R-Forge User's Manual

SVN Revision: 227, November 3, 2011

R-Forge Administration and Development Team

The current members of the R-Forge Administration and Development Team are Kurt Hornik, David Meyer, Stefan Theußl and Achim Zeileis. Former members include: Martin Kober. To contact the authors please write an email to `R-Forge@R-project.org`.

Contents

1 Introduction

Software developers typically use collaborative development tools like Subversion (SVN, see Pilato et al., 2004) or Concurrent Versions System (CVS, see Cederqvist et al., 2006) in order to manage their code base efficiently. E.g., the R Development Core Team is using a central SVN repository provided by ETH Zürich (`http://svn.R-project.org`) for the base R distribution. What is more, many R package developers around the world use similar infrastructure built upon their own solutions since collaborative development is a key factor for open source projects being successful. Thus, the **R-project** aims to provide infrastructure in order to facilitate such an open decentralized development process for the entire R community.

R-Forge (Theußl and Zeileis, 2009) provides a set of tools for source code management and various web-based features. It aims to provide a platform for collaborative development of R packages, R-related software or further projects. It is closely related to the most famous of such platforms—the world's largest open source software (OSS) development website—namely `http://SourceForge.net`. What is different is the provided set of tools with a strong focus on the need of the R community. All in all, R-Forge is a place where all R developers and users can come together and exchange their knowledge. Joining this community service gives you access to all we have to offer.

Basically, R-Forge is based on FusionForge (`http://fusionforge.org/`)—the open source successor of GForge (Copeland et al., 2006)—which is a framework integrating various tools like SVN for collaborative work on source code, mailing lists, bug tracking etc. into one platform. On R-Forge all work is organized in the same entity, namely a "Project". In these projects it is possible to host one or more packages and other R-related material. The main advantage is, that uploaded code is going to be packaged (i.e., built) and checked every day for various platforms not only with the latest patched version of R (see *What is the current version of R?* in Hornik, 2010) but also with the release candidate just before a new version of R is to be released. Furthermore, packages passing the quality management system can be released from R-Forge to CRAN in a standardized way.

Additionally, R-Forge provides other tools to coordinate the work between project members and to communicate with their user base.

- Project websites are a way for developers to present their work on a subdomain of R-Forge (E.g., `http://foo.R-forge.R-project.org`). It is also possible to offer a link to another website on the project summary page instead.
- Mailing lists: By default one is automatically created when setting up a project. Additional mailing lists can be set up easily as well.
- Projects can be categorized into different topics (e.g., biostatistics, finance, regression analysis, ...). This enables other people to quickly find what they are looking for. People can browse the categories in the so-called "Project Tree" tab. In the default setting the project tree lists alphabetically all projects including a short description similar to CRAN. It is clear that this tree cannot be complete, so people are welcome to make suggestions on how we could improve it.
- Forums can be set up separately by the project administrators.
- News can be submitted to the project summary page as well as to the front page. The latter needs approval by one of the R-Forge Administration and Development Team members. It is also possible to download them as RSS feeds.

In this manual we present all relevant steps to get started with R-Forge. If you need further help or like to submit comments regarding R-Forge please send an email to `R-Forge@R-project.org`. For a more detailed documentation regarding the underlying FusionForge system we refer to `https://fusionforge.org/docman/view.php/6/1/gforge_manual.plain.html`.

2 Registration

2.1 Registering a New User

To register a new user, click on the "New Account" link on the top right side of the browser window at `http://R-Forge.R-project.org`. Fill out the form (there are descriptions and hints for each field on this site) and click on "Submit" afterwards. You will receive an email with a URL to your specified email address. After clicking on this link your account has been verified. Now you're able to login to the website.

Important note: The tab "My Page" is the most important page on R-Forge. Here you configure your account, you see your project memberships and see the items, which have been assigned to you (i.e. bugs, feature requests, etc.).

Now you can start your own project (see Section 2.3 for details) or become a member of an existing project (Section 2.2).

2.2 Joining a Project

If you like to join an existing project you achieve this by doing the following steps:

1. First you need the name of the project you want to become a member of. You can "search" for the project (top middle side of the browser window) or you click on one of the projects showing in the "Project Tree". The latter is an alphabetically sorted list of all R-Forge projects.

2. Then you go to the project summary page (should be the default entry point). There is a window on the right side called "Developer Info". To join this project you need the permission of the project admin. So click on "Request to join" to send the project admin an email.

3. If the project admin decides to add you as developer you will receive an email confirming your developer account. Now you have full SVN access (see section 3 for details).

2.3 Registering a New Project

Registering a new project is easy: Go to the R-Forge website, login and go to "My Page" section. You have a "Register Project" link in the menu at the top of your page. Now fill in the form and submit your project. After approval of the project by the R-Forge admins you will be notified via email and you will be able to start with your project on R-Forge.

3 Source Code Management

3.1 Introduction

When carrying out software projects, source files change over time, new files get created and old files deleted. Typically, several authors work on several computers on the same and/or different files and keeping track of every change can become a tedious task. In the open source community, the general solution to this problem is to use version control, typically provided by the majority of SCM tools. For this reason R-Forge utilizes SVN to facilitate the developer's work when creating software.

A central repository ensures that the developer has always access to the current version of the project's source code. Any of the authorized collaborators can "checkout" (i.e. download) or "update" the project file structure, make the necessary changes or additions, delete files from the current revision and finally "commit" changes or additions to the repository. More than that, SVN

keeps track of the complete history of the project file structure. At any point in the development stage it is possible to go back to any stage in the history as well as to inspect and restore old files. This is called version control as every stage automatically is assigned a unique version number which increases over time.

On R-Forge such a version-controlled repository is automatically created for each project. To get started, the project members just have to install the client of their choice (e.g., Tortoise SVN on Windows or svnX on Mac OSX) and check out the repository. In addition to the inherent backup of every version within the repository a backup of the whole repository is generated daily.

A rights management system assures that, by default, anonymous users have read access and developers have write access to the data associated with a certain project on R-Forge. More precisely, registered users can be granted one of several roles: e.g., the "Administrator" has all rights including the right to add new users to the project or release packages directly to CRAN. He/she is usually the package maintainer, the project leader or has registered the project originally. Other members of a project typically have either the role "Senior Developer" or "Junior Developer" which both have permission to commit to the project SVN repository and examine the log files in the *R Packages* tab. The differences between those roles are subtle, e.g., senior developers additionally have administrative privileges in several places in the project. When we speak of developers in subsequent sections we refer to project members having the rights at least of a junior developer.

3.2 How to Get SVN to Work

R-Forge uses **Subversion** (SVN, `http://subversion.tigris.org`) for source code management. You need an SVN client (e.g. "Tortoise SVN" on Windows machines, `http://tortoisesvn.tigris.org`) to take full advantage of your SVN repository. For security reasons we use secure shell (SSH) tunneling for developer accounts which means that all network traffic is encrypted. This means that you have to setup your machine accordingly.

3.2.1 Windows

The software mentioned in this section can be found on
`http://www.chiark.greenend.org.uk/~sgtatham/putty/`.

1. get and install the latest TortoiseSVN client from `http://tortoisesvn.net/downloads`. Note: Windows Vista users have to take care that they choose the right installation package for their architecture (i.e., Windows Vista 64-bit users have to download the 64-bit installer).

2. it is sufficient to use password authentication to write to the project repository (if you decided to do so please go to step 5). But for convenience purposes (avoiding password queries) it is probably better to generate an SSH keypair and upload the **public** key to R-Forge:

   - Download and execute `puttygen.exe`.
   - leave ssh-2 rsa marked and click on "generate" (make some random movements).
   - save the private key using the corresponding button (you don't need to set a password).
   - Mark the text in the text field "Public key for pasting into OpenSSH authorized_keys file".
   - copy and paste it into your shell account information configuration: Go to `http://R-forge.R-project.org` and login. Go to "My Page" and then click on "Account maintenance". At the bottom of this page you click on "edit keys" in the "Shell Account Information" window (**Important note:** you must be a project admin or member of a project to do this, otherwise there won't be an option "Shell Account Information", see section 2.2 for joining a project or section 2.3 for registering a project). The key you enter here is typically of the form:

```
ssh-rsa AAAA... foo@bar
```

The first field describes the type of key, the second field is the key itself, and the third field is a comment. **It is important that there are no newlines within a key**.

3. Now you have to wait until the next full hour. The keys are activated once an hour only.

4. Next you need an authentication agent like `pageant.exe`. Load your **private** key with `pageant.exe` (right click on the pageant tray icon and then "add key".

5. Finally check out the repository using the URL given on the project website (tab "SCM") under "developer account" with "Tortoise SVN" or the SVN client of your choice. (Note: if this last step fails for some reason, wait an hour an try again. The logins are created only once an hour only)

3.2.2 Linux/Unix

1. It is sufficient to use password authentication to write to the project repository (if you decided to do so please go to step 4). But for convenience purposes (avoiding password queries) it is probably better to upload an SSH keypair: Generate and save the keys using `ssh-keygen` on the command line or use your existing keypair.

2. Upload the **public** key to R-Forge using the following website: Go to "My Page" and then click on "Account maintenance". At the bottom of this page you click on "edit keys" in the "Shell Account Information" window (**Important note:** you need to be a project admin or member of a project to do this, otherwise there won't be an option "Shell Account Information", see section 2.2 for joining a project or section 2.3 for registering a project). The key you enter here is typically of the form:

   ```
   ssh-dsa AAAA... foo@bar
   ```

   The first field describes the type of key, the second field is the key itself, and the third field is a comment. **It is important that there are no newlines within a key**.

3. Now you have to wait until the next full hour (the key gets activated).

4. Finally check out the repository using the URL given on the project website (tab "SCM") under "developer account" using `svn checkout`. (Note: if this last step fails for some reason, wait an hour and try again. The logins are created once an hour only)

3.2.3 Mac OS X

Mac OS X users should consult Section 3.2.2 (Linux/Unix) for generating and uploading keypairs. However, alternativ GUI applications may be used to manage SVN repositories which is described here.

1. Create a keypair and upload the public key to R-Forge as explained in Section 3.2.2.

2. Wait until the next full hour so that the key gets activated.

3. In order to manage your SVN repository in the *Finder*, install **SCPlugin**. To install it follow the instructions at `http://scplugin.tigris.org`. Once SCPlugin is installed, navigate in the Finder to the directory where you want to place the local copy of your SVN repository. Create a folder with the name of your project, right click on this folder and select *Subversion -¿ Checkout*. Indicate the URL given on the project website (tab "SCM") under *Developer Subversion Access via SSH*.

4. Alternatively, download and install the **svnX** client from `http://www.apple.com/downloads/macosx/development_tools/svnx.html`. It has a reasonable GUI, provides parallel views into Subversion repositories and working directories, supports all SVN-like operations, and can also handle keypairs with password. More information for svnX installation on Mac OS X is given on the page `http://www.wikihow.com/Install-Subversion-on-Mac-OS-X`.

4 R Package Development

R-Forge aims to provide a platform for collaborative development of R packages, R-related software or further projects. Especially providing good infrastructure for the development of R packages is a main mission of the R-Forge Administration and Development Team. Packages committed to the SVN repositories are built and checked on multiple platforms, including Linux, Windows (32 bit) and Mac OS.

4.1 Building and Checking

To build your packages, simply place your package or, alternatively, multiple packages in the `pkg/` directory in your SVN repository (see Section 3.2 on how to achieve this) and check it in. Typically this directory contains either the R package with the usual `DESCRIPTION` file and `R/`, `man/`, `data/` directories (see R Development Core Team, 2011, for more details) or contains two or more directories containing the actual package sources, i.e., you can build **more than one package** by putting the packages in subdirectories, e.g. `pkg/foo/`, `pkg/bar/`, etc. Note however, that anonymous access to the source code repository has to be enabled for the above to work (see Section 5.2).

Packages are automatically checked out every night (Central European Time), built and checked on several platforms (make sure that the `DESCRIPTION` files are well-formed and contain a proper package name, otherwise your package may not get checked out). All packages successfully checked out appear on the project home page in the R packages tab. Project members also have access to the **check and build logs** on all supported platforms.

Furthermore, these packages are made available in a CRAN-style repository—the R-Forge package repository. This allows packages to be installed simply by typing `install.packages("foo", repos = "http://R-Forge.R-project.org")` on the R command prompt.

To install dependencies from other repositories as well use e.g., `install.packages("mypackage", repos = c("http://R-Forge.R-project.org", "http://your.nearest.CRAN.mirror.org"), dep = TRUE)`

4.2 External SVNs

Developers who wish to use their own external package repository can use the **external SVN** feature in the R packages Admin tab (follow the instructions on the site). Please note that only repositories with anonymous access and using the R-Forge repository structure (i.e., `pkg` is available in the repositories) are supported. It is possible to use external and internal SVN repositories side-by-side. Regardless of origin, all packages will show up on the R packages tab.

If you decided to migrate your repository to R-Forge please go to Section 6.

4.3 Submit Packages to CRAN

Packages can be **submitted easily to CRAN** through the web interface. Note that you have to be project Administrator which indicates maintainership of the package to achieve this.

5 Additional Features

5.1 Project Homepage

Each project on R-Forge has its own homepage with the URL `http://foo.r-forge.r-project.org/` where `foo` is the project's Unix name. Use the pre-defined `www` directory in your SVN repository to create or modify your homepage. Note that it will be checked out daily, so please take into consideration that it will not be available right after you commit your changes or updates. Please also note that only html or xhtml will be allowed in the future.

5.2 SCM Options

Two options can be accessed through the SCM Admin tab:

1. Enable anonymous access Self-explanatory. If disabled, only project members can access the SVN repository and **packages will not be checked out for automatic building**. The project homepage will still be available though. Default is enabled.
2. Delivery of Commit Messages This allows you to direct commit messages to an email address of your choice, e.g. the project's commit mailing list (see below). Default is disabled.

5.3 Mailing Lists

R-Forge provides a mailing list service for all projects accessible through the Lists tab.

A list named `foo-commits` where `foo` is the project's unix name is already created with each project and can be used for distributing commit messages (see Section 5.2 on how to enable this feature). Project administrators can create and manage additional lists for their project.

Furthermore, **searching the lists** is provided by the Swish-e engine and can be accessed via the List tab. Please note that private lists are not included in the search index.

6 Migration from an Existing CVS/SVN Repository

Steps to include the complete history of your current repository in your R-Forge project:

1. Register a project on R-Forge first.
2. If you use CVS please convert from CVS to SVN (see e.g., `http://www.xs4all.nl/~carlo17/svn/cvs2svn.html` on how to do this).
3. Then dump the whole SVN repository (using `svnadmin dump > foo.dump`).
4. Verify that the dump file can be loaded into a freshly created SVN repository (`svnadmin load newrep < foo.dump`).
5. Send it to us: If the dump file is smaller than 10 MB you can send it via email to `R-Forge@R-project.org` otherwise you have to provide it somewhere for download.
6. The last step includes to `svn move` your package to the /pkg directory if this has not been done automatically.

7 Acknowledgements

Setting up this project would not have been possible without Douglas Bates and the University of Wisconsin as they provided us with a server for hosting this platform. Furthermore, the authors would like to thank IT Services of the WU Wirtschaftsuniversität Wien for their support and for providing us with additional hardware as well as a professional server infrastructure. Thanks to Philippe Grosjean and Hans Werner Borchers for comments on using SVN on Mac OS X (Section 3.2.3).

References

Per Cederqvist et al. *Version Management with* **CVS**. Network Theory Limited, Bristol, 2006. Full book available online at `http://ximbiot.com/cvs/manual/`.

Tom Copeland, Roland Mas, Ken McCullagh, Tim Perdue, Guillaume Smet, and Reinhard Spisser. *GForge Manual*, 2006. URL `http://gforge.org/docman/view.php/1/34/gforge_manual.pdf`.

Kurt Hornik. The R FAQ, 2010. URL `http://CRAN.R-project.org/doc/FAQ/R-FAQ.html`. ISBN 3-900051-08-9.

C. Michael Pilato, Ben Collins-Sussman, and Brian W. Fitzpatrick. *Version Control with* **Subversion**. O'Reilly, 2004. Full book available online at `http://svnbook.red-bean.com/`.

R Development Core Team. *Writing R Extensions*. R Foundation for Statistical Computing, Vienna, Austria, 2011. URL `http://www.R-project.org`. ISBN 3-900051-11-9.

Stefan Theußl and Achim Zeileis. Collaborative software development using R-Forge. *The R Journal*, 1(1):9–14, May 2009. URL `http://journal.r-project.org/2009-1/RJournal_2009-1_Theussl+Zeileis.pdf`.