR VALUE: 5

Fatal error

Oh dear! Your code doesn't work as expected

ACTION: Discard 1 code card each turn until this error is fixed

ERROR VALUE: 5

Imports

Dependencies allow you to use code in other packages. Imports must be installed for your package to run.

Imports

Dependencies allow you to use code in other packages. Imports must be installed for your package to run.

Imports

Dependencies allow you to use code in other packages. Imports must be installed for your package to run.

Imports

Dependencies allow you to use code in other packages. Imports must be installed for your package to run.

ACTION: Discard 1 code card from your package

1 UNIT OF CODE

Imports

Dependencies allow you to use code in other packages. Imports must be installed for your package to run.

ACTION: Discard 1 code card from your package

1 UNIT OF CODE

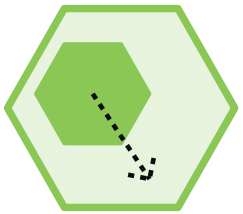
Imports

Dependencies allow you to use code in other packages. Imports must be installed for your package to run.

ACTION: Discard 1 code card from your package

1 UNIT OF CODE

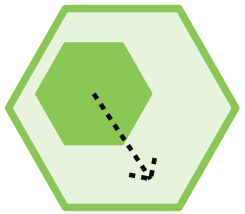
Suggests



Dependencies allow you to use code in other packages. Suggested packages don't need to be installed for your package to run.

ERROR VALUE: 1

Suggests



Dependencies allow you to use code in other packages. Suggested packages don't need to be installed for your package to run.

ERROR VALUE: 1

Function help

Documentation helps end users understand how to your package.

Function help

Documentation helps end users understand how to your package.

Function help

Documentation helps end users understand how to your package.

Function help

Documentation helps end users understand how to your package.

Function help

Documentation helps end users understand how to your package.

Function help

Documentation helps end users understand how to your package.

Function help

Documentation helps end users understand how to your package.

Function help

Documentation helps end users understand how to your package.

Vignette

Vignettes provide long-form guides to using aspects of your package.

Vignette

Vignettes provide long-form guides to using aspects of your package.

Vignette

Vignettes provide long-form guides to using aspects of your package.

Vignette

Vignettes provide long-form guides to using aspects of your package.

Debug minor error

MINOR FIX

Fixing an error is often called "debugging"

ACTION: **Discard error(s) up to a value of 1 from your package (and this card)**

ERROR VALUE: -1

Debug minor error

MINOR FIX

Fixing an error is often called "debugging"

ACTION: **Discard error(s) up to a value of 1 from your package (and this card)**

ERROR VALUE: -1

Debug minor error

MINOR FIX

Fixing an error is often called "debugging"

ACTION: **Discard error(s) up to a value of 1 from your package (and this card)**

ERROR VALUE: -1

Debug minor error

MINOR FIX

Fixing an error is often called "debugging"

ACTION: **Discard error(s) up to a value of 1 from your package (and this card)**

ERROR VALUE: -1

Debug minor error

MINOR FIX

Fixing an error is often called "debugging"

ACTION: **Discard error(s) up to a value of 1 from your package (and this card)**

ERROR VALUE: -1

Debug minor error

MINOR FIX

Fixing an error is often called "debugging"

ACTION: **Discard error(s) up to a value of 1 from your package (and this card)**

ERROR VALUE: -1

Debug minor error

MINOR FIX

Fixing an error is often called "debugging"

ACTION: **Discard error(s) up to a value of 1 from your package (and this card)**

ERROR VALUE: -1

Debug minor error

MINOR FIX

Fixing an error is often called "debugging"

ACTION: **Discard error(s) up to a value of 1 from your package (and this card)**

ERROR VALUE: -1

Debug major error

Fixing an error is often called ”debugging”

ACTION: Discard error(s) up to a value of 3 from your package (and this card)

ERROR VALUE: -3

Debug major error

Fixing an error is often called ”debugging”

ACTION: Discard error(s) up to a value of 3 from your package (and this card)

ERROR VALUE: -3

Debug major error

Fixing an error is often called ”debugging”

ACTION: Discard error(s) up to a value of 3 from your package (and this card)

ERROR VALUE: -3

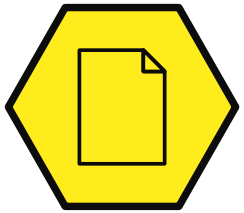
Debug major error

Fixing an error is often called ”debugging”

ACTION: Discard error(s) up to a value of 3 from your package (and this card)

ERROR VALUE: -3

Write tests

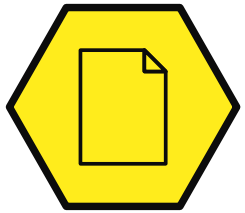


Using unit tests can help you detect errors more easily

ACTION: Discard error(s) up to a value of 1 from your package (one time only)

ERROR VALUE: -1

Write tests

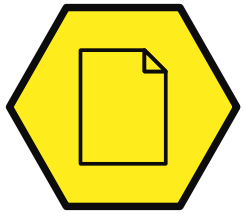


Using unit tests can help you detect errors more easily

ACTION: Discard error(s) up to a value of 1 from your package (one time only)

ERROR VALUE: -1

Write tests

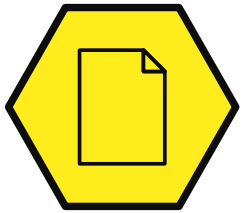


Using unit tests can help you detect errors more easily

ACTION: Discard error(s) up to a value of 1 from your package (one time only)

ERROR VALUE: -1

Write tests



Using unit tests can help you detect errors more easily

ACTION: Discard error(s) up to a value of 1 from your package (one time only)

ERROR VALUE: -1

Write tests

Using unit tests can help you detect errors more easily

ACTION: **Discard error(s) up to a value of 1 from your package (one time only)**

ERROR VALUE: -1

Write tests

Using unit tests can help you detect errors more easily

ACTION: **Discard error(s) up to a value of 1 from your package (one time only)**

ERROR VALUE: -1

Write tests

Using unit tests can help you detect errors more easily

ACTION: **Discard error(s) up to a value of 3 from your package (one time only)**

ERROR VALUE: -3

Write tests

Using unit tests can help you detect errors more easily

ACTION: **Discard error(s) up to a value of 3 from your package (one time only)**

ERROR VALUE: -3

Write tests

Using unit tests can help you detect errors more easily

ACTION: **Discard error(s) up to a value of 3 from your package (one time only)**

ERROR VALUE: -3

Write tests

Using unit tests can help you detect errors more easily

ACTION: **Discard error(s) up to a value of 3 from your package (one time only)**

ERROR VALUE: -3

Write tests

Using unit tests can help you detect errors more easily

ACTION: **Discard error(s) up to a value of 3 from your package (one time only)**

ERROR VALUE: -3

Write tests

Using unit tests can help you detect errors more easily

ACTION: **Discard error(s) up to a value of 3 from your package (one time only)**

ERROR VALUE: -3

Collaborate on code

Collaboration can help you develop your package more quickly

ACTION: **Use this card in place of 5 code cards (discard after use)**

5 UNITS OF CODE

Collaborate on code

Collaboration can help you develop your package more quickly

ACTION: **Use this card in place of 5 code cards (discard after use)**

5 UNITS OF CODE

Collaborate on code

Collaboration can help you develop your package more quickly

ACTION: **Use this card in place of 5 code cards (discard after use)**

5 UNITS OF CODE

Collaborate on fixing

Collaboration can help you develop your package more quickly

ACTION: **Discard error(s) up to a value of 5 from your package (discard after use)**

ERROR VALUE: -5

Collaborate on fixing

Collaboration can help you develop your package more quickly

ACTION: **Discard error(s) up to a value of 5 from your package (discard after use)**

ERROR VALUE: -5

Collaborate on fixing

Collaboration can help you develop your package more quickly

ACTION: **Discard error(s) up to a value of 5 from your package (discard after use)**

ERROR VALUE: -5

Collaborate on documentation

Collaboration can help you develop your package more quickly

ACTION: **Use this card in place of a documentation card**

Collaborate on documentation

Collaboration can help you develop your package more quickly

ACTION: **Use this card in place of a documentation card**