

# Build Your Own Package

This short article will introduce you how to build your own r-package by a step-by-step simple example illustration.

1. First, if you are using windows, make sure you have the tools you need:

- Perl: ActivePerl is recommended by Greg.
- Rtools.
- Microsoft HTML Help (hhc.exe) for windows. Y
- Latex

All above tools can be obtained by following the link in <http://www.murdoch-sutherland.com/Rtools/>

2. you need to edit your PATH variable as follows:

```
PATH=c:\Rtools\bin;c:\perl\bin;c:\Rtools\MinGW\bin;  
c:\htmlhelp;c:\R\bin;c:\texmf\miktex\bin;
```

where you will substitute appropriate directories for the ones listed above, but please keep the path in the same order as shown. Note that there is no space between different directories.

Sometimes you need add TMPDIR as an environment variable, pointing to, e.g., "C:\tmp".

3. Define two exemplary functions in R console.

- quadratic.equation:

```
quadrattic.equation=function(a,b,c)  
{  
  x.pos=(-b+sqrt(b^2-4*a*c))/(2*a);  
  x.neg=(-b-sqrt(b^2-4*a*c))/(2*a);  
  cbind(x.pos,x.neg)  
}
```

- is.right.triangle

```
is.right.triangle=function(a,b,c)  
{  
  return(c^2==a^2+b^2|a^2==c^2+b^2|b^2==c^2+a^2)  
}
```

Note: The above is.right.triangle function only works for integer values a,b,and c. For non-integer values, "==" will fail due to rounding and digital representation issues.

To solve this problem, try using the following function to do the comparisons instead of "==". It uses *all.equal()* which makes appropriate adjustments for numerical issues:

```
numericEqual=function(x,y)  
{  
  lenx=length(x);
```

```

    leny=length(y);
    longer=max(c(lenx,leny));
    shorter=min(c(lenx,leny));
    if(longer%%shorter!=0)
    warning("length of x and y do not match")
    x=rep(x,length=longer)
    y=rep(x,length=longer)
    retval=sapply(1:longer,function(i) isTRUE(all.equal(x[i],y[i])))
    retval
  }

```

and now the `is.right.triangle` function can be modified as

```

is.right.triangle=function(a,b,c)
{
  valid=(a>0) & (b>0) & (c>0); # check for valid length
  test.1=numericEqual(a^2+b^2,c^2);
  test.2=numericEqual(a^2+c^2,b^2);
  test.3=numericEqual(b^2+c^2,a^2);
  # now put it together. Return NA if invalid length
  retval=ifelse(valid,test.1|test.2|test.3,NA);
  retval
}

```

#### 4. Create Package Template

A folder with the name “trig” will be created under your current working directory by executing the following command in the R command window.

```
package.skeleton("trig",list=c("quadratic.equation","is.right.triangle","numericEqual"))
```

Note that you should usually tell R the path so you’ll be able to find the package after you’ve created it.

```
package.skeleton("trig",list=c("quadratic.equation","is.right.triangle","numericEqual"),
  path="c:\\Documents and Settings\\My Documents")
```

Otherwise R will create the package in whatever directory R is using. To find out where that is, use `getwd()`.

As you use the `package.skeleton()` command, the help files should have been created automatically. If you want to create separate help files, use the following commands:

```

prompt("trig")
prompt("quadratic.equation")
prompt("is.right.triangle")
prompt("numericEqual")

```

Remember to move these files into the package's "\man" directory.

#### 5. Edit the Help Files

Edit .Rd files if necessary and do not forget the DESCRIPTION file.

Note that these files are not syntactically correct as they are originally generated. For instance, you will need to replace or remove any line containing “~~”, otherwise R CMD check will complain if some fields are not modified.

#### 6. Check the Package

To check the package, go to the console (not R console), change your directory to just above the package folder, and run the following command

```
R CMD check trig
```

Note any errors or warnings generated here and fix them.

#### 7. Install the Package

Stay in the console, and submit the following command

```
R CMD INSTALL trig
```

Then the package trig will be automatically installed in the library folder of your R system.

#### 8. Load the Package

You can now try out your new R package using “library(trig)” in your R console.

## Call C or Fortran in R

1. First, create a file named `quad.c` with the following contents under, say, `c:` drive.

```
#include <math.h>
void quad(double *a, double *b, double *c, double *xpos, double *xneg, int *length)
{
  int i;
  for(i=0;i<=*length; i++)
  {
    xpos[i]=(-b[i]+sqrt(b[i]*b[i]-4.0*a[i]*c[i]))/(2*a[i]);
    xneg[i]=(-b[i]-sqrt(b[i]*b[i]-4.0*a[i]*c[i]))/(2*a[i]);
  }
}
```

2. If you just want to use C or Fortran codes as dynamical library, build the shared library first by using the following command under C directory

```
R CMD SHLIB quad.c
```

now you can manually dynamically load this library and run it by using the command in R

```
dyn.load("c:/quad.dll")
```

or in unix, use

```
dyn.load("quad.so")
```

Then, we can submit the following codes to do the computation.

```
a=c(1,1); b=c(9,4); c=c(0,0);
.C("quad",as.double(a),as.double(b),as.double(c),as.double(1:2)
,as.double(1:2),,as.integer(length(a)))
```

3. If you want to build the C or Fortran codes into this package, follow these steps:

- Copy the source file to “`\src`” directory (you need create this directory under the package first).
- Create the trig file “`First.lib.R`” under “`\R`” with the following contents

```
.First.lib=function(lib, pkg)
  {library.dynam("trig", pkg, lib)}
```

This `.First.lib()` function will ensure that the `.DLL` or `.so` is properly loaded when you load the library.

- Now you need to modify the `quadratic.equation()` function to call the C code instead of performing the calculation directly. The file should be

```
quadratic.equation=function(a,b,c)
{
  s=C("quad",
      a=as.double(a),
      b=as.double(b),
      c=as.double(c),
      positive.x=double(length(a))
      negative.x=double(length(a))
      len=as.integer(length(a)),
      PACKAGE="trig")
  cbind(s$positive.x,s$negative.x)
}
```

Now, run

```
R CMD INSTALL trig
```

again, this time, the `.c` file will be automatically compiled and library will be generated. It should now work just like other packages.