Prepare CPU

## Step 0: Who needs this document?

This documentation is relevant to all users setting up a personal or government-/company-owned CPU for R/RStudio for the first time.

### *Do you have administrative control over your CPU?*

**I DO NOT Have Administrative Access on My CPU**

Installation of R requires administrative access privileges on your CPU. In general, most government- or company-owned CPUs do not grant administrative access to employees. Consequently, you will need to involve your local IT specialist when installing R, as well as some of the other required software outlined below. Ensure that YOU log on as the USER, download the R executable and have your IT specialist use their administrative control (i.e., right–click, Run as administrator in W10 OS) to install R; otherwise, the R library may be created under administrative control and various R/RStudio functions will not operate properly.

**I DO Have Administrative Access on My CPU**

Simply follow the step below.

### *Do you access files on a computer network?*

**I DO Access Files on a Computer Network**

When working on a government- or company-owned CPU, it is often the case that you will have access to network drives. These drives are shared across the computer network, and typically provide users with the same copy, edit, upload, and delete privileges you would have on a local drive. Unfortunately, R and RStudio do not speak SMB (samba, cifs, etc.), but your CPU will default to provide a URI with this structure: //<servername>/path/to/file. Thus, by default, we ask R to speak to a windows file sharing protocol (which it cannot do) to a host named “<servername>”, and then to navigate to the respective subdirectory. By ignoring the details in the steps below, you *will* run into errors because of this network-quirk – R/RStudio will not run properly.

**I Am Working on a Personal CPU With No Network Connections**

Working from a personal CPU simplifies the set-up process. R and RStudio will typically suggest correct locations for unpacking, installing, and storing software and packages.

### *How do I move forward?*

Follow the steps below in full, taking note of differences between set-up on personal CPUs and networked CPUs, and R/RStudio should function properly on your OS/CPU/instance.

# Step 1: Access and Install R

If you do not already have R installed on your CPU, follow this [link](https://www.r-project.org/), which will take you to **The R Project for Statistical Computing** home page.Having multiple versions of R installed on the same CPU is perfectly fine; to ensure we are on the same page, download 4.1.2 or greater.

For Windows users:

* Click [here](https://cran.r-project.org/bin/windows/base/old/4.1.2/) to access the latest release of R from CRAN.
  + Next, click the **R-4.1.2-win.exe** file for installation; run the downloaded .exe file and follow the messages prompted in the download. (Maintain *all* defaults, including the Path for installation, to your C: drive)

For MAC Operating System users:

* Click [here](https://cloud.r-project.org/bin/macosx/) to access the MAC Operating System CRAN site and carefully read the guidance.
  + Scroll down until you find the **R–4.1.3.pkg** link to download; run the downloaded .exe file and follow the messages prompted in the download.

### *Check the R Version*

Once R has been installed, open R, and copy, paste, and run the code below in the console. It will return whichever version of R you install.

# check R version after install of R  
print(paste("R", getRversion())) # obtain current R version if not known

## [1] "R 4.1.2"

If you need to update your version of R, you can use the {installR} package to do that update from within RStudio.

# install {installR}, check newest version and install a newer version if it exists  
install.packages(“installR”) # install the package

library(installR) # call the package library into your session

updateR() # use the updateR() command to update R

# Step 2: Access and Install RStudio

Access RStudio [here](https://rstudio.com/products/rstudio/download/#download). Scroll down to “All Installers” and select the appropriate version for your CPU Operating System (Mac v. Windows); run the downloaded .exe file and follow the messages prompted in the download. Maintain all defaults.

Rebooting at this point is **highly** recommended; close everything and restart your computer.

# Step 3: Installation of Rtools

The next steps are *only* relevant to Windows systems. If you are using a MAC, skip this section and head to **Step 4.**

### *Rtools*

Rtools is a collection of software needed to build R packages from the source. Because we are using R-4.1.2 you MUST install [rtools40-x86\_64.exe](https://cran.r-project.org/bin/windows/Rtools/history.html); run the downloaded .exe file and follow the messages prompted in the download. Maintain all defaults.

# Step 4: Define R\_LIBS\_USER, R\_USER, and .libPaths()

Prior to package installation, NSO employees working on a network mapped drive must properly define their USER environment.

### *Find R\_USER*

Open RStudio. Run the following code to determine the location of your folder:

# find Home folder  
Sys.getenv("R\_USER")

If you are on a *personal* CPU, this will return something like: C:/Users/<account>. If you are on a *network*, this will return something like: //<domain>/files/UserData/<office>/<account>/Documents. In the latter case, you must properly define R\_USER.

### *Properly define R\_USER*

Next, run the following command to set the Home folder.

# define your value based on your account location in the network drive   
Sys.setenv(R\_USER = 'C:/Users/<account> ')

Now when you run Sys.getenv("R\_USER") you should see the path you defined, above. This step allows R to properly access the pandoc.exe file for building output.html files.

### *Find R\_LIBS\_USER*

Run the following code to determine your current user library environment:

# find User environment  
Sys.getenv("R\_LIBS\_USER")

If you are on a personal CPU, this will return something like BLANK. If you are on a network computer, this will return something like: //<domain>/<VDI>/<office>/<account>/Documents.

### *Properly define R\_LIBS\_USER*

Run the following command to set your library:

# define your value based on your account location in the network drive   
Sys.setenv(R\_LIBS\_USER = 'C:/Users/<account>/R/win-library/4.1')

### *Find .libPaths()*

Next, run the following command to determine the path in which R wants to build your package lbirary:

# find library paths  
.libPaths()

If you are on a personal CPU, this will return something like BLANK. If you are on a network computer, this will return something like:

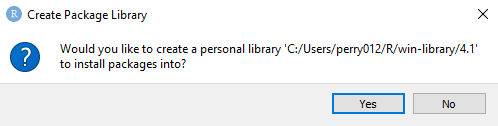
[1] "\\\\<domain>/<VDI>/<office>/<account>/Documents/R/win-library/4.1"

[2] "C:/Program Files/R/R-4.1.2/library"

In the latter case, run the following code to prompt R to build a library in the C: drive under your username:

# define new library path  
.libPaths("C:/Users/<account>/R/win-library/4.1")

Once you have run this code, go to Tools, Install Packages. If you are installing packages for the first time R will return one (possibly two) queries asking if you wish to build a personal library.



<account>

Click **YES** on the message, indicating you wish to build a personal R library. This is where all of your downloaded packages will populate. If it is not your first time downloading packages, R will simply move the location of your library to the path specified in the previous line of code.

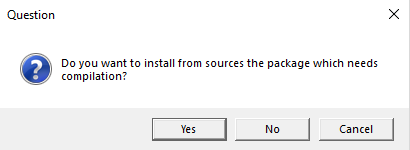
This library will be located at C:/Users/<account>/R/win-library/4.1 for Windows and /Library/R.framework/Versions/4.X/Resources for MAC Operating Systems.

Step 5: Install R Package {devtools}

Open RStudio and run the code below to install {devtools}.

# install package devtools  
install.packages("devtools", dependencies = T, repos = "https://cloud.r-project.org/")

When prompted with this message:



Click **YES**.

Once the packages have successfully unpacked (i.e. the code stops running), run the code below to determine if {devtools} links with Rtools. A response of *TRUE* indicates Rtools is operational.

# load library devtools and test for Rtools  
devtools::find\_rtools() # returns TRUE if Rtools if operational

## [1] TRUE

By following these instructions in full, your CPU should be set up to work in R/RStudio successfully.