



Créer ses packages avec Rstudio

Travail collaboratif avec Rstudio

Rstudio

The screenshot displays the RStudio environment with the following components:

- Script Editor:** Contains R code for a function named `calendrier`. The code includes comments for title, usage, description, parameters, and examples, followed by the function definition.
- Console:** Shows the R version (3.0.2), copyright information, and introductory text about R.
- Environment Pane:** Displays "Global Environment" and states "Environment is empty".
- Files Pane:** Lists files in the current directory, including `.Rbuildignore`, `DESCRIPTION`, `man`, `NAMESPACE`, `R`, `Read-and-delete-me`, and `Utils.Rproj`.

```
1 #! @title Donne la date en format "aaaa-mm-jj" a partir de l'annee et du numero du jour.
2 #!
3 #! @usage calendrier(annee,jour,separateur="-")
4 #! @description Prend une date definit par une annee et un numero de jour et determine l'ecritur
5 #! de cette date dans le format "aaaa-mm-jj".
6 #! @param annee Un vecteur numerique contenant les annees des jours dont on souhaite avoir la
7 #! date.
8 #! @param jour Un vecteur numerique contenant le numero des jours.
9 #! @param separateur Une chaine de caractere specifiant opar quoi seront separes l'annee, le mo
10 #! le jour dans le format final. Par default : \code{separateur} = "-".
11 #! @return Un vecteur de chaine de caractere contenant les dates au format "aaaa-mm-jj"
12 #! pour chacun des jours decrits par les valeurs d'entrees \code{annee} et \code{jour}.
13 #! @export
14 #! @examples
15 #! calendrier(1989,73)
16 #! calendrier(2000:2005,c(30,31,32,91,91,90))
17 calendrier <- function(annee,jour,separateur="-"){
18   annee <- as.numeric(annee)
19   jour <- as.numeric(jour)
20
21   njour.mois <- c(0,31,28,31,30,31,30,31,31,30,31,30,31)
22   njour.mois <- sapply(annee,function(valeur){
23     res <- njour.mois
24     res[3] <- res[3] + as.numeric(is.bissextile(valeur))
25   })
26 }
```

R version 3.0.2 (2013-09-25) -- "Frisbee Sailing"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

01

Créer ses packages avec Rstudio

Pourquoi faire un package R ?

❖ POUR :

- altruisme : moyen de faire un retour à la communauté R
- pratique : moyen de diffuser son travail
- extra testing : d'autres peuvent trouver des bugs, améliorer le package

❖ CONTRE :


- fastidieux : nécessité de se conformer aux standards
- commodité : il peut être plus difficile d'adapter un package plutôt qu'un bout de code

Créer un nouveau package

Create project - New directory – R package

New Project

Back **Create R Package**



Type: Package Package name: Utils

Create package based on source files:

~/Bureau/RSTUDIO/calendrier.R
~/Bureau/RSTUDIO/is_bissextile.R

Add...
Remove

Create project as subdirectory of: ~ Browse...

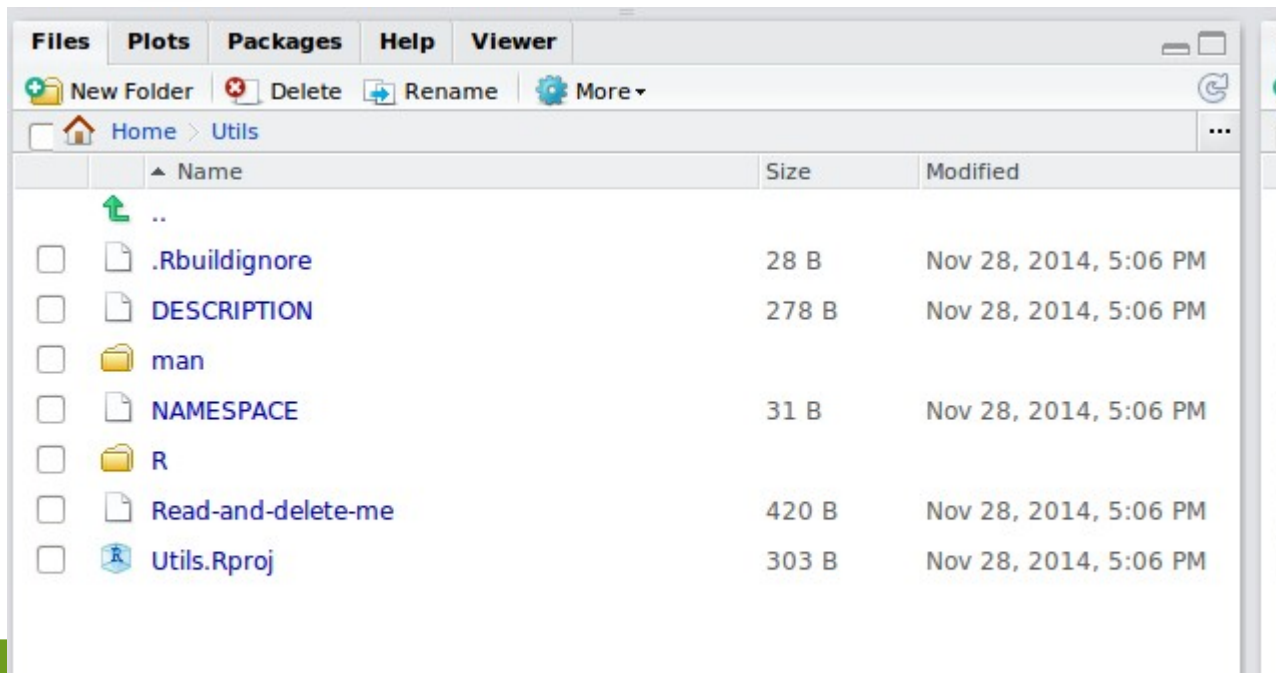
Use packrat with this project

Open in new window

Create Project Cancel

Créer un nouveau package

L'arborescence du package est créée



Construire un package

- Renseigner les titres dans les Rd files (R documentation)
- Onglet Build – **build&reload** (Ctrl+Build+B)

Vérifier un package

- Onglet **Build – check**

- Options (**More - configure build tools**) :

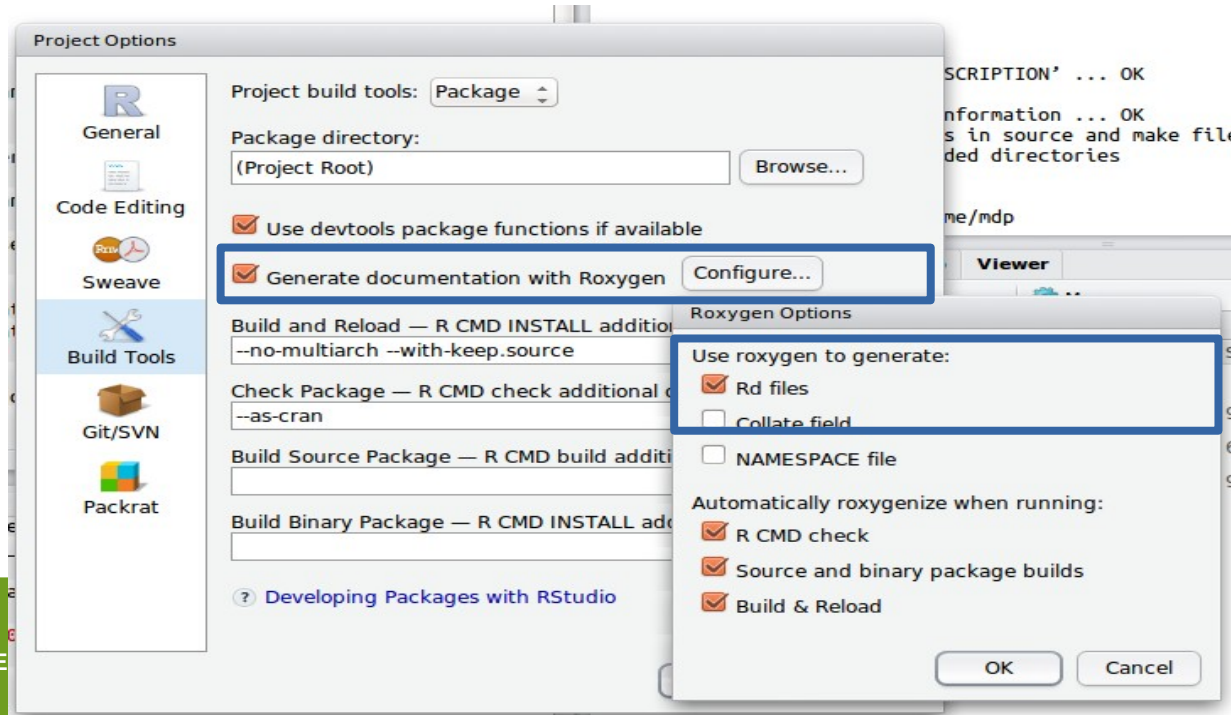
--as-cran — Select customizations similar to those used for CRAN incoming checking

--no-manual — Do not produce the PDF manual

--no-vignettes — Do not check Sweave vignettes

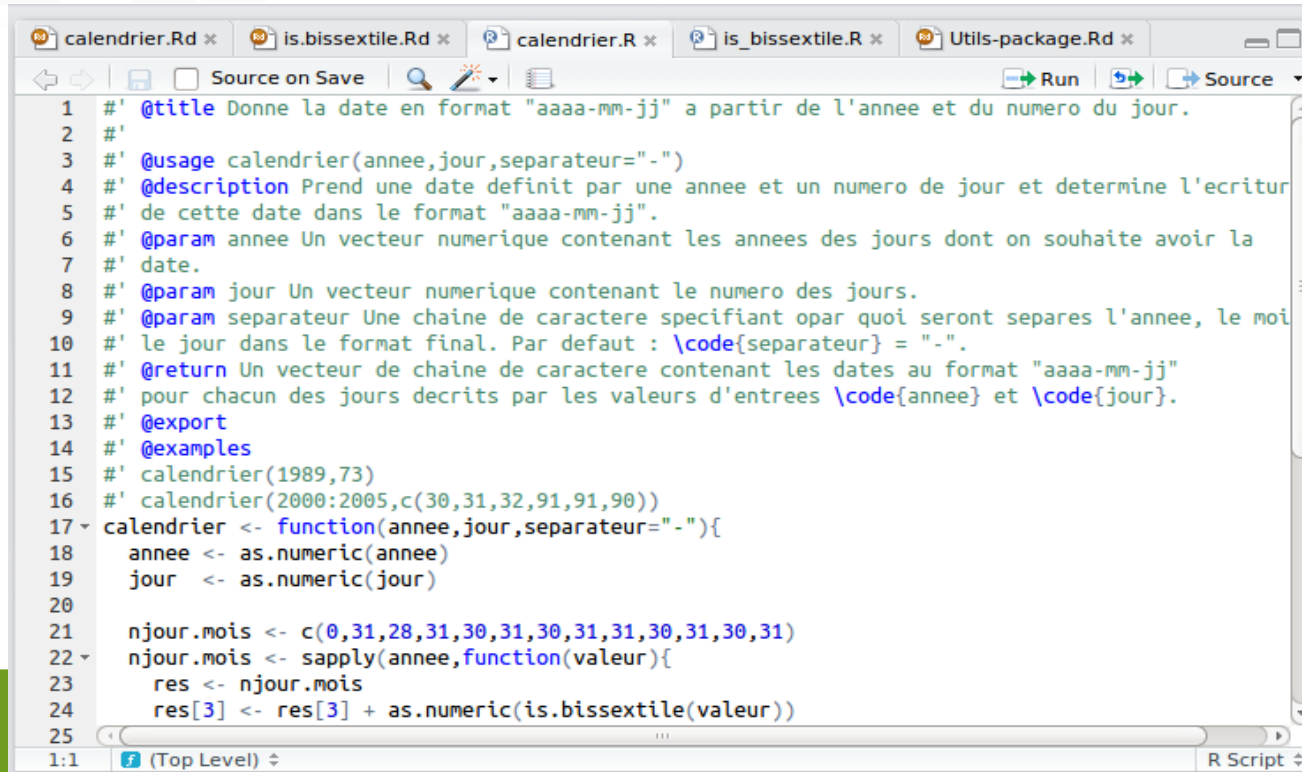
Documenter son package

- installer le package **Roxygen2**
- configurer Rstudio (**More-configure build tools**) :



Documenter son package

- documenter ses scripts R et reconstruire le package (**Build&Reload**)



```
1 #' @title Donne la date en format "aaaa-mm-jj" a partir de l'annee et du numero du jour.
2 #'
3 #' @usage calendrier(annee,jour,separateur="-")
4 #' @description Prend une date definit par une annee et un numero de jour et determine l'ecriture
5 #' de cette date dans le format "aaaa-mm-jj".
6 #' @param annee Un vecteur numerique contenant les annees des jours dont on souhaite avoir la
7 #' date.
8 #' @param jour Un vecteur numerique contenant le numero des jours.
9 #' @param separateur Une chaine de caractere specifiant opar quoi seront separees l'annee, le moi
10 #' le jour dans le format final. Par default : {separateur} = "-".
11 #' @return Un vecteur de chaine de caractere contenant les dates au format "aaaa-mm-jj"
12 #' pour chacun des jours decrits par les valeurs d'entrees {annee} et {jour}.
13 #' @export
14 #' @examples
15 #' calendrier(1989,73)
16 #' calendrier(2000:2005,c(30,31,32,91,91,90))
17 calendrier <- function(annee,jour,separateur="-"){
18   annee <- as.numeric(annee)
19   jour <- as.numeric(jour)
20
21   njour.mois <- c(0,31,28,31,30,31,30,31,31,30,31,30,31)
22   njour.mois <- sapply(annee,function(valeur){
23     res <- njour.mois
24     res[3] <- res[3] + as.numeric(is.bissextile(valeur))
25
```

Distribuer un package

- Un package peut être distribué de différentes façons :
 - source package
 - binary package
- Plusieurs dépôts possibles :
 - CRAN (Comprehensive R Archive Network)
 - R-forge, GitHub

DEMO

- créer un nouveau projet de type package, y inclure 2 scripts R
- vérifier le package
- construire le package (avec roxygen)
- faire une modification dans la documentation et reconstruire le package
- générer un binaire/un package source pour diffusion

Distribuer un package

- Publier sur **R-forge** :

```
install.packages("MyPackage", repos="http://R-Forge.R-project.org")
```

- Publier sur **GitHub** :

```
library(devtools)  
install_github("MyUsername", "MyPackage")
```

- Publier sur **CRAN** :

cf <http://cran.r-project.org/web/packages/policies.html>

02

Travail collaboratif avec Rstudio

Travail collaboratif

Un logiciel de **gestion de versions** facilite le travail collaboratif.

Il permet de stocker un ensemble de fichiers en conservant la chronologie de toutes les modifications qui ont été effectuées dessus.

Travail collaboratif

Pourquoi utiliser un gestionnaire de version :

- gérer l'historique des modifications (faites par vous ou un tiers),

comparer différentes versions

- sauvegarder son travail
- collaborer plus facilement

Gestionnaire de version

Rstudio supporte 2 gestionnaires de version :

- **git**

- **svn**

```
sudo apt-get install git-core  
sudo apt-get install subversion
```

Gestionnaire de version

Création d'un nouveau projet basé sur git ou subversion :

Create New project – Version control – subversion ou git

New Project

Back Checkout Subversion Repository

Repository URL:
svn+ssh://vitsec@scm.mulcyber.toulouse.inra.fr/

Username (if required for this repository URL):
vitsec

Project directory name:
vitsec

Create project as subdirectory of:
~ Browse...

Open in new window Create Project Cancel

Création d'un projet déposé
sur la forge MULCYBER (INRA, MIA)

Gestionnaire de version

The screenshot displays the RStudio environment with the following components:

- Script Editor:** Shows the source code for the `calendrier` function in `calendrier.R`. The code includes comments in French, parameter definitions, and the function implementation.
- Environment/History/SVN Panel:** A blue-bordered panel on the right side of the script editor, currently empty.
- Files Panel:** Shows the file explorer for the `~/vit-sec` directory. It lists several folders and files:

Name	Size	Modified
..		
branches		
tags		
trunk		
vit-sec.Rproj	205 B	Nov 30, 2014, 5:39 PM
calendrier.R	1.6 KB	Dec 1, 2014, 11:25 AM
- Console:** Shows the R startup output, including the version (3.0.2), copyright information, and instructions for using R.

DEMO

1. ajout de fichiers/comparaison des versions

- Ajouter un fichier dans le projet => il est marqué **unversionned** dans Rstudio
Cliquer sur Add => il fait parti des fichiers suivis (**Added**)
- Faire un commit
- Faire une modif sur le fichier et enregistrer => son status passe à Modified
- Faire un diff pour voir les différences
- Si les modifs sont OK on commite sinon on fait un revert

2. suppression d'un fichier :

- Supprimer le fichier en local => le statut passe à missing
- Revert permet de récupérer le fichier
- Commit permet de valider la suppression



Merci pour votre attention

Des questions ?