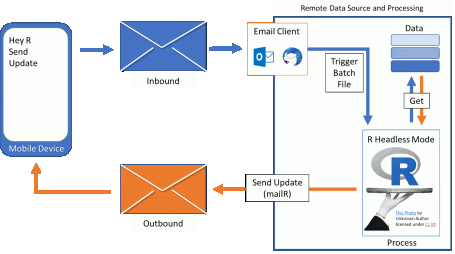
Have you ever been on the road and wished you could run an R script from your mobile device and see the results? Maybe you’re a business person who needs a quick update on a project or production schedule. Or, possibly you need an up-to-the-minute report out for a meeting, and you don’t have a “cloud based solution to get you the important data. In short, there is data you want and can’t get because you’re not onsite to run an R script. In this post, I’ll show you how to resolve this issue using headless R and E-mail triggers.

**Objectives:** Run an R script from a mobile device

1. [Review the Overall Process](https://r-bar.net/r-scripts-mobile-device-email-triggers/#TheIdea)
2. [Checkout a DEMO Scenario to Test our Process](https://r-bar.net/r-scripts-mobile-device-email-triggers/#remoteData)
3. [Load DEMO Data into R](https://r-bar.net/r-scripts-mobile-device-email-triggers/#Data2R)
4. [E-mail Processed DEMO Results to our Mobile from R](https://r-bar.net/r-scripts-mobile-device-email-triggers/#mailR)
5. [Setup a Batch File and Headless R](https://r-bar.net/r-scripts-mobile-device-email-triggers/#batchHeadless)
6. [Create an E-mail Trigger to Run the Batch File](https://r-bar.net/r-scripts-mobile-device-email-triggers/#emailRules)
   1. [Outlook](https://r-bar.net/r-scripts-mobile-device-email-triggers/#Outlook)
   2. [ThunderBird](https://r-bar.net/r-scripts-mobile-device-email-triggers/#ThunderBird)

**The General Idea**

This is a big article, so here is quick summary of what we are going to do to get our data on demand on our mobile. First, let's pretend your mobile e-mail address is [myMobile@home.com](mailto:myMobile@home.com). From your mobile, you send an E-mail to your desktop machine [mydesktop@work.com](mailto:mydesktop@work.com) with the subject "Hey R – Send Update”. Your ever vigilant desktop e-mail client receives the e-mail, and runs a “rule” you setup before leaving the office. That rule will launch an application. For us, that application will be a simple batch file or shell script. The batch file will do two things i) launch headless R and ii) call the R script you want to run. When the R script runs, it will get the data from the target data source, process the data, and e-mail the result back to your mobile device. Here is a little flow chart that sums it up:



**Process Benefits**

* You get the data on demand. No script spamming your email account.
* Headless R only runs when it’s called.

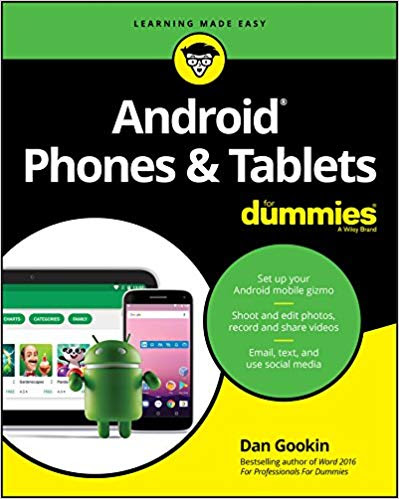
**Things to Consider**

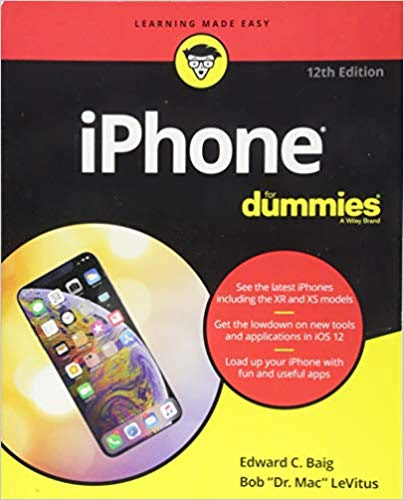
* Your desktop e-mail client needs to be open and receiving e-mails
* There maybe some security considerations depending on how you implement this. Talk with IT.

Great! If you've got the idea, let's get into the details.

**E-mails from the Mobile Device**

If you know how to send an e-mail from your mobile device, you are off to a great start. You'll need to pick something unique for your subject line like “Hey R – Send Update”, “Hey R – Run Report”, or something of that sort. If you're not sure how to send an email from your mobile device, perhaps you'll find these texts useful:

[[](https://amzn.to/2XvBqdv)](https://amzn.to/2XvBqdv" \t "_blank)

[](https://amzn.to/2tMky4s)

**The Remote Data Source**

The remote data source is the data you'd like access on the road but can't using your mobile device. Here are a few important notes about the remote data source:

1. It needs to be accessible with an R script on your desktop and
2. the machine running that R script needs a functioning e-mail client like Outlook or ThunderBird.

To keep us all on the same page for this post, I've created a demo data source which can be found [here](https://apps.r-bar.net/PID1025/index2.php). We are going to **pretend** that our mobile devices cannot access this. Looking at the data source, you'll see it's a simple HTML page showing the current production status of some widget. There are two tables in the data source, the “Product Overview” and the “Work In Progress”. Both of these tables update throughout the day as work gets done. If HTML iframes are working, you should see the contents of the data source below. If not, check the data source out [here](https://apps.r-bar.net/PID1025/index2.php).

**Summary of the Table Items**

**Product Overview**

**Today’s Builds:** These are the products in queue for today  
**Shipping Today’s Builds:** The products that will be complete by the shipping deadline at 5pm EST.  
**Shipping Yesterday’s Builds:**Products that missed yesterday’s shipping deadline and will ship today.  
**Potentially Shipping Tomorrow:** Today’s builds that aren’t in the shipping queue yet.  
**Tomorrow’s Orders:** What have customers ordered for tomorrow.

**Work In Progress**

This table has two rows. The “Building Product” row tells you which product ID is currently being built and how much time remains. If a product is in the packaging process, it will list its product ID and how much time remains.

**Remember:** We are pretending that we can see this data with a desktop device but not our mobile device.

**Pre-Processing the Data for Our Mobile Report**

Great, we have a data source and we are pretending that it can't be directly accessed with our mobile device. But, we can process it with our desktops using R. So, suppose we are on the road, and we really need to know the following: **As of right now, how much product will be shipping today and how much time is there before the shipping deadline at the New York site?**

Using our desktops, we generate the following code which i) scrapes the data from the data source, ii) sums the appropriate data, and iii) makes a short text summary:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | # Parse the HTML  **require**(rvest) #package to parse HTML  **page** <- read\_html("<https://apps.r-bar.net/PID1025/>") #Get the Page  tbls <- html\_nodes(**page**, "table") #Get the Tables  Product\_Status <- html\_table(tbls[1])[[1]] #Get First Table    #Detemine how much product is shipping today as of right now  ToShip <- **grep**(pattern = "Shipping Yesterday|Shipping Today",  x = Product\_Status$`Product Metrics`)  Shipping\_Product <- **sum**(Product\_Status$Count[ToShip])    #Detemine how many minute before the shipping dead line  MinLeft <- **gsub**(pattern = '**\\**D',  replacement = "" ,  Product\_Status$`Product Metrics`)[[2]]    #The summary message to be e-mailed  email.msg <-  paste0(Shipping\_Product, " units of product shipping as of ", **date**(),  ". There is ", MinLeft,  " minutes remaining before today's shipping deadline.") |

The results should look something like:

28 units of product shipping as of Fri Mar 01 16:55:23 2019. There is 5 minutes remaining before today’s shipping deadline.

**E-mailing the Data from R to our Mobile Device**

Awesome, we have the data in R and we generated the message we want to send. Now, we need to send it to our mobile device using R. To make this work, you will need to have the following information on hand:

* SMTP server for outgoing e-mail
* The SMTP server port
* Username
* Password

If you are not sure about these items, check with IT or your internet service provider (ISP). Using this information and R's mailR package, we can generate some code to send our message from the previous section to our mobile device via e-mail. Here is some example code.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | #Email Trigger when "Hey R - Send Update"  **library**(mailR)  sender <- "[mydesktop@work.com](mailto:mydesktop@work.com)" #The email address R will use to send  recipients <- **c**("[myMobile@home.com](mailto:myMobile@home.com)") #your mobile email here  send.mail(from = sender,  to = recipients,  subject = "Requested Status Report",  **body** = email.msg, #your summary message  smtp = **list**(host.name = "your.smtp.server",#check with IT or ISP  port = 587, #port number (IT or ISP)  user.name = "your.username", #likely your email address passwd = "your.password", #email password  ssl = TRUE #if using secure connection  ),  authenticate = TRUE,  send = TRUE) |

You will need to update the sever and account information as appropriate. If everything is working you should be able to see the results using your mobile after tuning this code. If it doesn't work, check your server details and email addresses.

**Batch Files and Headless R**

If you were able to receive the e-mail on your mobile device using R's mailR package discussed in the previous section, you are ready for this next step. In this section, we are going to do two things:

1. Show how to start R in headless mode
2. Execute a R script in headless mode using a batch file.

**Running R in Headless Mode**

To run R in headless mode, you need to locate a file called Rscript.exe. To do this, go to your “program files” folder. Find the R sub-folder. Locate the current version of R. Open up the bin folder. For me, the full path looks something like this:

C:\Program Files\R\R-3.5.1\bin

Within this folder, you will find a file called Rscript.exe. This is the file you will need to run R in headless mode. So, for me the full path to this executable is C:\Program Files\R\R-3.5.1\bin\Rscript.exe

**Run an R script in Headless Mode**

The Rscript we are going to run headless is our E-mail script described above. Here is the complete script if you haven't saved it already:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | # Parse the HTML  **require**(rvest) #package to parse HTML  **page** <- read\_html("<https://apps.r-bar.net/PID1025/>") #Get the Page  tbls <- html\_nodes(**page**, "table") #Get the Tables  Product\_Status <- html\_table(tbls[1])[[1]] #Get First Table    #Detemine how much product is shipping today as of right now  ToShip <- **grep**(pattern = "Shipping Yesterday|Shipping Today",  x = Product\_Status$`Product Metrics`)  Shipping\_Product <- **sum**(Product\_Status$Count[ToShip])    #Detemine how many minute before the shipping dead line  MinLeft <- **gsub**(pattern = '**\\**D',  replacement = "" ,  Product\_Status$`Product Metrics`)[[2]]    #The summary message to be e-mailed  email.msg <-  paste0(Shipping\_Product, " units of product shipping as of ", **date**(),  ". There is ", MinLeft,  " minutes remaining before today's shipping deadline.")    #Email Trigger when "Hey R - Send Update"  **library**(mailR)  sender <- "[mydesktop@work.com](mailto:mydesktop@work.com)" #The email address R will use to send  recipients <- **c**("[myMobile@home.com](mailto:myMobile@home.com)") #your mobile email here  send.mail(from = sender,  to = recipients,  subject = "Requested Status Report",  **body** = email.msg, #your summary message  smtp = **list**(host.name = "your.smtp.server",#check with IT or ISP  port = 587, #port number (IT or ISP)  user.name = "your.username", #likely your email address passwd = "your.password", #email password  ssl = TRUE #if using secure connection  ),  authenticate = TRUE,  send = TRUE) |

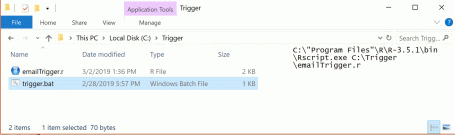
Make sure you have updated the email addresses, SMTP sever, port, username and password details appropriately (lines 25,26, 31:33). Save this script to a file called emailTriggers.R in a folder called “trigger” at c:. You will need to create the trigger folder. After you have saved the file, make sure you can find it at c:\Trigger\emailTriggers.R .

Awesome! From here, we are going to make the batch file. If you are not familiar with batch files, they are little shell scripts for the windows operating system. I first encountered them before windows was a thing… back in the DOS days. I digress…

All we are going to do with our batch file is call headless R and tell it to run our emailTrigger.R script. Making a batch file is simple. Open up notepad or notepad++ and copy the following code:

C:”Program Files"\R\R-3.5.1\bin\Rscript.exe C:\Trigger\emailTrigger.r

Simple, right? The first part calls headless R; the second part calls the R script. Now, save the batch file in the c:\trigger folder, and call it trigger.bat . Once you are done, the contents of c:\Trigger should look like this:



Great. Now execute the trigger.bat file. To execute the file, you can use the window's file explorer or the command prompt. If you used the file explorer. Go to c:\trigger and double click on the trigger.bat file. If the script runs successfully, you should get an e-mail with the most current results from the DEMO site. Make sure this works before moving onto the next section.

**The Email Rules and Triggers**

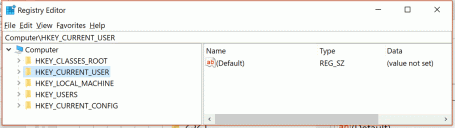
This section assumes you were able to run the batch file and get the newest DEMO site results sent to your mobile device's e-mail. If so, you are at the last step. On your desktop with the trigger.bat file, you need to have either Outlook or ThunderBird installed. Using these email clients, we are going to make a rule that does the following: When an e-mail comes in with the subject “Hey R – Send Update”, and is from [myMoble@home.com](mailto:myMoble@home.com) launch the trigger.bat file you created in the previous section.

**Outlook**

I'm assuming you already have Outlook installed and your desktop e-mail setup. If this is true, we want to make a new rule. In the newer versions of Outlook, you can't make a rule to launch an application without making a minor change to the window's registry. Here is what you need to do to make the change:

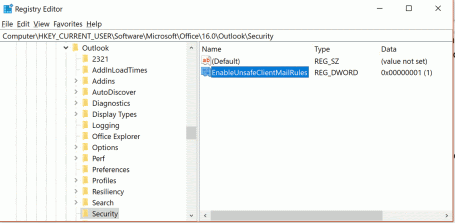
**Edit Registry for Outlook Application Launch**

1. Press the Windows Key + R: This will open the run dialogue box
2. In the “open” text box type: regedit
3. You’ll get an “are you sure you want to use this application” warning from windows. Press OK…
4. The registry editor should look like this:



Cool we got the registry editor open. Here are the changes we need to make

1. In the left-hand panel drill down to:  
   **Computer\HKEY\_CURRENT\_USER\Software\Microsoft\Office\16.0\Outlook**
2. Make a new folder (key) called “Security” by right clicking on the Outlook folder and selecting “New”→“Key”
3. Enter the new “Security” folder. Full path should now be:  
   **Computer\HKEY\_CURRENT\_USER\Software\Microsoft\Office\16.0\Outlook\Security**
4. Create a new DWORD(32bit) value in the right panel by right clicking, call it “EnableUnsafeClientMailRules”
5. Double click on EnableUnsafeClientMailRules and set the data value to 1.
6. Should look like this when you are done:

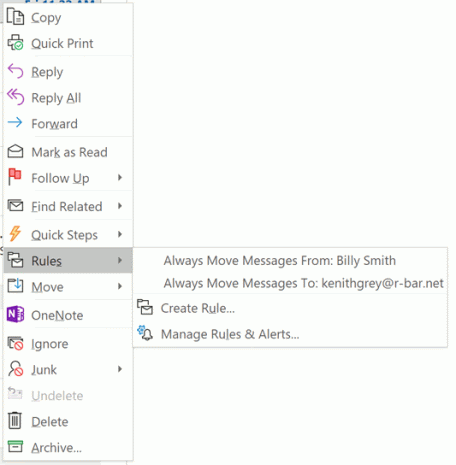


All done.

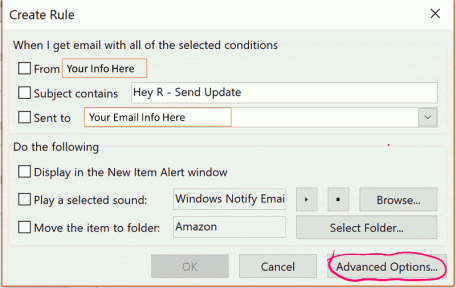
**Outlook Email Rules**

With the registry updated. Restart Outlook. Now let's make the rule to launch the trigger.bat file when we receive an e-mail from [myMobile@home.com](mailto:myMobile@home.com) with the subject line “Hey R – Send Update”

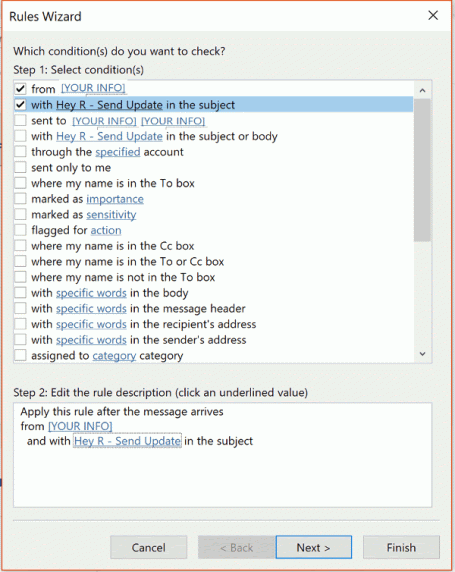
1. Send an email from your mobile device to your desktop with the subject line “Hey R – Send Update”
2. Find the e-mail you sent in Outlook
3. Right click on the message and go to Rules→Create Rule



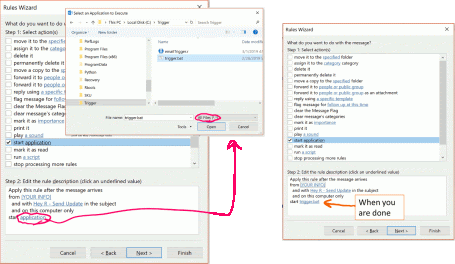
1. Select advanced from the Create Rule Dialogue Box



1. **First screen** – Set rule conditions
   1. Select the “From [your info]” and
   2. “With Hey R – Send Update in the subject”
   3. Press Next



1. **Second screen** – set the action if the previous conditions were met
   1. In the Step 1: Select action box, select “start application”. This option won’t be available if you haven’t made the windows registry change discussed above.
   2. In the Step 2: Edit the Rule Description… Click the word “application”
   3. An “open file” dialog box will appear. Find the file c:\Trigger\trigger.bat (to see the file, you will need to change the select box from “executable files (\*.exe)” to “All files (\*.\*)” ) Click the file and press “open”.
   4. Press next to move to the third screen



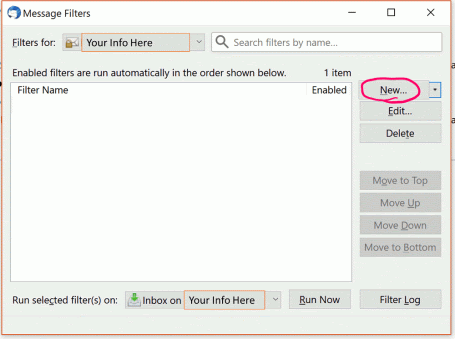
1. **Third screen** – set exceptions to the rules. We aren't going to worry about this right now.
   1. Press Next
2. **Last Screen** – Finish Rule
   1. Step 1: Give your Rule a name
   2. Step 2: Make sure the rule is turned on
   3. Press Finish

Great the rule is all setup. Now send an e-mail to your desktop with the subject line “Hey R – Send Update” from your mobile device to make sure the batch file gets launched. On the desktop, you should see a command prompt open briefly with some R output. If your mobile gets an e-mail back with the result from R, your setup is working. Nice job.

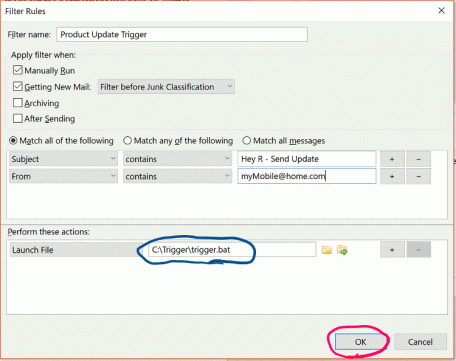
**ThunderBird**

I'm assuming you already have ThunderBird installed and your desktop e-mail setup. If this is true, we need to install a plugin called [FiltaQuilla](https://addons.thunderbird.net/en-US/thunderbird/addon/filtaquilla/), so we can make some e-mail rules. To install the plugin, open ThunderBird and update your mail box as necessary. Press the ALT key on your keyboard. At the top of the ThunderBird window, the program menu should appear. Click on “Tools→Add-ons ”. This will open a new tab within ThunderBird called the “Add-ons Manager”. You will want to find a search box… somewhere. To find one, I had to click on one of the many “See all” links scattered across the page. Clicking one of the “See all” links will open another tab called “Up & Coming Extensions”. Next, find the search for add-ons text box. Search for FiltaQuilla and install it. When you are done, you'll need to restart ThunderBird.

After restarting, press the ALT key on your keyboard to bring up the program menu. Then, click on tools, and select “Message Filters”. The following window will pop up:



Press the “new”“ button and fill in the rule details as shown:



As shown above, when a new message comes in the subject will be checked for the text "Hey R – Send Update” and the sender's e-mail will be checked for the appropriate senders address which will be your mobile e-mail address. If these items match, launch the batch file we made in the previous section. This will launch headless R and your script which will use the mailR package to send your mobile the most up-to-date results on the DEMO site.

After pressing OK, send an e-mail to your desktop with the subject line “Hey R – Send Update” from your mobile device to make sure the batch file gets launched. You should see a command prompt open briefly with some R output. If you mobile gets an e-mail back with the result, your setup is working.

**Summary**

Sometimes we need data from a home or office data source, and we can't access it directly with our mobile devices. In this post, you learned how to get at this data by launching an R script on your desktop using an e-mail trigger from your mobile device. Now, if you need to get to that data. All you need to do is follow three easy steps

1. Write an R script on the home or office desktop that grabs the data and e-mails the results to your mobile device.
2. Setup a batch file to have headless R call your script.
3. Setup an e-mail rule to launch your batch file when it receives an e-mail with the subject “Hey R – Send Update”