leihs administration and installation guide

COLLABORATORS			
	TITLE : leihs administration and installation guide		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Ramon Cahenzli	April 18, 2012	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Introduction

leihs is web-based inventory handling and resource booking system. It allows users to view available equipment and place reservations through the frontend. Inventory managers and sysadmins use the backend to handle incoming reservations and manage items in the inventory.

This guide shows you how to install a leihs server. The guide is written from the perspective of a system administrator or developer. If you are interested in running leihs in your own organization but aren't a sysadmin, talk to your IT department. leihs is not intended to be installed on a client, so any software installation on your own machine isn't necessary. All you need is a web browser.

Consulting and installation services are also available from independent companies supporting Free Software all around the world. Ask around for a company or individual who knows Ruby on Rails applications, you will surely find someone who can help you install leihs.

If you are such a person yourself and would like to have your services listed on the leihs project website and in this document, please write me an e-mail.

2 Quick Install

This section is meant for admins experienced with installing Ruby on Rails applications. It lists the necessary installation steps in the briefest possible way, so you can get up and running quickly.

2.1 Installation on Debian GNU/Linux

These instructions were tested on a minimal install of Debian GNU/Linux 5.0 (Lenny) and Debian GNU/Linux 6.0 (Squeeze). They might also work on Ubuntu. You might have to substitute `sudo su` for `su` because Ubuntu does not configure a root password, thus `su` would not work.

1. Install Ruby, irb, libxslt, Cairo, MySQL client libraries, libxml2:

   ```
   # apt-get install ruby irb rdoc libopenssl-ruby ruby-dev libxslt-dev libcairo2-dev lib
   ```

2. Install RubyGems from the RubyGems website. Make sure **not to install** the edition of RubyGems that is available from Debian's package archives. RubyGem development moves so quickly that we need to use the one from upstream.

   ```
   # cd /tmp
   # wget http://production.cf.rubygems.org/rubygems/rubygems-1.5.2.tgz
   # tar xvfz rubygems-1.5.2.tgz
   # cd rubygems-1.5.2
   # ruby setup.rb
   # ln -s /usr/bin/gem1.8 /usr/bin/gem
   ```

   Note that the URL above might change! Please visit the RubyGems site to find the exact URL under "Downloads". It is important that you use version 1.5.2 or 1.5.3 or RubyGems because newer versions don't work with Rails 2.3.5, which leihs uses.

3. Install the MySQL header files and the MySQL gem:

   ```
   # apt-get install libmysqlclient15-dev make build-essential
   # gem install mysql
   ```

4. Download the latest version of leihs from our SourceForge project page. Unpack it to a convenient directory. We use the home directory of the *leihs* user (/home/leihs) to install leihs in. Of course you can use any directory.

5. Download Sphinx (a fulltext search system) and install thinking-sphinx (a gem). In this example we also include libstemmer, a library that allows for word stem searching in various languages. We use version 0.9.9:

```
$ cd /tmp
$ wget http://sphinxsearch.com/downloads/sphinx-0.9.9.tar.gz
$ tar xvfz sphinx-0.9.9.tar.gz
$ cd sphinx-0.9.9
$ wget http://snowball.tartarus.org/dist/libstemmer_c.tgz
$ tar xvfz libstemmer_c.tgz
$ ./configure --with-libstemmer && make
$ su
# make install
```

6. Optional: If you want to use images of inventory items, install ImageMagick:

```
# apt-get install imagemagick libmagickwand-dev
```

7. Install the required version of Rails as well as a few gems that cannot be installed automatically:

```
# cd /home/leihs
# gem install -v=2.3.5 rails
# gem install bundler
# gem install rake -v 0.8.7
# bundle install --deployment --without cucumber
```

8. Configure database access for this installation of leihs. Copy the file config/database.yml.example to config/database.yml and set things up according to your needs. You will need a MySQL database for leihs. Here is an example of a production database configuration:

```
production:
   adapter: mysql
   database: leihs2_production
   encoding: utf8
   username: root
   password:
   host: localhost
   port: 3306
```

9. Create and migrate the database:

```
# su - leihs
$ RAILS_ENV=production bundle exec rake db:migrate
$ RAILS_ENV=production bundle exec rake db:seed
```

10. Create any temporary directories that are necessary for e.g. image uploads, temporary files etc. Make sure to create these directories so that the leihs user has write permission to them.

```
$ cd /home/leihs
$ mkdir -p public/images/attachments
$ mkdir -p tmp/sessions
$ mkdir tmp/cache
$ mkdir -p log
```

11. Configure and start the Sphinx server:

```
$ cd /home/leihs
$ RAILS_ENV=production bundle exec rake ts:config
$ RAILS_ENV=production bundle exec rake ts:reindex
$ RAILS_ENV=production bundle exec rake ts:start
```

12. Start the leihs server:

    ```
    $ cd /home/leihs
    $ RAILS_ENV=production ./script/server
    ```

    Now you should see your local leihs server at http://localhost:3000. You can log in with username "super_user_1" and password "pass".

    This gives you a test setup using the pure Ruby WebRick web server. For production setups, we recommend mod_passenger. See the "Installing a production environment" section of this guide for more information.

    Please change the super_user_1 password immediately after logging in the first time. Otherwise other people will also be able to log in using the well known default password.

13. Set up a system cronjob that sends nightly e-mail reminders and, more importantly, updates all the models' availability counts. There are many ways to schedule repeating tasks on GNU/Linux, but here's a line in crontab-format that you can add to your leihs user's crontab using e.g. `crontab -e`:

    ```
    1 00    * * *   cd /home/leihs && RAILS_ENV=production rake leihs:cron
    ```

    The important bit here is to run the "leihs:cron" rake task. How you do this exactly is irrelevant.

14. Optional: Speed boost thanks to memcached

    You can install memcached in order to make leihs perform faster, especially for activities that require recalculations of item availability. Memcached can speed up the system by several orders of magnitude.

    ```
    # apt-get install memcached
    ```

    Don't forget to set ENABLE_MEMCACHED to "yes" in /etc/defaults/memcached

3 Users, logins and levels

3.1 Default admin username/password

After installation, a default user is created for the Database Authentication module. Username: super_user_1. Password: pass.

+ Please change the super_user_1 password immediately after logging in the first time. Otherwise other people will also be able to log in using the well known default password.

3.2 Hooking up to LDAP for logins

Currently, leihs contains only a very rudimentary LDAP login adapter. It was developed by the Zurich University of the Arts specifically for the Hochschule der Künste Bern, Switzerland, and some parts of it are hardcoded to fit the environment there.

However, it's not impossible to get leihs to authenticate against your own LDAP server if you are willing to modify two files in leihs.

3.2.1 Modifying config/LDAP.yml

Open config/LDAP.yml and adapt the configuration to your own LDAP server:

```
production:
 host: your.ldap.server
 port: 636
 encryption: simple_tls
 base: dc=yourcompany,dc=com
 log_file: log/ldap_server.log
 log_level: warn
 search_field: uid
 bind_dn: *****
 bind_pwd: ******
```

You may have to adapt the `search_field` option to point at the LDAP attribute that contains your usernames. The `search_field` dictates what users will have to write in the "Login" field on login.

3.2.2 Modifying app/controllers/authenticator/ldap_authentication_controller.rb

This controller handles the login itself. You will have to modify a few strings to point at the right attributes for your own LDAP server.

On line 33:

```
email = users.first.mail if users.first.mail
email ||= "#{user}@hkb.bfh.ch"
```

On line 52:

```
u.firstname = users.first["givenname"].to_s
u.lastname = users.first["sn"].to_s
u.phone = users.first["telephonenumber"].to_s unless users.first["telephonenumber"].blank?
```

You can add any Ruby code here that extracts this user information from your LDAP. Currently things are set up to point at the default fields from the InetOrgPerson class (givenname, sn, telephonenumber, etc.). This should be okay for most LDAP servers, but feel free to change the strings in e.g. `users.first["givenname"].to_s` to your own field names. For a field called "firstname", that would change to: `users.first["firstname"].to_s`.

3.2.3 Enabling LDAP authentication in the system

Finally, you need to tell leihs that you want to use LDAP, not local database authentication. Start a Rails console inside your leihs directory:

```
$ RAILS_ENV=production bundle exec ./script/console
```

Then enable LDAP authentication and switch off database authentication:

```
>> ldap = AuthenticationSystem.find_by_class_name("LDAPAuthentication")
>> ldap.is_default = true
>> ldap.is_active = true
>> ldap.save
```

```
>> db = AuthenticationSystem.find_by_class_name("DatabaseAuthentication")
>> db.is_default = false
>> db.is_active = false
>> db.save
```

Now restart your Rails application and next time you try to access it, you should be forwarded to /authenticator/ldap/login instead of /authenticator/db/login, and then you can log in via LDAP.

Do you want to be able to configure all these settings directly in LDAP.yml so it's easier to hook up to your LDAP server? Feel free to improve our LDAP connector and send us a pull request on GitHub. Alternatively, you could also pay some Rails developers (even us!) to develop this feature for you.

Warning: Please make sure that your Rails application server has SSL enabled before you put this configuration into production. You don't want to send passwords unencrypted over the web.

3.3 User levels and roles

leihs decides what it lets users do based on their role and level. Each role or level is specific to an inventory pool, the only exception is the admin role, which covers the whole system.

Therer are three available roles: customer, manager and admin.

3.3.1 Customer

Customers may only use the frontend and submit orders.

3.3.2 Manager level 1

Think of this manager as the "lending and borrowing manager".

- May only use the "Booking" section of the backend and parts of the "Administration" section
- May acknowledge orders for their inventory pool
- May hand over orders and create contracts
- May take back orders

3.3.3 Manager level 2

Think of this manager as a "junior manager".

- Everything that a manager at level 1 can do, plus:
- May assign levels and permissions (within their own inventory pool) up to and including level 2
- May create new models
- May create new items that are not inventory relevant, and the manager may not change this setting
  - These items have "Responsible department" set to their own inventory pool and the manager may not change this setting

3.3.4 Manager level 3

Think of this manager as a "senior manager".

- Everything that levels 1 and 2 can do, plus:
- Sees the "Inventory" section of leihs
- Has no restrictions on editing items
  - May create things that are inventory relevant
  - May assign items to any inventory pool as "Responsible department"
  - Using these functions, managers of level 3 can purchase items using their own inventory pool, but assign responsibility for the items to other inventory pools, so those other pools can manage their borrowing and lending independently
- May manage categories

3.3.5 Admin

- May manage users and assign permissions
- May manage inventory pools
- May manage groups

4 Performing upgrades

4.1 Upgrading from leihs 2.0 to leihs 2.1

The upgrade should be painless.

1. Download the new leihs version (in this example, the .tar.gz version is used) and unpack it to a new location:

   ```
   $ tar xvfz leihs-2.1.tar.gz
   $ cd leihs-2.1
   ```

2. Copy any images uploaded to your old leihs version so they are also available in the new leihs:

   ```
   $ rm -rf public/images/attachments
   $ cp -pr ../leihs-2.0/public/images/attachments public/images/
   ```

3. Copy your old database configuration file:

   ```
   $ cp ../leihs-2.0/config/database.yml config/
   ```

4. Look at your old config/environment.rb file in a text editor. Open the new config/environment.rb. Copy the configuration items from your old config to your new config as necessary. You might find new options in the new environment.rb that weren't present in the old one. Save your new configuration.

5. Run the database migration to get all your data up to the new version:

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

6. Rerun the steps from the section "Configure and start the Sphinx server" relevant for your operating system in order to create/update your fulltext search index and make sure Sphinx is running.

7. Run the server. Make sure you stop your previous leihs server!

   ```
   $ RAILS_ENV=production ./script/server
   ```

If everything went correctly, you should see leihs coming up at http://localhost:3000.

4.2 Upgrading from leihs 2.1 to leihs 2.2

The upgrade should be painless.

1. Download the new leihs version (in this example, the .tar.gz version is used) and unpack it to a new location:

```
$ tar xvfz leihs-2.2.tar.gz
$ cd leihs-2.2
```

2. Copy any images uploaded to your old leihs version so they are also available in the new leihs:

```
$ rm -rf public/images/attachments
$ cp -pr ../leihs-2.1/public/images/attachments public/images/
```

3. Copy your old database configuration file:

```
$ cp ../leihs-2.1/config/database.yml config/
```

4. Look at your old config/environment.rb file in a text editor. Open the new config/environment.rb. Copy the configuration items from your old config to your new config as necessary. You might find new options in the new environment.rb that weren't present in the old one. Save your new configuration.

5. Run the database migration to get all your data up to the new version:

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

6. Run the server. Make sure you stop your previous leihs server!

```
$ RAILS_ENV=production ./script/server
```

If everything went correctly, you should see leihs coming up at http://localhost:3000.

4.3 Upgrading from leihs 2.2 to leihs 2.9

Warning
You must upgrade from 2.2.x to 2.9.1 before continuing on with the 2.9 series. The reason is that with 2.9, we rolled up all previous migrations into one single file to make future migrations faster and easier. The system will warn you if you try to upgrade beyond 2.9.1 without installing 2.9.1 first. The sequence of actions is the same for all 2.9 releases, so once you have 2.9.1, you can just grab e.g. 2.9.4 and follow these instructions again.

The upgrade should be painless, but please note that the access levels were replaced by user groups in this release. This can give you almost the same behavior, but it's different to manage. You now specify groups in the admin interface or your inventory pool's backend, and then specify how many items of each model that each group can have.

If you had existing user levels in your system, these will be automatically converted to user groups during the database migration (db:migrate). This is documented below.

1. Download the new leihs version (in this example, the .tar.gz version is used) and unpack it to a new location:

```
$ tar xvfz leihs-2.9.tar.gz
$ cd leihs-2.9
```

2. Copy any images uploaded to your old leihs version so they are also available in the new leihs:

```
$ rm -rf public/images/attachments
$ cp -pr ../leihs-2.2/public/images/attachments public/images/
```

3. Copy your old database configuration file:

```
$ cp ../leihs-2.2/config/database.yml config/
```

4. Look at your old config/environment.rb file in a text editor. Open the new config/environment.rb. Copy the configuration items from your old config to your new config as necessary. You might find new options in the new environment.rb that weren't present in the old one. Save your new configuration.

5. Install bundler and update rake as root:

```
# gem install bundler
# gem install rake
```

6. Run the database migration to get all your data up to the new version:

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

7. Run the server. Make sure you stop your previous leihs server! If you use mod_passenger, do `touch tmp/restart.txt` instead.

```
$ RAILS_ENV=production bundle exec ./script/server
```

8. Stop the Sphinx server that's running for leihs 2.2.:

```
$ cd leihs-2.2
$ RAILS_ENV=production rake ts:stop
```

9. Reindex Sphinx at the new location and restart the Sphinx server:

```
$ cd leihs-2.9
$ RAILS_ENV=production bundle exec rake ts:config
$ RAILS_ENV=production bundle exec rake ts:reindex
$ RAILS_ENV=production bundle exec rake ts:start
```

If everything went correctly, you should see leihs coming up at http://localhost:3000 or, if using mod_passenger, at the location you configured.

4.3.1 Installation on Mac OS X (not officially supported)

The following steps were tested on Mac OS X 10.5 and 10.6 on an x86. They may also work for later versions.

Running leihs on Mac OS X is unsupported at this time. You may try the following steps, but they are not guaranteed to work and no longer updated since Apple stopped producing servers. If you manage to get leihs to work well, feel free to update this installation guide. leihs usually isn't the problem, the problems start when installing all the dependencies on OS X. Environment variables, bundler, compiling native extensions etc. is more easily done on a Linux distribution.

Make sure you are running Ruby 1.8.x, not Ruby 1.9.x since leihs is a Rails 2.x project for now and Ruby 1.9.x is not supported by Rails 2.

1. Install Apple Xcode. This is required because some modules may need a C/C++ compiler. XCode is available from Apple after a free developer registration.

2. Install MySQL 5.0. Use the 32-bit version if you are running OS X Leopard (10.5) but use the 64-bit version if you're on Snow Leopard (10.6). **Make absolutely sure to install the 32-bit version on Leopard (10.5) even if you are running a 64-bit operating system.** Some bugs in Mac OS X's MySQL and Ruby integration make it impossible to continue otherwise.

3. Update the local gem system:

```
$ sudo gem update --system 1.5.3
```

   It is important that you use version 1.5.2 or 1.5.3 or RubyGems because newer versions don't work with Rails 2.3.5, which leihs uses. You may need to download a newer version of RubyGems from rubygems.org and then downgrade to 1.5.3 using the command above.

4. Install the native MySQL gem:

   If you are installing on Snow Leopard (OS X 10.6) use the following installation options:

```
$ sudo env ARCHFLAGS='-arch x86_64' gem install \
mysql -- --with-mysql-config=/usr/local/mysql/bin/mysql_config
```

   Please do not copy/paste the above, as that might copy typographic apostrophes instead of single quotes around the "-arch x86_64" option. Please type the lines above into a single line in your terminal instead.

   If you are using Leopard (OS X 10.5) or below, use the following string instead:

```
 $ sudo gem install mysql -- --with-mysql-config=/usr/local/mysql/bin/mysql_config \
--with-mysql-dir=/usr/local/mysql --with-mysql-lib=/usr/local/mysql/lib
\ --with-mysql-include=/usr/local/mysql/include
```

5. Download the latest version of leihs from our SourceForge project page. Unpack it to a convenient directory. We use the home directory of the *leihs* user (/Users/leihs) to install leihs in. Of course you can use any directory.

6. Download Sphinx (a fulltext search system) and install thinking-sphinx (a gem). In this example we also include libstemmer, a library that allows for word stem searching in various languages. We use version 0.9.9:

```
$ cd /tmp
$ wget http://sphinxsearch.com/downloads/sphinx-0.9.9.tar.gz
$ tar xvfz sphinx-0.9.9.tar.gz
$ cd sphinx-0.9.9
$ wget http://snowball.tartarus.org/dist/libstemmer_c.tgz
$ tar xvfz libstemmer_c.tgz
$ ./configure --with-libstemmer && make
$ sudo make install
```

7. Install ImageMagick. On Mac OS X, we recommend using MacPorts and then installing the ImageMagick port:

```
$ sudo port install imagemagick
```

   Make sure that the ImageMagick binaries (e.g. convert) are in your $PATH. Please consult the Mac OS X documentation to find out how to add paths to your $PATH environment variable. It can be done e.g. by editing /etc/profile and adding the following lines:

```
PATH=$PATH:/opt/local/bin
export PATH
```

   But Apple may at any point recommend a different approach for configuring your PATH, so please don't take our word for it.

8. Install the required version of Rails as well as a few gems that cannot be installed automatically:

```
$ cd /home/leihs
$ sudo gem install -v=2.3.5 rails
$ sudo gem install bundler
$ sudo gem install rake -v 0.8.7
$ bundle install --without cucumber
```

9. Configure database access for this installation of leihs. Copy the file config/database.yml.example to config/database.yml and set things up according to your needs. You will need a MySQL database for leihs. Here is an example of a production database configuration:

```
production:
   adapter: mysql
   database: leihs2_production
   encoding: utf8
   username: root
   password:
   host: localhost
   port: 3306
```

10. Create and migrate the database:

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

11. Create any temporary directories that are necessary for e.g. image uploads, temporary files etc. Make sure to create these directories so that the leihs user has write permission to them.

```
$ cd /home/leihs
$ mkdir -p public/images/attachments
$ mkdir -p tmp/sessions
$ mkdir tmp/cache
$ mkdir -p log
```

12. Configure and start the Sphinx server:

```
$ RAILS_ENV=production rake ts:config
$ RAILS_ENV=production rake ts:reindex
$ RAILS_ENV=production rake ts:start
```

13. Start the server:

```
$ RAILS_ENV=production ./script/server
```

    Now you should see your local leihs server at http://localhost:3000. You can log in with username "super_user_1" and password "pass".

    This gives you a test setup using the pure Ruby WebRick web server. For production setups, we recommend mod_passenger. See the "Installing a production environment" section of this guide for more information.

    Please change the super_user_1 password immediately after logging in the first time. Otherwise other people will also be able to log in using the well known default password.

14. Set up a system cronjob that sends nightly e-mail reminders and, more importantly, updates all the models' availability counts. There are many ways to schedule repeating tasks on Mac OS X. Please see Apple's documentation on launchd for how to do this.

    In crontab format, an example job for a production environment would look like this:

```
1 00    * * *   cd /home/leihs && RAILS_ENV=production rake leihs:cron
```

The important bit here is to run the "leihs:cron" rake task. How you do this exactly is irrelevant.

15. Optional: Speed boost thanks to memcached

    You can install memcached in order to make leihs perform faster, especially for activities that require recalculations of item availability. Memcached can speed up the system by several orders of magnitude. Unfortunately, because Apple does not include an official packaging system with Mac OS X , we can't give specific instructions for installing memcached on OS X.

5 Installing a production environment

Please note that this quick-start guide does not cover running a Ruby on Rails application for production. Please look into Phusion Passenger or the Apache Mongrel cluster in the reference section for how to set up a production environment.

Without a real production environment, leihs can handle upwards of 1000 users (not concurrently, of course!). If you need better performance or want an easier to handle setup, you must set up a proper production environment.

6 Free Software Statement

The Zurich University of the Arts supports Free Software as defined by the Free Software Foundation. That's why leihs is Free Software licensed under the GNU GPL version 3.0.

One of the advantages this freedom brings with it is that it enables anyone in the world to provide local support services for leihs at the same quality level as the Zurich University of the Arts can provide itself.

If you would like to take part in the development of leihs, please see our project page.

7 References

8 Appendices

A ZHdK configuration file for mongrel cluster

As an example to help you set up a production environment, here is the Apache configuration we use. You will need mod_proxy:

```
<VirtualHost *:80>
   ServerName ausleihe.zhdk.ch

   DocumentRoot /home/rails/leihs/leihs/public/
   ErrorLog /var/log/apache2/leihs/error.log

   # Which Rails environment to use (production, testing, production)
   DefaultInitEnv RAILS_ENV production
   SetEnv RAILS_ENV production

  # Proxy balancer for leihs
  <Proxy balancer://leihs_cluster>
    BalancerMember http://127.0.0.1:10010
    BalancerMember http://127.0.0.1:10011
    BalancerMember http://127.0.0.1:10012
    BalancerMember http://127.0.0.1:10013
  </Proxy>
```

```
   # We want to completely ignore the application's own
   # .htaccess, as all relevant options are configured
   # right here in this file.
   <Directory /home/rails/leihs/leihs/public>
      AllowOverride none
   </Directory>


   # Don't do forward proxying
   ProxyRequests Off

   # Enable reverse proxying
   <Proxy *>
     Order deny,allow
     Allow from all
   </Proxy>

   RewriteEngine On

   # Check for maintenance file. Let apache load it if it exists
   RewriteCond %{DOCUMENT_ROOT}/system/maintenance.html -f
   RewriteRule . /system/maintenance.html [L]

   # Rewrite index to check for static
   RewriteRule ^/$ /index.html [QSA]

   # Let apache serve static files (send everything via mod_proxy that
   # is *no* static file (!-f)
   RewriteCond %{DOCUMENT_ROOT}%{REQUEST_FILENAME} !-f
   RewriteRule .* balancer://leihs_cluster%{REQUEST_URI} [L,P,QSA]
</VirtualHost>
```

B ZHdK configuration file for mod_passenger

This configuration is recommended for anyone who is already running an Apache or nginx web server.

Download Phusion Passenger and install it according to the installation instructions.

Afterwards, a simple VirtualHost entry will be enough for Apache to pick up your leihs/Rails app:

```
<VirtualHost *:80>
   ServerName your.leihs.example.com
   DocumentRoot /home/leihs/public/

   <Directory /home/leihs/public>
      AllowOverride all
      Options -MultiViews
   </Directory>

</VirtualHost>
```