


# R パッケージをつくる

@Tokyo.R #119, Sep 20, 2025



The screenshot shows a web browser window with the address bar displaying `ryo-asashi.github.io/somniloquy/`. The page title is "somniloquy 0.1.0 Reference". The main content area features the package name "somniloquy" in a large, bold font. Below it, a paragraph describes the package as a personal collection of miscellaneous R functions. To the right of the text is a blue hexagonal logo with a cartoon squirrel and the text "Somniloquy...". Further right, under the heading "Links", are two blue hyperlinks: "Browse source code" and "Report a bug". Below these links, under the heading "License", is the text "What license is it under?".


▼ A Collection of Miscellaneous R x +

← → ↻ ryo-asashi.github.io/somniloquy/ ⌵ ☆

somniloquy 0.1.0 Reference Search for

## somniloquy

This package is a personal collection of miscellaneous R functions that were developed during several study groups and personal projects. The package serves as a personal toolkit and a record of learning experiences.



**Links**

- [Browse source code](#)
- [Report a bug](#)

**License**

What license is it under?

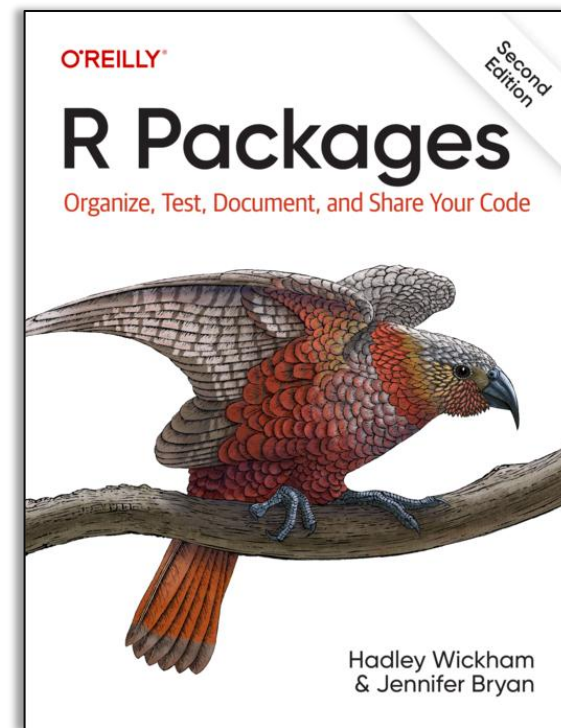
# R Packages (2e) を読めば十分...？

- R パッケージを自作するために必要な情報はすべて [R Packages \(2e\)](#) に書かれています

...ですが、この本の説明はとてもしっかりした作法にのっとっていて、完璧に従うのはけっこう大変です。（第 1 章の最初のセクションだけで 20 項目もあります！）

まずは、もっと雑にやってみましょう！

- この資料では以下の 3 点を前提しています。
  - Git がインストールされていること  
[git: 1.5 使い始める - Gitのインストール](#)
  - GitHub アカウントを持っていること  
[GitHub Docs: GitHub でのアカウントの作成](#)
  - **devtools** がインストールされていること  
[devtools: Tools to Make Developing R Packages Easier](#)  
`install.packages("devtools")`

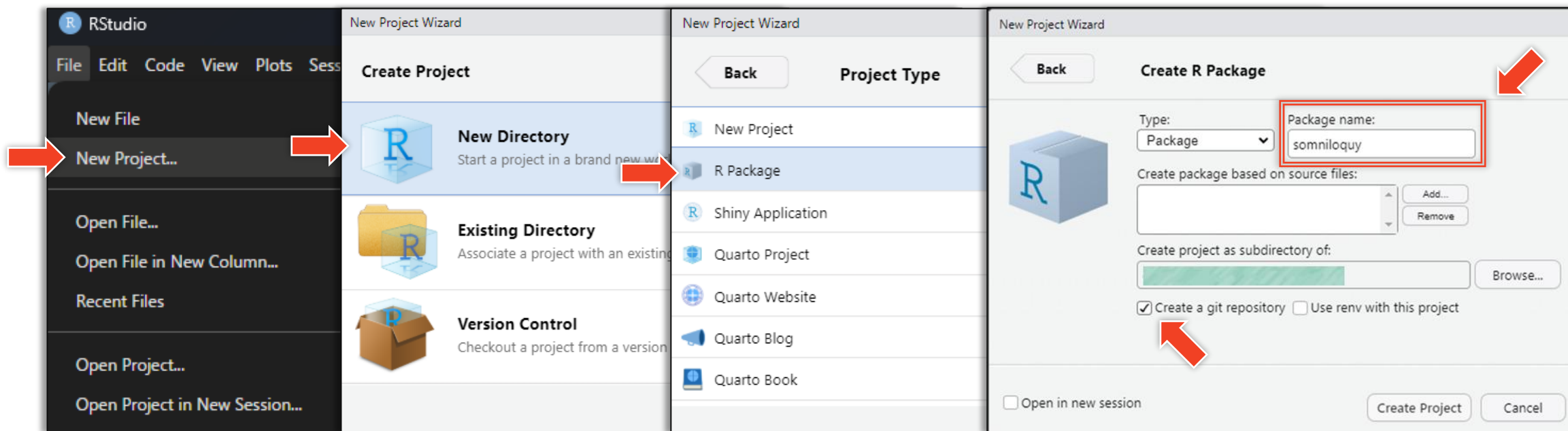


## Getting started > 1. The Whole Game

- 1.1 Load devtools and friends
- 1.2 Toy package: regexcite
- 1.3 Preview the finished product
- 1.4 `create_package()`
- 1.5 `use_git()`
- 1.6 Write the first function
- 1.7 `use_r()`
- 1.8 `load_all()`
- 1.9 `check()`
- 1.10 Edit DESCRIPTION
- 1.11 `use_mit_license()`
- 1.12 `document()`
- 1.13 `check()` again
- 1.14 `install()`
- 1.15 `use_testthat()`
- 1.16 `use_package()`
- 1.17 `use_github()`
- 1.18 `use_readme_rmd()`
- 1.19 The end: `check()` and `install()`
- 1.20 Review

# 1. パッケージのプロジェクトを作成する

- RStudio のメニューから File > New Project... > New Directory > R Package を選択し、パッケージ名とプロジェクトフォルダの置き場所を指定します。
  - ☒ Create a git repository の欄にチェックを入れておきましょう。
- このデモでは **somniloquy** という名前のパッケージを作ります。
  - 勉強会などで自作した雑多な関数を放り込むためのパッケージにすることを想定しています。



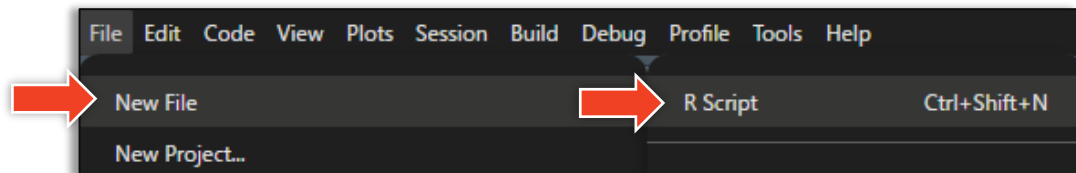
## 2. 作成されたフォルダを整理する

- パッケージと同名のプロジェクトフォルダが作成されたことを確認します。
- 自動的に開かれるスクリプト R/hello.R を閉じて、man/hello.Rd と一緒に削除します。

| パス (プロジェクトフォルダ内)         | 概要                                  |
|--------------------------|-------------------------------------|
| DESCRIPTION              | パッケージのメタデータ (基本情報や依存関係など) を定義するファイル |
| NAMESPACE                | 関数の外部へのエクスポートや外部からのインポートを管理するファイル   |
| R                        | パッケージの関数などを書いた R スクリプトの保存場所         |
| ! R/hello.R              | <b>削除</b> 例示のための R スクリプト            |
| man                      | 関数などのドキュメントに対応する Rd ファイルの保存場所       |
| ! man/hello.Rd           | <b>削除</b> hello.R に対応する Rd ファイル     |
| .Rbuildignore            | パッケージのビルド時に無視するファイルを定めるファイル         |
| .gitignore               | Git のバージョン管理において無視するファイルを定めるファイル    |
| somniloquy(パッケージ名).Rproj | R プロジェクトの設定ファイル                     |

### 3. パッケージの関数を作成する

- File > New File > R Script で新しいファイルを作成し、File > Save as... で R/ に保存します。



- ファイル名は、そのファイルで定義する関数と同名にしておくとうりやすいです。
  - ファイルの作成と保存は、`usethis::use_r("excess_plot")` のようにしても OK。
- 作成されたスクリプトの中で関数を定義し、ファイルを上書き保存します。
    - ここでは平均／メディアン超過プロットを描くための関数 `excess_plot()` を作成します。

excess\_plot.R

```
1 excess_plot <- function(  
2   x, method = c("mean", "median"), mode = c("exact", "interpolate"),  
3   show_plot = TRUE, ...  
4 ) {  
5   method <- match.arg(method)  
6   mode <- match.arg(mode)  
  
32  if (is.null(args$type)) args$type <- "l"  
33  do.call(base::plot, c(list(y ~ u, data = res), args))  
34  invisible(res)  
35 } else {  
36   res  
37 }  
38 }
```

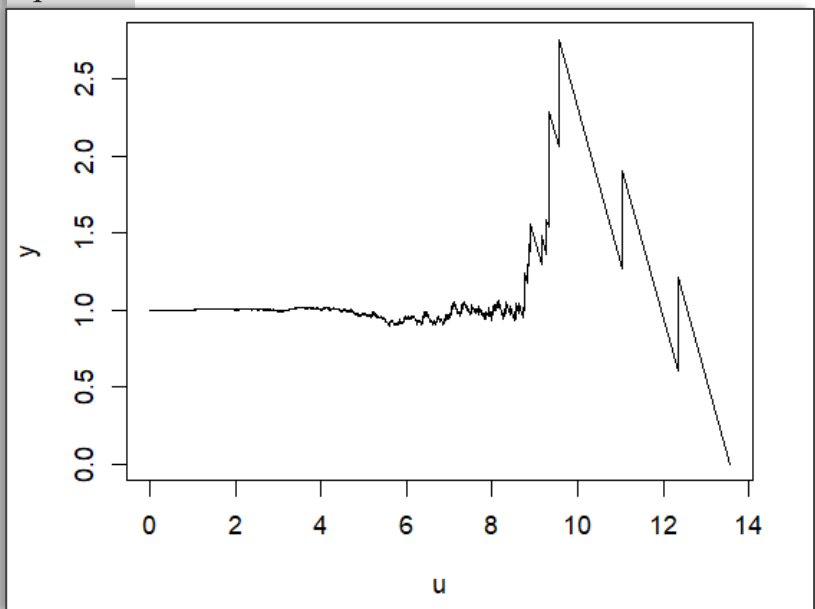
## 4. 関数が正しく機能することを確認する

- `devtools::load_all()` を使って作成した関数が正しく機能することを確認めます。

console

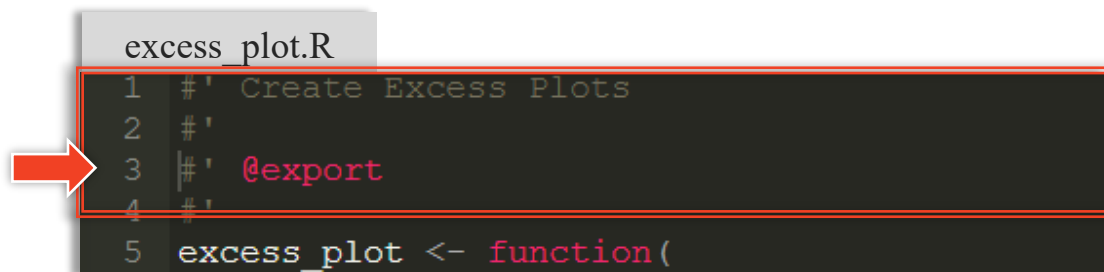
```
> devtools::load_all()
i Loading somniloquy
> search()
[1] ".GlobalEnv"      "package:somniloquy" "devtools_shims"    "tools:rstudio"
[5] "package:stats"   "package:graphics"  "package:grDevices" "package:utils"
[9] "package:datasets" "package:methods"   "Autoloads"         "package:base"
> excess_plot(rexp(1e5))
```

plots



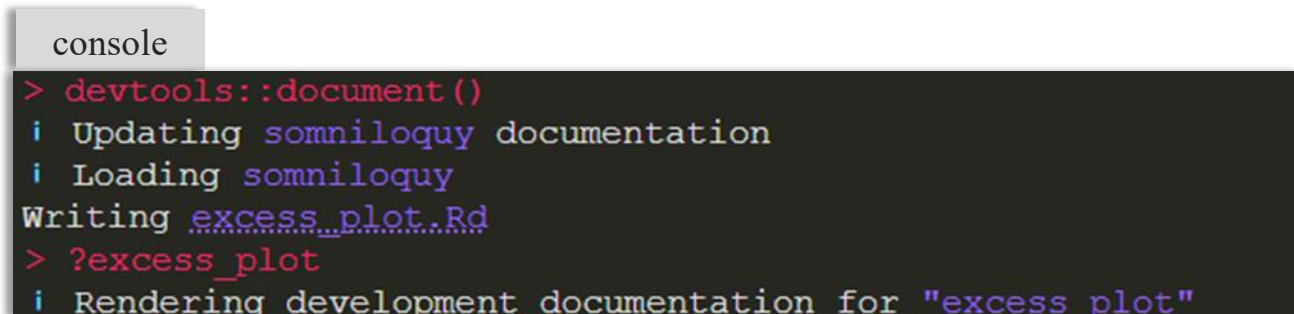
## 5. 関数がエクスポートされるように設定する

- 関数を作成した R スクリプトの冒頭に、その関数を端的に説明する「見出し」と `@export` タグを追加します。



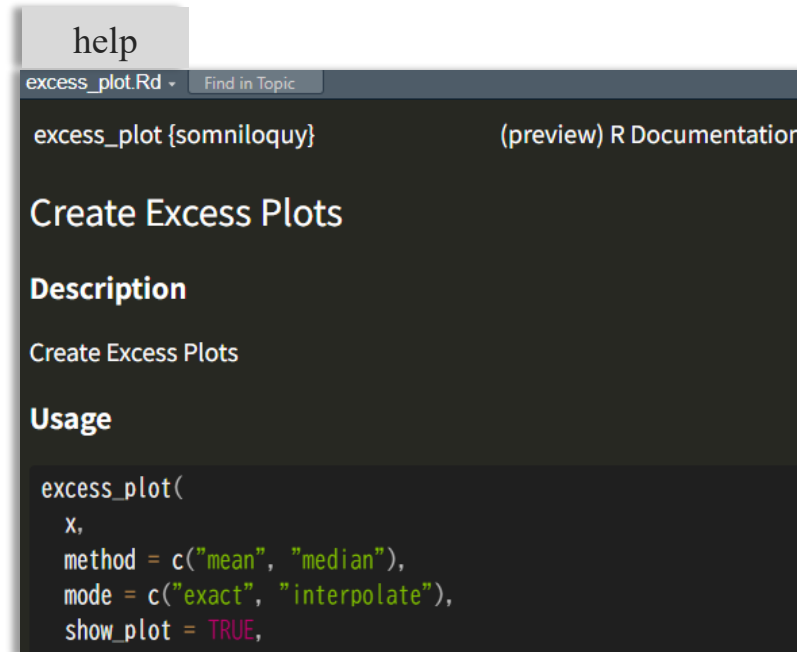
```
excess_plot.R
1 #' Create Excess Plots
2 #'
3 #' @export
4 #'
5 excess_plot <- function(
```

- `devtools::document()` を実行して、`man/excess_plot.Rd` を作成します。



```
console
> devtools::document()
i Updating somniloquy documentation
i Loading somniloquy
Writing excess_plot.Rd
> ?excess_plot
i Rendering development documentation for "excess_plot"
```

[excess\\_plot.Rd](#) / ?excess\_plot



```
help
excess_plot.Rd - Find in Topic
excess_plot {somniloquy} (preview) R Documentation

Create Excess Plots

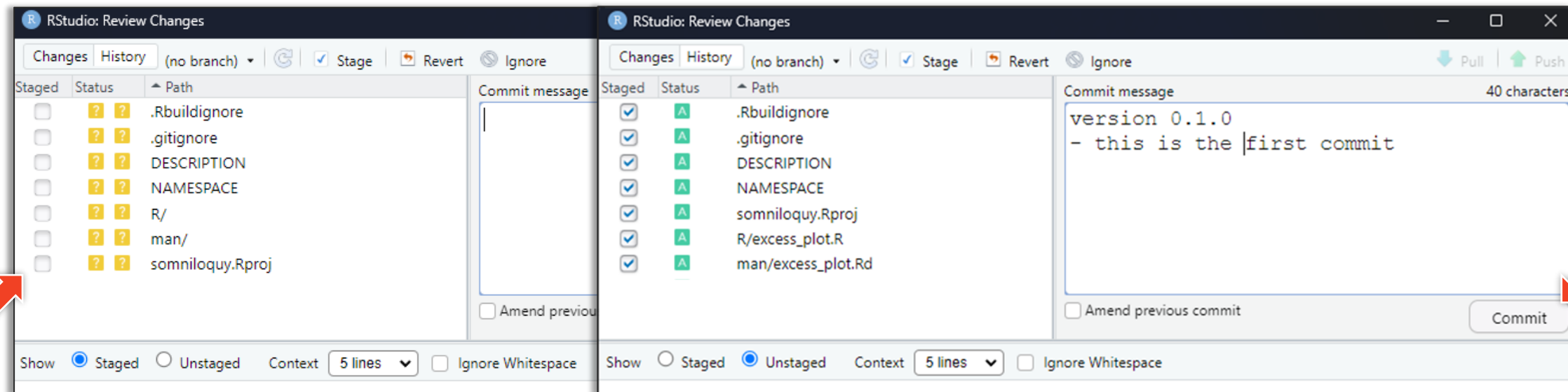
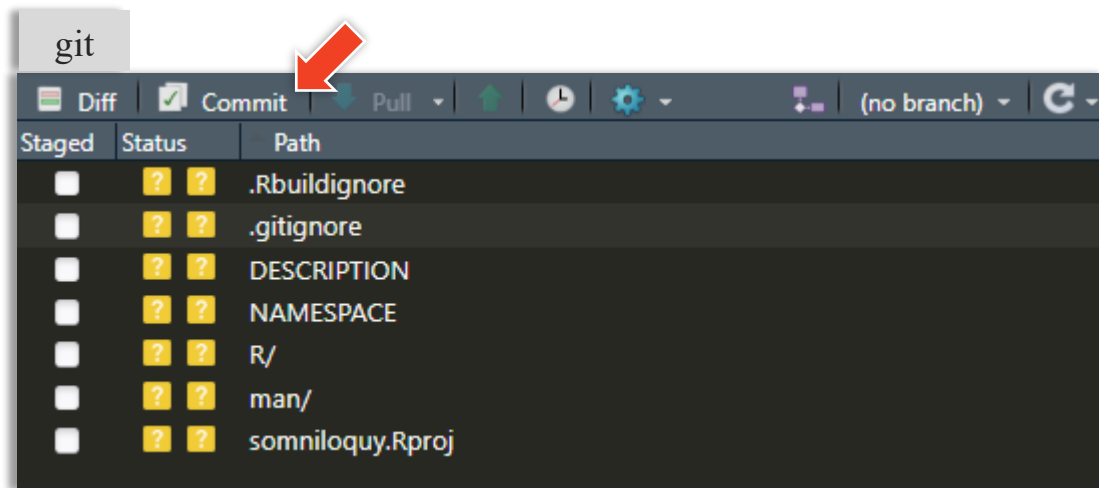
Description
Create Excess Plots

Usage
excess_plot(
  x,
  method = c("mean", "median"),
  mode = c("exact", "interpolate"),
  show_plot = TRUE,
```



## 6. すべての変更をコミットする

- R Studio の git タブにある ☒ Commit から進み、すべての変更をコミットします。





# 7. GitHub にアップロードする

- `usethis::use_github()` でプロジェクトフォルダを GitHub にアップロードします。
  - 初めて `use_github()` を使うときは、ブラウザ経由で GitHub アカウントの認証を要求されます。

```
console
> usethis::use_github()

✓ Creating GitHub repository "ryo-asashi/somniloquy".
✓ Setting remote "origin" to
  "https://github.com/ryo-asashi/somniloquy.git".
✓ Adding "https://github.com/ryo-asashi/somniloquy" to URL.
✓ Adding "https://github.com/ryo-asashi/somniloquy/issues" to BugReports.
i There is 1 uncommitted file:
• DESCRIPTION

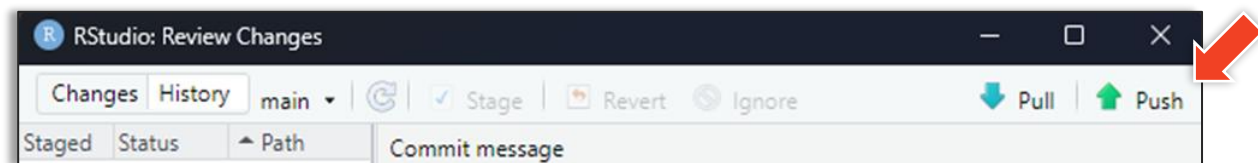
! Is it ok to commit it?

1: Nope
2: Negative
3: For sure

Selection: 3

✓ Adding files.
✓ Making a commit with message "Add GitHub links to DESCRIPTION".
✓ Pushing "main" branch to GitHub and setting "origin/main" as upstream
  branch.
✓ Opening URL <https://github.com/ryo-asashi/somniloquy>.
```

- 今後、ファイルの変更をコミットしたら、GitHub リポジトリにも変更をプッシュしましょう。



## 8. 完成！ 🎉 GitHub からインストールする

- これで完成！

`pak::pak()` や `devtools::install_github()` でのインストールが可能です。

console

```
> pak::pak("ryo-asashi/somniloquy")
```

```
→ Will install 1 package.
→ Will download 1 package with unknown size.
+ somniloquy 0.1.0 [bld][cmp][dl] (GitHub: 123ef63)
i Getting 1 pkg with unknown size
✓ Got somniloquy 0.1.0 (source) (1.56 MB)
✓ Downloaded 1 package (1.56 MB) in 941ms
i Packaging somniloquy 0.1.0
✓ Packaged somniloquy 0.1.0 (4.3s)
i Building somniloquy 0.1.0
✓ Built somniloquy 0.1.0 (1.4s)
✓ Installed somniloquy 0.1.0 (github::ryo-asashi/somniloquy@123ef63) (119ms)
✓ 1 pkg: added 1, dld 1 (NA B) [13.6s]
```

console

```
> library(somniloquy)
> excess_plot(rnorm(1e5))
```

- なお、開発上は GitHub を経由せず `devtools::install()` でインストールしても問題ありません。

# Optional. DESCRIPTION を更新する

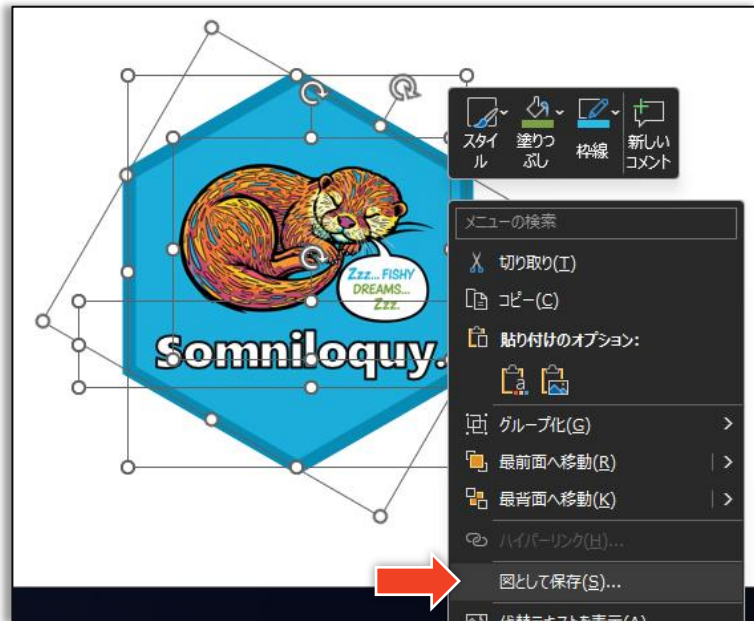
- DESCRIPTION は、パッケージのメタデータ（基本情報や依存先パッケージ）を記載するファイルです。ここでは、Title / Authors@R / Description などを書き換えておきます。
  - このうち、Title は GitHub リポジトリの About 欄にも反映させておくによさそうです。

## DESCRIPTION

```
1 Package: somniloquy
2 Type: Package
3 Title: A Collection of Miscellaneous R Functions
4 Version: 0.1.0
5 Authors@R: c(
6   person(
7     "Ryoichi", "Asashiba",
8     email = "ryoichi.asashiba@gmail.com",
9     role = c("aut", "cre")
10  )
11 )
12 Description: This package is a personal collection of miscellaneous R functions
13 License: What license is it under?
14 Encoding: UTF-8
15 LazyData: true
16 RoxygenNote: 7.3.2
17 URL: https://github.com/ryo-asashi/somniloquy
18 BugReports: https://github.com/ryo-asashi/somniloquy/issues
19
```

# Optional. パッケージロゴを作成する

- PowerPoint で外枠・背景・タイトルを用意します。
  - 外枠と背景はそれぞれ「挿入>図形>六角形」で作成  
「図形の書式」でサイズを 10.22 cm×11.82 cm に変更して 90 度回転
  - 外枠のみ、「図形の枠線>太さ」で枠の幅を 12pt に設定
- ロゴ用のイラストは生成AIに描いてもらいました。
- 作成したロゴは man/figures/logo.png に保存します。
  - パーツをまとめて選択して「右クリック>図として保存」



# Optional. README を追加する

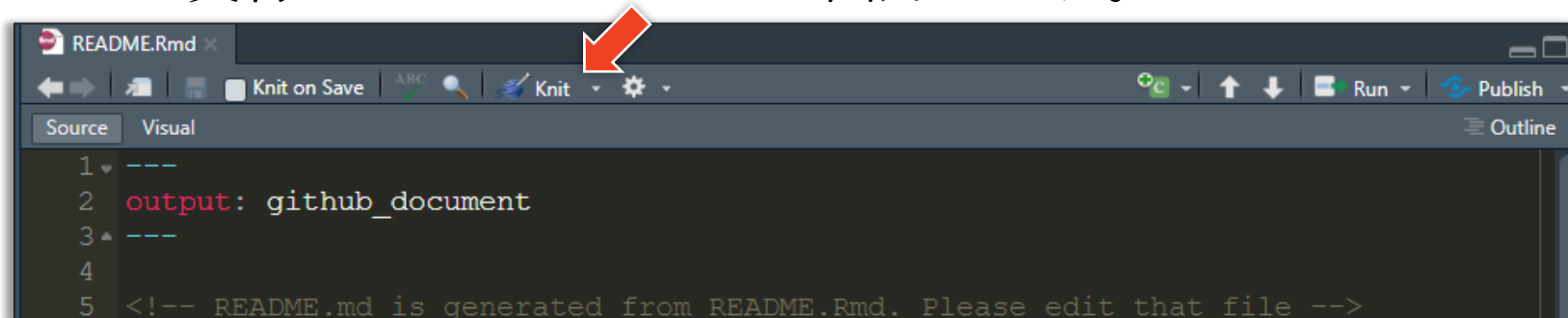
- `usethis::use_readme_rmd()` を実行して README.Rmd を作成します。

console

```
> usethis::use_readme_rmd()
✓ Setting active project to "C:/Users/daysb/Documents/RFiles/somniloquy".
✓ Writing README.Rmd.
✓ Adding "^README\\.Rmd$" to ..Rbuildignore.
❑ Modify README.Rmd.
✓ Writing ..git/hooks/pre-commit.
```

- README.Rmd の16行目を以下のように書き換えるとロゴを入れられます。  
# somniloquy 
- 21行目は DESCRIPTION の Description をコピーしておけば OK です！
- 32行目以下は「パッケージの使用例」のサンプルです。とりあえず消しておきましょう。

- Knit を実行して README.md を出力します。



# Optional. GitHub Pages で Web サイトを作る

- `usethis::use_pkgdown()` と `usethis::use_pkgdown_github_pages()` でパッケージを紹介するための [Web サイト](#) (GitHub Pages) を作れます。
  - ロゴを作っているなら、`pkgdown::build_favicons()` でファビコンも作成できます。

console

```
> usethis::use_pkgdown()

> usethis::use_pkgdown_github_pages()
! Overwrite pre-existing file _pkgdown.yml?

1: Definitely
2: Nope
3: No

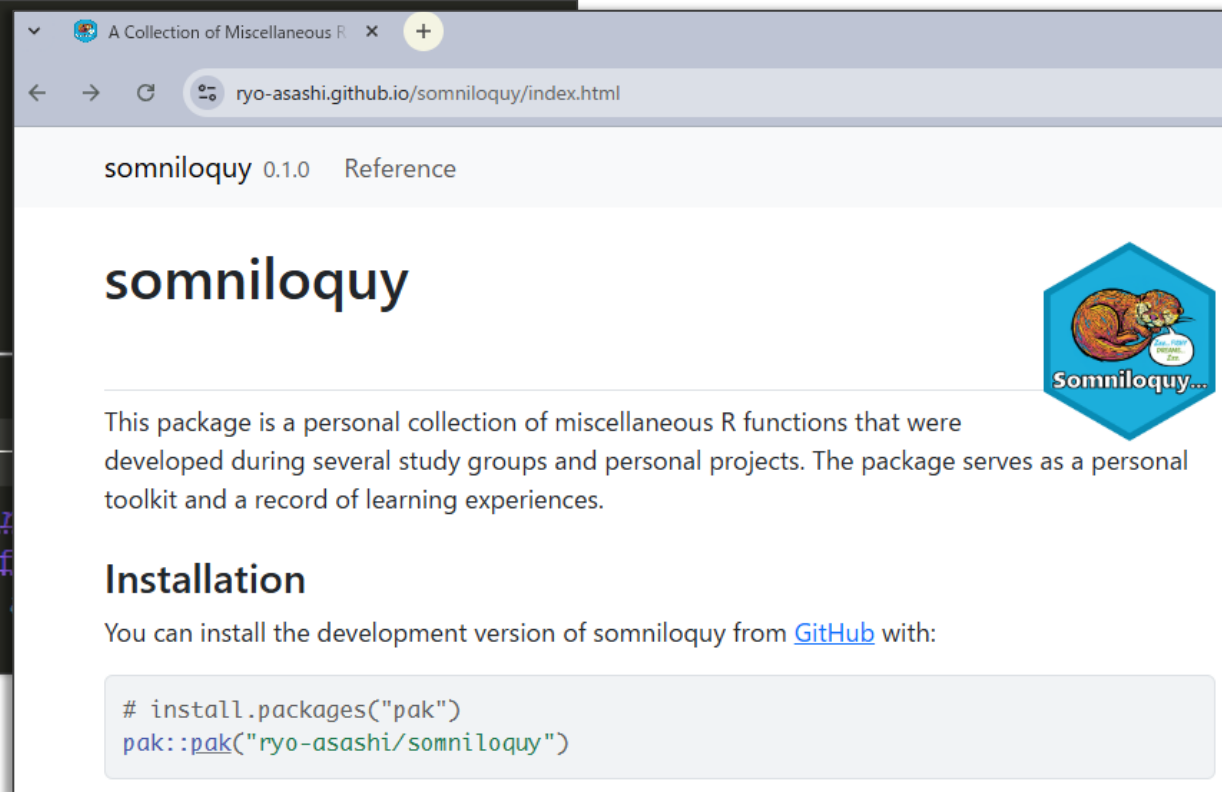
Selection: 1

> pkgdown::build_favicons()

— Building favicons —

i Building favicons with <https://realfavicongenerator.com/>
✓ Added apple-touch-icon.png, favicon-96x96.png, favicon-16x16.png,
  site.webmanifest, web-app-manifest-192x192.png,
  web-app-manifest-512x512.png.
```

☒ Commit &  Push





# この資料で説明しなかったこと（の一部）

- 他のパッケージの関数を利用する（**base**を除く）
  - DESCRIPTION の Imports: か Suggests: にパッケージ名を記載し、スクリプト内では `stats::median()` のようにパッケージ環境を指定して参照します。 `usethis::use_package("stats")` でも OK。
  - 特に、**ggplot2** を利用する場合は [vignette\("ggplot2-in-packages"\)](#) が参考になります。
- パッケージのライセンスを設定する
  - ライセンスは、パッケージのコードを著作権者でない人がどのように利用できるかを規定します。
  - `usethis::use_mit_license()` を実行すると MIT ライセンスを設定したものとして体裁が整います。
- パッケージの関数に対するコードテストを実行する
  - `usethis::use_testthat()` によって **testthat** パッケージのコードテスト機能を利用できます。
  - その後は `usethis::use_test()` 関数などを用いてテスト用スクリプトを追加していきます。
- パッケージの問題点をチェックする
  - `devtools::check()` を用いてパッケージが問題を抱えていないかを包括的にチェックできます。
  - 潜在的な問題があれば、`errors`, `warning`, `notes` という 3 種類の注意が表示されます。（CRAN への投稿を考えていないなら、`errors` 以外はそれほど深刻にとらえなくてもよいと思っています。）