Smart Programming
Mark Friedman 2024-12-10

Think Long Term

How long does this thing have to last?

We are often asked to write programs or create systems, and an important consideration is its life expectancy. It is time well spent considering the long-term implications when writing something. For a system, the question is, do I need to make it last multiple cycles such as over several yearly cycles, or versions of a system it interacts with. Even when writing a piece of code or function, it pays to ask, "is this a one-off or can it lead to something better?" I have often found myself writing some small thing, but realizing that if it was generic, it could be used in many other situations.

I'll provide an example from some volunteer work I do for the Soroka Clinical Research Center/Negev Environmental Health Research Institute. We often receive spreadsheets or Microsoft Access files with a mix of Hebrew or English text in the variable names and data. Working with the Hebrew character set in an English language-based environment like SAS or R can drive you nuts. The cursor will sometimes jump around when typing certain characters like punctuation or using the cursor and getting quotes in the right place will make you want to cry.

It also is inefficient to manually type statements like this for large numbers of variables:

```
mutate(newvar=case_when (var == "אלפ"  ~ 1,
                         var == "ביטא" ~ 2,
                         TRUE          ~ NA))
```

This was my first time facing this issue and I tried endlessly to come up with a way to write code in multiple languages. I even tried writing the code in an external editor and doing copy/paste (spoiler alert: it didn't always work). I took a step back, and knowing this will happen in future projects, realized that the best approach is to develop a 'generic' process that can output what needs to be translated/recoded/corrected, and then input a file that has the corrections and apply them. This approach works for all situations regardless of language or purpose (translation, recoding, cleaning, etc.).

Workbooks are easy for people to use, so it became a 'no-code' approach. You give someone a spreadsheet and say, type what you want these things to mean. You then process that spreadsheet and it's done. An added benefit: because the metadata is structured as tables with rows and columns, you can use the information later to do other things (like create formats in SAS). Big warning: workbooks are lousy for ensuring the 'type' of a column is consistent. One non-number in a column of numbers and the column imports as character not number, so always QC or force the type explicitly.

An example of a tab in the generated worksheet is shown below.

We can do recodes and corrections. In the following list of districts, we want to replace long text strings with short codes. It is clear that some of the values are just slightly different versions of the same thing.

Freq	OriginalValue	RevisedValue	Comment
6	אשקלון		
3	חיפה		
4	מחוז אשקלון		
4	מחוז דרום		
7	מחוז דרום.		
9	מחוז חיפה		
8	מחוז חיפה והקריות		
8	מחוז ירושלים		
5	מחוז מרכז		
2	מחוז צפון		
3	מחוז תל אביב		
13	מרכז		
7	צפון		
21	תל אביב מחוז		

After translating the original values, I **sort** by the translation and then used the actual district codes from the 2012 Israel districts GIS information.

Freq	OriginalValue	RevisedValue	Comment
6	אשקלון	2	Ashkelon District
4	מחוז אשקלון	2	Ashkelon District
5	מחוז מרכז	4	Central District
13	מרכז	4	Central District
3	חיפה	3	Haifa District
9	מחוז חיפה	3	Haifa District
8	מחוז חיפה והקריות	3	Haifa District
8	מחוז ירושלים	110	Jerusalem District
2	מחוז צפון	5	Northern District
7	צפון	5	Northern District
4	מחוז דרום	2	Southern District
7	מחוז דרום.	2	Southern District
3	מחוז תל אביב	7	Tel Aviv District
21	תל אביב מחוז	7	Tel Aviv District

Because this is a bit of a ‘black box’ when used, it is vital to print out a cross tab of before and after values for quality control purposes. This confirms that changes worked as expected (and that you didn’t forget something). It only makes sense to add this into the function, so I did.

Smart Programming
Mark Friedman 2024-12-10

Here is an example in R of using the functions. The first part assumes you have looked at the data and know what fields you want to handle.

```
# list of fields to correct
variableList <- c("Sex", "מחוז", "Patient.Language")

# create a name for the saved workbook
workbook <- paste0("Corrections-DemogRaw-",
                   format(Sys.time(), '%Y-%m-%d'),
                   "-Draft.xlsx")
# call the function
fnCreateMetaList(DemogRaw, workbook, variableList)
```

After the workbook has been edited and checked, run the next step.

```
# list of fields to correct
variableList <- c("Sex", "מחוז", "Patient.Language")

# Apply the function
Demog <- fnCorrectFields(DemogRaw, workbook, variableList)
```

QC check of מחוז

מחוז_old (original)	מחוז (revised)	Freq
אשקלון	A	6
מחוז אשקלון	A	4
מחוז מרכז	C	5
מרכז	C	13
חיפה	H	3
מחוז חיפה	H	9
מחוז חיפה והקריות	H	8
מחוז ירושלים	J	8
מחוז צפון	N	2
צפון	N	7
מחוז דרום	S	4
מחוז דרום.	S	7
מחוז תל אביב	T	3
תל אביב מחוז	T	21

What started as a basic coding exercise to rename variables and recode values, turned into a reusable, lasting set of functions. The complete code for the functions is available if anyone wants to see it.

Smart Programming
Mark Friedman 2024-12-10

Switching gears to life, the concept is just as valid. We often take shortcuts or do something 'good enough' for the moment.

We had a wonderful property in Silver Spring, Maryland. A micro-farm in the middle of the suburbs with a giant barn. Because my son Lev and I both loved to work with wood, we had a lot of wood that we'd find or purchase and throw in the barn. This also meant that getting to it got increasingly difficult.

We stepped back and built a structure (out of wood) to store and organize the wood. It took time and effort, but eventually saved time when we had to find something, and it kept the wood from getting damaged.

Maybe an even better example is how I handle important information. It used to be, you'd keep a file cabinet with folders. You'd maybe put a (sealed) envelope in it somewhere with critical information such as social security info, passwords, ATM pins, etc. When you get something new to add, you throw it on a desk until it piles up enough to deal with.

But thinking of the lifespan of this information (and its sensitivity), I early on began scanning paper documents, and also using a password management application to store not only passwords, but important information (such as what someone should do if …). I also ensured that the right family members have access to the information. It takes a little longer, but survives moves, fires, floods, forgetfulness and other life events (as long as I can remember that master password 😟).

Thanks for letting me share this with you all.

Sites/People mentioned:

NEHRI https://www.nehri.org/

Soroka CRC https://sorokacrc.org/

Roboform https://www.roboform.com/

Lev Friedman linkedin.com/in/lev-friedman-1079751b5