V Skills

Skills for a secure future

Certified Redmine Project Management Professional Sample Material

V-Skills Certifications

A Government of India

&

Government of NCT Delhi Initiative

V-Skills

Skills for a secure future

1. INSTALLATION

1.1. Installing Redmine

This is the installation documentation for Redmine 1.4.0 and higher. You can still read the document for 1.3.x here.

Requirements
Operating system
Redmine should run on most Unix, Linux, Mac, Mac Server and Windows systems as long as Ruby is available on this platform. See specific installation HowTos here.

Ruby interpreter
The required Ruby versions for a given Redmine version is:

Redmine version	Supported Ruby versions	Rails version used
current trunk	ruby 1.8.72, 1.9.2, 1.9.3, 2.0.01, jruby-1.7.6	Rails 3.2
2.4 2.5	ruby 1.8.72, 1.9.2, 1.9.3, 2.0.01, jruby-1.7.6	Rails 3.2

Ruby 2.1 on Rails 3.2 has a bug.
MRI 1.9.3p327 contains a bug breaking plug-in loading under Windows which 1.9.3p194 or 1.9.3p392 hasn't.

- ✓ At time of writing (3/19/2013), SQL Server support is reported broken with ruby 2.0.0 under Windows because of a database adapter gem incompatibility
- ✓ Ruby MRI 1.8.7 support has reached its EOL and its use is discouraged. See Important: Ruby 1.8.7 out of support and #14371 for additional information.

Supported database back-ends
- ✓ MySQL 5.0 or higher
  - ✓ make sure to install the C bindings for Ruby that dramatically improve performance. You can get them by running gem install mysql2.

- ✓ PostgreSQL 8.2 or higher
  - ✓ make sure your database datestyle is set to ISO (Postgresql default setting). You can set it using: ALTER DATABASE "redmine_db" SET datestyle="ISO,MDY";
  - ✓ some bugs in PostgreSQL 8.4.0 and 8.4.1 affect Redmine behavior (#4259, #4314), they are fixed in PostgreSQL 8.4.2
  - ✓
- ✓ Microsoft SQL Server 2008 or higher (since Redmine 2.3.0)
- ✓ SQLite 3 (not for multi-user production use!)

Optional components

- ✓ SCM binaries (eg. svn), for repository browsing (must be available in your PATH). See RedmineRepositories for SCM compatibility and requirements.
- ✓ ImageMagick (to enable Gantt export to PNG image and thumbnails generation).
- ✓ Ruby OpenID Library (to enable OpenID support). Version 2 or greater is required.

Redmine Version

It is recommended that the majority of users install the proper point releases of redmine. Redmine currently releases a new version every 6 months, and these releases are considered very usable and stable. It is not recommended to install redmine from trunk, unless you are deeply familiar with Ruby on Rails and keep up with the changes - Trunk does break from time-to-time.

MySQL

```
CREATE DATABASE redmine CHARACTER SET utf8;
CREATE USER 'redmine'@'localhost' IDENTIFIED BY 'my_password';
GRANT ALL PRIVILEGES ON redmine.* TO 'redmine'@'localhost';
```

For versions of MySQL prior to 5.0.2 - skip the 'create user' step and instead:\

```
GRANT ALL PRIVILEGES ON redmine.* TO 'redmine'@'localhost' IDENTIFIED BY
'my_password';
```

PostgreSQL

```
CREATE ROLE redmine LOGIN ENCRYPTED PASSWORD 'my_password'
NOINHERIT VALID UNTIL 'infinity';
CREATE DATABASE redmine WITH ENCODING='UTF8' OWNER=redmine;
```

SQL Server

The database, login and user can be created within SQL Server Management Studio with a few clicks.

If you prefer the command line option with SQLCMD, here's some basic example:

Step 3 - Database connection configuration

Copy config/database.yml.example to config/database.yml and edit this file in order to configure your database settings for "production" environment.

Example for a MySQL database using ruby 1.8 or jruby:

```
production:
  adapter: mysql
  database: redmine
  host: localhost
  username: redmine
  password: my_password
```

Example for a MySQL database using ruby 1.9 (adapter must be set to mysql2):

```
production:
  adapter: mysql2
  database: redmine
  host: localhost
  username: redmine
  password: my_password
```

If your server is not running on the standard port (3306), use this configuration instead:

```
production:
  adapter: mysql
  database: redmine
  host: localhost
  port: 3307
  username: redmine
  password: my_password
```

Example for a PostgreSQL database (default port):

```
production:
  adapter: postgresql
  database: <your_database_name>
  host: <postgres_host>
  username: <postgres_user>
  password: <postgres_user_password>
  encoding: utf8
  schema_search_path: <database_schema> (default - public)
```

Example for a SQL Server database (default host localhost, default port 1433):

```
production:
  adapter: sqlserver
  database: redmine
  username: redmine # should match the database user name
  password: redminepassword # should match the login password
```

Step 4 - Dependencies installation
Redmine uses Bundler to manage gems dependencies.

You need to install Bundler first:

```
gem install bundler
```

Then you can install all the gems required by Redmine using the following command:

```
bundle install --without development test
```

Optional dependencies
RMagick (allows the use of ImageMagick to manipulate images for PDF and PNG export)

If ImageMagick is not installed on your system, you should skip the installation of the rmagick gem using:

```
bundle install --without development test rmagick
```

If you have trouble installing rmagick on Windows, refer to this HowTo.

Database adapters
Redmine automatically installs the adapter gems required by your database configuration by reading it from the config/database.yml file (eg. if you configured only a connection using the mysql2 adapter, then only the mysql2 gem will be installed).

Don't forget to re-run bundle install --without development test ... after adding or removing adapters in the config/database.yml file!

Additional dependencies (Gemfile.local)
If you need to load gems that are not required by Redmine core (eg. Puma, fcgi), create a file named Gemfile.local at the root of your redmine directory. It will be loaded automatically when running bundle install.

Example:

```
# Gemfile.local
gem 'puma'
```

Step 5 - Session store secret generation
This step generates a random key used by Rails to encode cookies storing session data thus preventing their tampering.
Generating a new secret token invalidates all existing sessions after restart.

✓ with Redmine 1.4.x:

```
rake generate_session_store
```

✓ with Redmine 2.x:

```
rake generate_secret_token
```

Step 6 - Database schema objects creation
Create the database structure, by running the following command under the application root directory:

```
RAILS_ENV=production rake db:migrate
```

Windows syntax:

```
set RAILS_ENV=production
rake db:migrate
```

It will create tables by running all migrations one by one then create the set of the permissions and the application administrator account, named admin.

Ubuntu troubleshooting:

If you get this error with Ubuntu:

```
Rake aborted!
no such file to load -- net/https
```

Then you need to install libopenssl-ruby1.8 just like this: apt-get install libopenssl-ruby1.8.

Step 7 - Database default data set
Insert default configuration data in database, by running the following command:

```
RAILS_ENV=production rake redmine:load_default_data
```

Redmine will prompt you for the data set language that should be loaded; you can also define the REDMINE_LANG environment variable before running the command to a value which will be automatically and silently picked up by the task.

Unices:

```
RAILS_ENV=production REDMINE_LANG=fr rake redmine:load_default_data
```

Windows:

```
set RAILS_ENV=production
set REDMINE_LANG=fr
rake redmine:load_default_data
```

Step 8 - File system permissions
NB: Windows users can skip this section.

The user account running the application must have write permission on the following subdirectories:

- ✓ files (storage of attachments)
- ✓ log (application log file production.log)
- ✓ tmp and tmp/pdf (create these ones if not present, used to generate PDF documents among other things)
- ✓ public/plugin_assets (assets of plugins)

E.g., assuming you run the application with a redmine user account:

```
mkdir -p tmp tmp/pdf public/plugin_assets
sudo chown -R redmine:redmine files log tmp public/plugin_assets
sudo chmod -R 755 files log tmp public/plugin_assets
```

Step 9 - Test the installation
Test the installation by running WEBrick web server:

- ✓ with Redmine 1.4.x:

```
ruby script/server webrick -e production
```

✓ with Redmine 2.x:

```
ruby script/rails server webrick -e production
```

Once WEBrick has started, point your browser to http://localhost:3000/. You should now see the application welcome page.

Webrick is not suitable for production use, please only use webrick for testing that the installation up to this point is functional. Use one of the many other guides in this wiki to setup redmine to use either Passenger (aka mod_rails), FCGI or a Rack server (Unicorn, Thin, Puma, hellip;) to serve up your redmine.

Step 10 - Logging into the application

Use default administrator account to log in:

✓ login: admin
✓ password: admin

You can go to Administration menu and choose Settings to modify most of the application settings.

Configuration

Redmine settings are defined in a file named config/configuration.yml.

If you need to override default application settings, simply copy config/configuration.yml.example to config/configuration.yml and edit the new file; the file is well commented by itself, so you should have a look at it.

These settings may be defined per Rails environment (production/development/test).

Important: don't forget to restart the application after any change.

Email / SMTP server settings

Email configuration is described in a dedicated page.

SCM settings

This configuration section allows you to:

✓ Override default commands names if the SCM binaries present in the PATH variable doesn't use the standard name (Windows .bat/.cmd names won't work)
✓ Specify the full path to the binary

Examples (with Subversion):

Command name override:

```
scm_subversion_command: "svn_replacement.exe"
```

Absolute path:

```
scm_subversion_command: "C:\Program Files\Subversion\bin\svn.exe"
```

Attachment storage settings
You can set a path where Redmine attachments will be stored which is different from the default 'files' directory of your Redmine instance using the attachments_storage_path setting.

Examples:
```
attachments_storage_path: /var/redmine/files
attachments_storage_path: D:/redmine/files
```

Logging configuration

Redmine defaults to a log level of :info, writing to the log subdirectory. Depending on site usage, this can be a lot of data so to avoid the contents of the logfile growing without bound, consider rotating them, either through a system utility like logrotate or via the config/additional_environment.rb file.

To use the latter, copyconfig/additional_environment.rb.example to config/additional_environment.rb and add the following lines. Note that the new logger defaults to a high log level and hence has to be explicitly set to info.

```
#Logger.new(PATH,NUM_FILES_TO_ROTATE,FILE_SIZE)
config.logger = Logger.new('/path/to/logfile.log', 2, 1000000)
config.logger.level = Logger::INFO
```

Backups
Redmine backups should include:
- ✓ Data (stored in your redmine database)
- ✓ Attachments (stored in the files directory of your Redmine install)

Here is a simple shell script that can be used for daily backups (assuming you're using a mysql database):
```
# Database
/usr/bin/mysqldump -u <username> -p<password> <redmine_database> | gzip >
/path/to/backup/db/redmine_`date +%y_%m_%d`.gz

# Attachments
rsync -a /path/to/redmine/files /path/to/backup/files
```

Notes on Linux/Unix installation
Be sure to disable security hardenning tools during the installation process if you run into bizarre permission problems. These problems are mostly silent and can be caused by tools like extended ACLs, SELinux, or AppArmor. There tools are mostly used in big companies with a strict security policy, default Linux/Unix distributions settings shouldn't be a problem.

Notes on Windows installation
There is an prebuilt installer of Ruby MRI available from http://rubyinstaller.org.
After installing it, select Start Command Prompt with Ruby in the start menu.

Specifying the RAILS_ENV environment variable:

When running command as described in this guide, you have to set the RAILS_ENV environment variable using a separate command.

I.e. commands with the following syntaxes:

```
RAILS_ENV=production <any commmand>

<any commmand> RAILS_ENV=production
```

have to be turned into 2 subsequent commands:

```
set RAILS_ENV=production
<any commmand>
```

MySQL gem installation issue:

You may need to manually install the mysql gem using the following command:

```
gem install mysql
```

And in some case it is required to copy the libmysql.dll file in your ruby/bin directory.
Not all libmysql.dll are ok this seem to works
http://instantrails.rubyforge.org/svn/trunk/InstantRails-win/InstantRails/mysql/bin/libmySQL.dll.

Important note for Win7 and later
On Win7 and later, localhost is commented out in the hosts file1 and IPV6 is the default2. As the mysql2 gem does no support IPV6 addresses3, a connection can't be established and you get the error "Can't connect to MySQL server on 'localhost' (10061)".

You can confirm this by pinging localhost, if ping targets "::1:" IPV6 is being used.

Email Configuration

Configuration Directives
This page is a work in progress, the following configuration directives are only a partial list. Please consult Action Mailer Configuration
for detailed information.

authentication
The type of authentication method expected by your service provider.

Valid settings:
- ✓ Nil (or omit the key) for no authentication
- ✓ : plain
- ✓ : login
- ✓ : cram_md5

(Note: if you set this to nil or omit it, you must not include the user_name and password settings)

delivery_method
The mail transport method to be used.

Valid settings:
- ✓ :smtp
- ✓ :sendmail
- ✓ :async_smtp
- ✓ :async_sendmail

Asynchronous delivery_methods
The :async_smtp and :async_sendmail use asynchronous sends, which means Redmine does not wait for the email to be sent to display the next page. See Asynchronous Email Delivery for more details. Some SMTP servers have delay period before processing takes place as anti-spam feature, during which time synchronous method will block Redmine (10 seconds could be default value, see also #11376 for more information) .

With this delivery method, smtp configuration is specified using async_smtp_settings keyword:

```
development:
  email_delivery:
    delivery_method: :async_smtp
    async_smtp_settings:
        ...
```

Example configuration.yml Configurations
Simple Login Authentication (default settings)

```
# Outgoing email settings

production:
  email_delivery:
    delivery_method: :smtp
    smtp_settings:
      address: smtp.example.net
      port: 25
      domain: example.net
      authentication: :login
      user_name: redmine@example.net
      password: redmine

development:
  email_delivery:
    delivery_method: :smtp
    smtp_settings:
      address: 127.0.0.1
      port: 25
      domain: example.net
      authentication: :login
      user_name: redmine@example.net
      password: redmine
```

If you want to use GMail/Google Apps and other TLS-requiring SMTP servers, you'll have to add some TLS-related settings :

```
production:
  email_delivery:
    delivery_method: :smtp
    smtp_settings:
      enable_starttls_auto: true
      address: "smtp.gmail.com"
      port: '587'
      domain: "smtp.gmail.com"
      authentication: :plain
      user_name: "your_email@gmail.com"
      password: "your_password"
```

Here is an example for Office 365 users (Exchange online). The sender must have an account, or if you want to send from a shared mailbox, the account below must have authorization to "Send As" the sender which is defined in Redmine email notifications settings.

```
production:
  email_delivery:
    delivery_method: :smtp
    smtp_settings:
      enable_starttls_auto: true
      address: "smtp.office365.com"
      port: '587'
      domain: "your_domain.com"
      authentication: :login
      user_name: "email@your_domain.com"
      password: "password"
```

However, this will only work with "recent" enough ruby and rails versions (1.8.7 patchset 2xx and 2.3.5).
(See #5814)

No Authentication
Example for an SMTP service provider with no authentication.

```
production:
  email_delivery:
    delivery_method: :smtp
    smtp_settings:
      address: smtp.knology.net
      port: 25
      domain: cybersprocket.com
```

Using sendmail command
Example for a unix system that uses the /usr/sbin/sendmail command.

```
production:
  email_delivery:
    delivery_method: :sendmail
```

Troubleshooting

Error: "hostname was not match with the server certificate"
If you get this error, there's probably a problem verifying the SSL certificate of your smtp relay. As a temporary fix, you can set this option in the appropriate "email_delivery" section:

```
enable_starttls_auto: false
```

Error: "Mail failure - no recipient addresses"
When this error happens, the notification is not delivered to the destination address. Instead, you will probably receive a rejection message on your sender address, where you can see the header of the message sent, containing "From:" fields but not containing any "To:" fields.

This error is common on Debian boxes, due to the way exim4 is configured by default, which is "-i -t". This configuration tells exim4 that the destination address is inside the header of the message. Instead, we need to configure exim4 so that the destination address will be retrieved from the command line.

The solution consists on editing your config/configuration.yml and making sure you define arguments containing the string "-i", as shown below:

```
# default configuration options for all environments
default:
  email_delivery:
    delivery_method: :sendmail
    sendmail_settings:
      arguments: "-i"
```

The example above employs :sendmail method, which employs sendmail_settings. In case you are using :smtp or :async_smtp methods, try smtp_settings instead.

Error: "Timeout:Error" due to SSL SMTP server connection
add an ssl option to the configuration.yml #17239

```
default:
  # Outgoing emails configuration (see examples above)
  email_delivery:
    delivery_method: :smtp
  smtp_settings:
    address: smtp.domain.com
    port: 465
    ssl: true
    enable_starttls_auto: true
```

```
domain: domain.com
authentication: :login
user_name: redmine@domain.com
password: xxxx
```

1.2. Upgrading an Existing Installation

Step 1 - Check requirements
The first step to upgrading Redmine is to check that you meet the requirements for the version you're about to install.

Step 2 - Backup
It is recommended that you backup your database and file uploads. Most upgrades are safe but it never hurts to have a backup just in case.

Backing up the files
All file uploads are stored to the files/ directory. You can copy the contents of this directory to another location to easily back it up.

MySQL database
The mysqldump command can be used to backup the contents of your MySQL database to a text file. For example:

```
/usr/bin/mysqldump -u <username> -p<password> <redmine_database> | gzip >
/path/to/backup/db/redmine_`date +%y_%m_%d`.gz
```

SQLite database
SQLite databases are all contained in a single file, so you can back them up by copying the file to another location.

PostgreSQL
The pg_dump command can be used to backup the contents of a PostgreSQL database to a text file. Here is an example:

```
/usr/bin/pg_dump -U <username> -Fc --file=redmine.sqlc <redmine_database>
```

Step 3 - Perform the upgrade
Now it's time to perform the actual upgrade. This process is different depending on how you downloaded Redmine. You only need to perform one of the following options.

Option 1 - Downloaded release (tar.gz or zip file)

✓ Uncompress the new program archive in a new directory.

✓ Copy your database settings-file config/database.yml into the new config directory. If you're running Redmine >= 1.4 with mysql and ruby1.9, change the database adapter to `mysql2`.

- ✓ Copy your base configuration settings-file config/configuration.yml into the new config directory.

- ✓ Or, if upgrading from version older than 1.2.0, copy your email settings from your config/email.yml into the new config/configuration.yml file that can be created by copying the available configuration.yml.example file.

- ✓ Copy the files directory content into your new installation (this directory contains all your uploaded files).

- ✓ Copy the folders of your custom installed plugins from your vendor/plugins directory (if upgrading from a version prior to 2.0.0) or plugins directory (else, upgrading from a version >= 2.0.0) into new installation plugins directory. Make sure that you copy only plugins that are were not initially bundled with your previous Redmine setup.

- ✓ Install the required gems by running:

```
bundle install --without development test
```

If ImageMagick is not installed on your system, you should skip the installation
of the rmagick gem using:

```
bundle install --without development test rmagick
```

Only the gems that are needed by the adapters you've specified in your database configuration file are actually installed (eg. if your config/database.yml uses the 'mysql2' adapter, then only the mysql2 gem will be installed). Don't forget to re-run `bundle install` when you change config/database.yml for using other database adapters.

If you need to load some gems that are not required by Redmine core (eg. fcgi), you can create a file named Gemfile.local at the root of your redmine directory. It will be loaded automatically when running `bundle install`.

- ✓ Run the following command from your new Redmine root directory:

```
rake generate_secret_token
```

This will generate a file (config/initializers/secret_token.rb) with a random secret used to secure session data.

- ✓ Check for any themes that you may have installed in the public/themes directory. You can copy them over but checking for updated version is ideal.

VERY IMPORTANT: do NOT overwrite config/settings.yml with the old one.

Option 2 - Upgrading from a SVN checkout

✓

```
svn update
```

- ✓ Updating gems

  run the following command:

  ```
  bundle update
  ```

- ✓ If you are upgrading from an older version to 0.8.7+ or from the trunk version of Redmine to r2493 or above, you must generate a secret for cookie store. See the note at the bottom about generating a session_store.

Step 4 - Update the database
This step is the one that could change the contents of your database. Go to your new redmine directory, then migrate your database:

```
rake db:migrate RAILS_ENV=production
```

If you have installed any plugins, you should also run their database migrations:

```
rake redmine:plugins:migrate RAILS_ENV=production
```

Step 5 - Clean up

- ✓ You should clear the cache and the existing sessions:

  ```
  rake tmp:cache:clear
  rake tmp:sessions:clear
  ```

- ✓ Restart the application server (e.g. puma, thin, passenger)

- ✓ Finally go to "Admin -> Roles & permissions" to check/set permissions for the new features, if any.

Common issues
Errors with repository management
There were several new features added to the reposman.rb file, make sure you have a group specified if you're having issues (–group=groupnamehere). Also, make sure you follow the instructions here again if you only copied your Redmine.pm, and update your Apache configuration as the recommended configuration has changed.

Generating a new secret_token.rb
Before 2.0.0, a session_store.rb file needed to be generated in Redmine's config directory for the cookie based sessions to work.

Starting from 2.0.0, the session_store.rb file should not exist. Instead, the following command will generate the secret_token.rb file:

```
rake generate_secret_token
```

The code repository for Redmine does not contain the config/initializers/secret_token.rb file, it is created by the above rake command.

Error about member_roles
If you have had a failed upgrade/migration in the past then you may have a member_roles and/or group_users table already created. The db migration noted above will fail. Simply rename the tables by logging into MySQL command line and executing the rename table command:

```
mysql> rename table member_roles to member_roles_saved
mysql> rename table groups_users to groups_users_saved
```

Error about "undefined method `add_frozen_gem_path'"
If you see this error, check if the directory vendor/rails exists and remove or rename it if it does, it might have an old RoR version.

1.3. Migrating from Other Systems

Trac
The Trac importer migrates:

- ✓ Users
- ✓ Components
- ✓ Milestones
- ✓ Tickets
- ✓ Ticket comments and changes (status and resolution)
- ✓ Trac specific fields (eg. Resolution) are added as custom fields in Redmine
- ✓ Ticket files and custom fields
- ✓ Wiki pages with history

Notes:
- ✓ User passwords are all set to trac
- ✓ Ticket ids are preserved if there's no ticket in your Redmine database yet
- ✓ Custom fields are all created as text custom fields in Redmine

You need the sqlite-ruby gem to access the Trac database.

- For sqlite:

```
gem install sqlite-ruby
```

- For sqlite3:

```
gem install sqlite3-ruby
```

Before starting, you need a fresh Redmine database, with default data loaded (required). See Redmine installation.

The script was tested with a 0.10 Trac sqlite database and a 0.11 Trac sqlite3 database.

- ✓ Run the following command, where test is your Redmine target environment:

```
rake redmine:migrate_from_trac RAILS_ENV="test"
```

✓ The script (source:trunk/lib/tasks/migrate_from_trac.rake) asks you for your Trac settings:

```
Trac directory []: /var/trac/myproject
Trac database adapter (sqlite, sqlite3, mysql, postgresql) [sqlite]:
Database encoding [UTF-8]:
Target project identifier []: myproject
```

Trac directory is the root directory of your Trac environment. Redmine will look in this directory for db/trac.db (if you choose sqlite/sqlite3 database) and the attachments directory.
When using a mysql or postgresql Trac database, the script will ask you for the connection parameters (host, database name, username and password).
Target project identifier is the identifier of the Redmine project in which the data will be loaded (the project is created if not found).

✓ The script migrates your data:

```
Deleting data
Migrating components..............................
Migrating milestones..............
Migrating custom fields.......
Migrating tickets.................................
Migrating wiki...........

Components: 29/30
Milestones: 14/14
Tickets: 1275/1275
Ticket files: 106/106
Custom values: 4409/4409
Wiki edits: 102/102
```

The script gives you the total number of migrated objects.

Now, you should see a project called Myproject in Redmine with your Trac data loaded in this project.

Mantis

The Mantis importer migrates:
✓ Users
✓ Projects
✓ Project versions, categories and news
✓ Project memberships
✓ Bugs
✓ Bug notes, files, relations and monitors
✓ Custom fields

User passwords are all set to "mantis".

Bug files migration only works if they're stored in your Mantis database (this is the default Mantis behaviour).

The script was tested with different 1.0.x Mantis databases and should work with any other recent versions.

Before starting, you need a fresh Redmine database, with default data loaded (required). See Redmine installation. When migrating into filled Redmine database, you can use Ulrichs Non-destructive migration Script. Make sure to apply the patch #10504.

✓ Run the following command, where test is your Redmine target environment:

```
rake redmine:migrate_from_mantis RAILS_ENV="test"
```

✓ The script asks you for your Mantis database settings:

```
Please enter settings for your Mantis database
adapter [mysql]:
host [localhost]:
database [bugtracker]: mantis
username [root]:
password []:
encoding [UTF-8]:
```

Give the adapter, host name, database name, login, password and encoding of your Mantis database, or leave the default values.

The adapter can be mysql (default) or postgresql.

✓ The script migrates your data:

```
Migrating users...............
Migrating projects.............
Migrating bugs........................................
Migrating news...
Migrating custom fields..

Users: 15/15
Projects: 13/13
Memberships: 10/10
Versions: 33/33
Categories: 4/4
Bugs: 180/180
Bug notes: 336/336
Bug files: 46/46
Bug relations: 11/11
Bug monitors: 8/8
News: 3/3
Custom fields: 2/2
```