chunkr

@johnsonlab

chunkr ("chunk-er") is an R package that creates a central repository for R code chunks. chunkr can keep code chunks synced between separate files, or, allows for forking chunks so you can easily import and re-use code.

Copying and pasting code chunks between projects is simple and effective, and version control for .R or .Rmd files keeps your project moving forward. However, some data analyses truly "belong" to more than one project. For example, a single code chunk might produce a dataset analysis that's relevant both to a basic ovary project *and* an ovarian cancer project. In such a case, it is undesirable to repeat the same analysis in two different places (e.g., two separate .Rmd files in two different projects that import and process the same data). Essentially, chunkr is a method for 1) storing useful code chunks in a "central" location, the chunkbase file. 2) symlinking code chunks between files, and/or 3) pulling code into a new file a la "forking" and re-use.

Example workflow

Note: the below workflow does not include an example of data import, including the need for full file paths for raw data, images, etc. It is important to ensure that full paths for imported files are provided within code chunks if their execution is to work correctly in Rmarkdown files that are themselves stored in separate folders.

New code chunk within chunkr workflow

1. Inside file Oldproject.Rmd:

````
```{r, Oldproject.chunk1, echo=FALSE, messages=FALSE, chunkr=TRUE, chunkr.mode=SYNC}

print("hello world")

```
````

Execution of chunk generates and replaces "TRUE" with a **new** hash and the code chunk contents are copied to chunkbase, a database of chunks with unique hashes (below). Header option "chunkr.mode" is automatically populated with option "SYNC".

2. Inside file Oldproject.Rmd:

````
```{r, Oldproject.chunk1, echo=FALSE, messages=FALSE, chunkr=u87th44556t9coml15, chunkr.mode=SYNC}

print("hello world")

```
````

Subsequent editing of this code chunk is SYNC'd with the chunkbase entry corresponding to the original hash upon chunk execution.

Import code chunk to new project, keep SYNC'd with all other file locations that contain unique hash.

3. Inside file Newproject.Rmd, indicate desired chunk to clone by including hash:

````
```{r, Newproject.chunk1, echo=FALSE, messages=FALSE, chunkr=u87th44556t9coml15}
````
````

```

Execution of this new chunk in Newproject.Rmd imports the contents of the hash-identified chunk from chunkbase, autopopulates chunkr.mode to "FORK", reminds user that if they want to keep SYNC'd with original chunk, they have to manually edit chunkr.mode to SYNC:

```{r, Newproject.chunk1, echo=FALSE, messages=FALSE, chunkr=u87th44556t9coml15, chunkr.mode=FORK}

print("hello world")

```

Message in console:

> chunkr imported chunk *u87th44556t9coml15* from chunkbase. To keep SYNC'd with original code chunk, edit chunkr.mode to "SYNC" or your code chunk will be forked again at the next execution.

4. **OPTION 1 for Newproject.Rmd.** FORK original code chunk by keeping chunkr.mode=FORK. At next chunk execution:

```{r, Newproject.chunk1, echo=FALSE, messages=FALSE, chunkr=p8P9572693492slj8, chunkr.mode=FORK}

print("hello world")

```

Message in console:

> chunkr forked chunk *u87th44556t9coml15* from chunkbase and generated new chunk *p8P9572693492slj8* currently saved inside file(s) Newproject.Rmd. Edit chunkr.mode to "SYNC" or your code chunk will be forked again at the next execution.

Here, the user would likely change from FORK to SYNC to prevent generation of a new forked chunk at each execution.

5. **OPTION 2 for Newproject.Rmd.** SYNC original code chunk with the chunk in Oldproject.Rmd by editing chunkr.mode=SYNC. At next chunk execution:

```{r, Newproject.chunk1, echo=FALSE, messages=FALSE, chunkr=u87th44556t9coml15, chunkr.mode=SYNC}

print("hello world")

```

Message in console:

> chunkr chunk *u87th44556t9coml15* is SYNC'd between the following files: Oldproject.Rmd, Newproject.Rmd. This code chunk will be synchronized between these files as long as chunkr is associated with a hash in the chunk header, and chunkr.mode=SYNC.

Moving forward with a code chunk that's SYNC'd between two Rmarkdown files.

5. **Edit Newproject.Rmd.** and changes are SYNC'd with the same chunk in Oldproject.Rmd and stored in chunkbase. At next chunk execution:

```{r, Newproject.chunk1, echo=FALSE, messages=FALSE, chunkr=u87th44556t9coml15, chunkr.mode=SYNC}

print("hello world")
numbers<-c(1:10)
```
```

```
numbers


```

Message in console:

> [1] “hello world”
>
> [1] 1 2 3 4 5 6 7 8 9 10
>
> chunkr chunk *u87th44556t9coml15* is SYNC’d between the following files: Oldproject.Rmd, Newproject.Rmd. This code chunk will be synchronized between these files as long as chunkr is called in the chunk header, and chunkr.mode=SYNC.

chunkbase

chunkbase is the central repository for code chunks. The chunkbase file is a simple text database organized by unique chunk hashes.

Initialize chunkbase

After installation of chunkr (*using devtools*), the user initializes a chunkbase file in the R console as follows:

> > chunkr init chunkbase

This command returns a dialogue

> chunkr chunkbase will be saved to the selected folder. /home/user/chunkbase is a recommended location and .db is a recommended file extension.

and a “save as” dialogue is activated for filename and folder location. For example, a user might initialize under:

/home/chunkr/chunkbase/josh_chunkbase.db

After saving, chunkr will automatically refer to this location and chunkbase filename for handling new R chunk entries.

Report location of chunkbase

The command

> > chunkr location josh_chunkbase

will return the path and filename of the active chunkbase as follows

> The name and location of chunkbase is: **/home/chunkr/chunkbase/josh_chunkbase.db**

Working directly with the chunkbase file

The chunkbase file can be opened in Rstudio or a text editor if desired. While the workflow most often will consist of forking a working code chunk from an existing Rmarkdown file, entries in the chunkbase can be evaluated directly and hashes can be copied here to clone a code chunk into a separate .Rmd file (for example, by pasting into a new R code chunk header). Of course, the chunkbase file will include any comments found in the individual code chunks within. This means the file will be searchable for key words that can help users find relevant code chunks quickly and easily.

Example chunkbase file from above examples:

```
1 chunkr.hash=u87th44556t9coml15

2 print("hello world")
3 numbers<-c(1:10)
4 numbers

---

5 chunkr.hash=p8P9572693492slj8

6 print("hello world")

---
```