A Rails keretrendszer
Gyakorlat

Kovács Gábor

2015. szeptember 29.

1. Ruby telepítése, Gemek

A Rails keretrendszert egy virtuális környezetben [1] futó Debian Linuxra [2] telepítjük a gyakorlat folyamán. Ez a dokumentáció kiindulási pontként egy ilyen, grafikus felhasználói környezetet is tartalmazó előtelepített operációs rendszert feltételez.

A Rails környezetünk back-endje hosszú távon egy MySQL adatbázis kezelő lesz. Ezt az `apt-get install mysql-server` parancs konzolon történő kiadásával telepíthetjük, a telepítő többször rá fog kérdezni a root felhasználó jelszavára, azt tetszés szerinti értékre állíthatjuk, azonban jegyezzük meg, mert a Rails adatbázis konfigurációs fájljában ezt meg kell adnunk. A gyakorlatra telepített környezeten az egyszerűség kedvéért üres jelszót fogunk használni.

A Rails adatbázis adapterei és néhány további komponens fordításához szükségünk van C és C++ fordítóra, valamint a fejlesztői függvénykönyvtárakra [3], amelyeket az `apt-get install gcc` és `apt-get install g++` parancsok konzolon való kiadásával telepíthetünk.

A Rails környezet telepítéséhez egy Ruby környezet szükséges, ami mellé kiegészítőként telepítjük az RI dokumentációs rendszert és a natív kiegészítések fordításához szükséges Ruby fejlesztői függvénykönyvtárakat. A gyakorlaton a Ruby stabil, 2.1-s verzióját használjuk, ami előfeltétele az ősszel megjelenő Rails 5-ös változatának telepítésének. Ezt a következő paranccsal telepíthetjük rendszergazdaként:

[1] Oracle VirtualBox (`http://www.virtualbox.org`) az otthonra javasolt virtualizációs eszköz, viszont használható a vmware, illetve a parallels is

[2] Mivel az Ubuntu Linux csomagkezelője megegyezik a Debian Linux disztribúció csomagkezelőjével, a dokumentumban felsorolt csomagnevek egy az egyben átvehetők. Más disztribúciók esetén a csomagok nevei különbözhetnek.

[3] A Rails egyik függvénykönyvtára miatt szükségünk lesz még a zlib1g csomagra is.

```
kovacsg@debian:~# ruby
bash: ruby: command not found
kovacsg@debian:~# gem
bash: gem: command not found
kovacsg@debian:~# irb
bash: irb: command not found
kovacsg@debian:~# sudo bash
[sudo] password for kovacsg:
root@debian:/home/kovacsg# su
root@debian:/home/kovacsg# apt-get install ruby ri
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  javascript-common libjs-jquery libruby2.1 libyaml-0-2 ruby2.1 ruby2.1-doc
      rubygems-integration
Suggested packages:
  ruby-dev bundler
The following NEW packages will be installed:
  javascript-common libjs-jquery libruby2.1 libyaml-0-2 ri ruby ruby2.1
      ruby2.1-doc rubygems-integration
0 upgraded, 9 newly installed, 0 to remove and 6 not upgraded.
Need to get 0 B/7,098 kB of archives.
After this operation, 39.5 MB of additional disk space will be used.
Do you want to continue? [Y/n]
Selecting previously unselected package libyaml-0-2:amd64.
(Reading database ... 149003 files and directories currently installed.)
Preparing to unpack .../libyaml-0-2_0.1.6-3_amd64.deb ...
Unpacking libyaml-0-2:amd64 (0.1.6-3) ...
Selecting previously unselected package javascript-common.
Preparing to unpack .../javascript-common_11_all.deb ...
Unpacking javascript-common (11) ...
Selecting previously unselected package libjs-jquery.
Preparing to unpack .../libjs-jquery_1.7.2+dfsg-3.2_all.deb ...
Unpacking libjs-jquery (1.7.2+dfsg-3.2) ...
Selecting previously unselected package rubygems-integration.
Preparing to unpack .../rubygems-integration_1.8_all.deb ...
Unpacking rubygems-integration (1.8) ...
Selecting previously unselected package libruby2.1:amd64.
Preparing to unpack .../libruby2.1_2.1.5-2+deb8u2_amd64.deb ...
Unpacking libruby2.1:amd64 (2.1.5-2+deb8u2) ...
Selecting previously unselected package ruby2.1.
Preparing to unpack .../ruby2.1_2.1.5-2+deb8u2_amd64.deb ...
Unpacking ruby2.1 (2.1.5-2+deb8u2) ...
Selecting previously unselected package ruby2.1-doc.
Preparing to unpack .../ruby2.1-doc_2.1.5-2+deb8u2_all.deb ...
Unpacking ruby2.1-doc (2.1.5-2+deb8u2) ...
Selecting previously unselected package ri.
Preparing to unpack .../ri_1%3a2.1.5+deb8u1_all.deb ...
Unpacking ri (1:2.1.5+deb8u1) ...
Selecting previously unselected package ruby.
Preparing to unpack .../ruby_1%3a2.1.5+deb8u1_all.deb ...
Unpacking ruby (1:2.1.5+deb8u1) ...
Processing triggers for man-db (2.7.0.2-5) ...
Setting up libyaml-0-2:amd64 (0.1.6-3) ...
Setting up javascript-common (11) ...
Setting up libjs-jquery (1.7.2+dfsg-3.2) ...
Setting up rubygems-integration (1.8) ...
Setting up libruby2.1:amd64 (2.1.5-2+deb8u2) ...
Setting up ruby2.1 (2.1.5-2+deb8u2) ...
Setting up ruby2.1-doc (2.1.5-2+deb8u2) ...
```

```
Setting up ri (1:2.1.5+deb8u1) ...
Setting up ruby (1:2.1.5+deb8u1) ...
Processing triggers for libc-bin (2.19-18+deb8u1) ...
```

Ezután ellenőrizzük, hogy elérhető-e a `ruby` értelmező és a `gem` Ruby csomagkezelő, és nézzük meg az előre telepített Ruby API-k listáját.

```
root@debian:/home/kovacsg# ruby --version
ruby 2.1.5p273 (2014-11-13) [x86_64-linux-gnu]
root@debian:/home/kovacsg# gem --version
2.2.2

root@debian:/home/kovacsg# gem list

*** LOCAL GEMS ***

bigdecimal (1.2.4)
io-console (0.4.2)
json (1.8.1)
minitest (4.7.5)
psych (2.0.5)
rake (10.1.0)
rdoc (4.1.0)
test-unit (2.1.5.0)
```

A következő lépés a Ruby dokumentációgeneráló függvénykönyvtárának telepítése, amit a `gem` Ruby csomagkezelővel teszünk meg. A lépést kihagyva a Rails keretrendszer dokumentációjának telepítése sikertelen lenne, ami mindazonáltal nem nagy tragédia lévén annak telepítését amúgy is előszeretettel kikapcsoljuk a `gem --no-rdoc --no-ri` kapcsolóival. A dokumentumban ezeket nem használjuk, teljes telepítést végzünk.

```
root@debian:/home/kovacsg#gem install rdoc
Fetching: rdoc-4.2.0.gem (100%)
Depending on your version of ruby, you may need to install ruby rdoc/ri data
    :

<= 1.8.6 : unsupported
 = 1.8.7 : gem install rdoc-data; rdoc-data --install
 = 1.9.1 : gem install rdoc-data; rdoc-data --install
>= 1.9.2 : nothing to do! Yay!
Successfully installed rdoc-4.2.0
Parsing documentation for rdoc-4.2.0
Installing ri documentation for rdoc-4.2.0
Done installing documentation for rdoc after 15 seconds
1 gem installed
```

A C függvénykönyvtárakhoz való illesztéshez szükségünk lesz a Ruby header fájlokra is, ezért telepítjük azok Linux csomagját:

```
root@debian:/home/kovacsg# apt-get install ruby-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  ruby2.1-dev
The following NEW packages will be installed:
  ruby-dev ruby2.1-dev
```

```
0 upgraded, 2 newly installed, 0 to remove and 6 not upgraded.
Need to get 8,298 B/1,110 kB of archives.
After this operation, 4,428 kB of additional disk space will be used.
Do you want to continue? [Y/n]
Get:1 http://ftp.hu.debian.org/debian/ stable/main ruby-dev all 1:2.1.5+
    deb8u1 [8,298 B]
Fetched 8,298 B in 0s (102 kB/s)
Selecting previously unselected package ruby2.1-dev:amd64.
(Reading database ... 165671 files and directories currently installed.)
Preparing to unpack .../ruby2.1-dev_2.1.5-2+deb8u2_amd64.deb ...
Unpacking ruby2.1-dev:amd64 (2.1.5-2+deb8u2) ...
Selecting previously unselected package ruby-dev.
Preparing to unpack .../ruby-dev_1%3a2.1.5+deb8u1_all.deb ...
Unpacking ruby-dev (1:2.1.5+deb8u1) ...
Setting up ruby2.1-dev:amd64 (2.1.5-2+deb8u2) ...
Setting up ruby-dev (1:2.1.5+deb8u1) ...
```

Ezután telepítjük a Rails keretrendszert. A folyamatot a már említett `--no-rdoc --no-ri` kapcsolókkal felgyorsíthatjuk

```
root@debian:/home/kovacsg# gem install rails
Fetching: thread_safe-0.3.5.gem (100%)
Successfully installed thread_safe-0.3.5
Fetching: minitest-5.8.1.gem (100%)
Successfully installed minitest-5.8.1
Fetching: tzinfo-1.2.2.gem (100%)
Successfully installed tzinfo-1.2.2
Fetching: i18n-0.7.0.gem (100%)
Successfully installed i18n-0.7.0
Fetching: activesupport-4.2.4.gem (100%)
Successfully installed activesupport-4.2.4
Fetching: rails-deprecated_sanitizer-1.0.3.gem (100%)
Successfully installed rails-deprecated_sanitizer-1.0.3
Fetching: mini_portile-0.7.0.rc4.gem (100%)
Successfully installed mini_portile-0.7.0.rc4
Fetching: nokogiri-1.6.7.rc3.gem (100%)
Building native extensions.  This could take a while...
Successfully installed nokogiri-1.6.7.rc3
Fetching: rails-dom-testing-1.0.7.gem (100%)
Successfully installed rails-dom-testing-1.0.7
Fetching: crass-1.0.2.gem (100%)
Successfully installed crass-1.0.2
Fetching: loofah-2.1.0.rc1.gem (100%)
Successfully installed loofah-2.1.0.rc1
Fetching: rails-html-sanitizer-1.0.2.gem (100%)
Successfully installed rails-html-sanitizer-1.0.2
Fetching: erubis-2.7.0.gem (100%)
Successfully installed erubis-2.7.0
Fetching: builder-3.2.2.gem (100%)
Successfully installed builder-3.2.2
Fetching: actionview-4.2.4.gem (100%)
Successfully installed actionview-4.2.4
Fetching: rack-1.6.4.gem (100%)
Successfully installed rack-1.6.4
Fetching: rack-test-0.6.3.gem (100%)
Successfully installed rack-test-0.6.3
Fetching: actionpack-4.2.4.gem (100%)
Successfully installed actionpack-4.2.4
Fetching: activemodel-4.2.4.gem (100%)
Successfully installed activemodel-4.2.4
Fetching: arel-6.0.3.gem (100%)
```

```
Successfully installed arel-6.0.3
Fetching: activerecord-4.2.4.gem (100%)
Successfully installed activerecord-4.2.4
Fetching: mime-types-2.6.2.gem (100%)
Successfully installed mime-types-2.6.2
Fetching: mail-2.6.3.gem (100%)
Successfully installed mail-2.6.3
Fetching: globalid-0.3.6.gem (100%)
Successfully installed globalid-0.3.6
Fetching: activejob-4.2.4.gem (100%)
Successfully installed activejob-4.2.4
Fetching: actionmailer-4.2.4.gem (100%)
Successfully installed actionmailer-4.2.4
Fetching: thor-0.19.1.gem (100%)
Successfully installed thor-0.19.1
Fetching: railties-4.2.4.gem (100%)
Successfully installed railties-4.2.4
Fetching: bundler-1.10.6.gem (100%)
Successfully installed bundler-1.10.6
Fetching: sprockets-3.3.5.gem (100%)
Successfully installed sprockets-3.3.5
Fetching: sprockets-rails-3.0.0.beta2.gem (100%)
Successfully installed sprockets-rails-3.0.0.beta2
Fetching: rails-4.2.4.gem (100%)
Successfully installed rails-4.2.4
Parsing documentation for actionmailer-4.2.4
Installing ri documentation for actionmailer-4.2.4
Parsing documentation for actionpack-4.2.4
Installing ri documentation for actionpack-4.2.4
Parsing documentation for actionview-4.2.4
Installing ri documentation for actionview-4.2.4
Parsing documentation for activejob-4.2.4
Installing ri documentation for activejob-4.2.4
Parsing documentation for activemodel-4.2.4
Installing ri documentation for activemodel-4.2.4
Parsing documentation for activerecord-4.2.4
Installing ri documentation for activerecord-4.2.4
Parsing documentation for arel-6.0.3
Installing ri documentation for arel-6.0.3
Parsing documentation for builder-3.2.2
Installing ri documentation for builder-3.2.2
Parsing documentation for bundler-1.10.6
Installing ri documentation for bundler-1.10.6
Parsing documentation for crass-1.0.2
Installing ri documentation for crass-1.0.2
Parsing documentation for erubis-2.7.0
Installing ri documentation for erubis-2.7.0
Parsing documentation for globalid-0.3.6
Installing ri documentation for globalid-0.3.6
Parsing documentation for loofah-2.1.0.rc1
Installing ri documentation for loofah-2.1.0.rc1
Parsing documentation for mail-2.6.3
Installing ri documentation for mail-2.6.3
Parsing documentation for mime-types-2.6.2
Installing ri documentation for mime-types-2.6.2
Parsing documentation for nokogiri-1.6.7.rc3
Installing ri documentation for nokogiri-1.6.7.rc3
Parsing documentation for rack-1.6.4
Installing ri documentation for rack-1.6.4
Parsing documentation for rack-test-0.6.3
Installing ri documentation for rack-test-0.6.3
Parsing documentation for rails-4.2.4
```

```
Installing ri documentation for rails-4.2.4
Parsing documentation for rails-dom-testing-1.0.7
Installing ri documentation for rails-dom-testing-1.0.7
Parsing documentation for rails-html-sanitizer-1.0.2
Installing ri documentation for rails-html-sanitizer-1.0.2
Parsing documentation for railties-4.2.4
Installing ri documentation for railties-4.2.4
Parsing documentation for sprockets-3.3.5
Installing ri documentation for sprockets-3.3.5
Parsing documentation for sprockets-rails-3.0.0.beta2
Installing ri documentation for sprockets-rails-3.0.0.beta2
Parsing documentation for thor-0.19.1
Installing ri documentation for thor-0.19.1
Done installing documentation for actionmailer, actionpack, actionview,
    activejob, activemodel, activerecord, arel, builder, bundler, crass,
    erubis, globalid, loofah, mail, mime-types, nokogiri, rack, rack-test,
    rails, rails-dom-testing, rails-html-sanitizer, railties, sprockets,
    sprockets-rails, thor after 653 seconds
25 gems installed
```

A Rails rendszerünk konfigurációja a következő lesz a félév során. Kétféle webszervert használunk, a fejlesztéshez a beágyazott Webricket, míg az éles rendszerhez az Apache2-t. Adatbáziskezelőből szintén kétfélét nézünk meg, a beágyazott SQLite3-at és a kliens-szerver alapú MySQL-t. A következőkben ezek illesztéséhez szükséges adapterek fordítását lehetővé tevő C++ és Ruby függvénykönyvtárakat telepítjük.

Railshez egy opcionális, ám az éles rendszerek szempontjából annál hasznosabb komponens az Apache webszerverrel való integrációt lehetővé tevő Passenger plugin.

```
root@debian:/home/kovacsg# gem install passenger
Fetching: passenger-5.0.20.gem (100%)
Building native extensions.  This could take a while...
Successfully installed passenger-5.0.20
Parsing documentation for passenger-5.0.20
Installing ri documentation for passenger-5.0.20
Done installing documentation for passenger after 210 seconds
1 gem installed
```

Ezután telepíthetjük először az SQLite, majd a MySQL adapterének Ruby API-ját. Ez két lépésből áll, először a operációs rendszertől függő fájlok fordításához szükséges forrásokat (C és C++ headereket) telepítjük.

```
root@debian:/home/kovacsg# apt-get install libsqlite3-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  sqlite3-doc
The following NEW packages will be installed:
  libsqlite3-dev
0 upgraded, 1 newly installed, 0 to remove and 6 not upgraded.
Need to get 0 B/538 kB of archives.
After this operation, 1,542 kB of additional disk space will be used.
Selecting previously unselected package libsqlite3-dev:amd64.
(Reading database ... 165701 files and directories currently installed.)
```

```
Preparing to unpack .../libsqlite3-dev_3.8.7.1-1+deb8u1_amd64.deb ...
Unpacking libsqlite3-dev:amd64 (3.8.7.1-1+deb8u1) ...
Setting up libsqlite3-dev:amd64 (3.8.7.1-1+deb8u1) ...
root@debian:/home/kovacsg# apt-get install libmysqlclient-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  libmysqlclient-dev
0 upgraded, 1 newly installed, 0 to remove and 6 not upgraded.
Need to get 0 B/948 kB of archives.
After this operation, 5,714 kB of additional disk space will be used.
Selecting previously unselected package libmysqlclient-dev.
(Reading database ... 165643 files and directories currently installed.)
Preparing to unpack .../libmysqlclient-dev_5.5.44-0+deb8u1_amd64.deb ...
Unpacking libmysqlclient-dev (5.5.44-0+deb8u1) ...
Processing triggers for man-db (2.7.0.2-5) ...
Setting up libmysqlclient-dev (5.5.44-0+deb8u1) ...
```

Ezután telepíthetjük magukat az adatbáziskezelő-adaptereket, a `sqlite3`-t és a `mysql2`-t. A PostgreSQL adapterét `pg`-nek hívják, a gyakorlaton azt nem fogjuk használni.

```
root@debian:/home/kovacsg# gem install sqlite3
Fetching: sqlite3-1.3.10.gem (100%)
Building native extensions.  This could take a while...
Successfully installed sqlite3-1.3.10
Parsing documentation for sqlite3-1.3.10
Installing ri documentation for sqlite3-1.3.10
Done installing documentation for sqlite3 after 1 seconds
1 gem installed
root@debian:/home/kovacsg# gem install mysql2
Fetching: mysql2-0.4.1.gem (100%)
Building native extensions.  This could take a while...
Successfully installed mysql2-0.4.1
Parsing documentation for mysql2-0.4.1
Installing ri documentation for mysql2-0.4.1
Done installing documentation for mysql2 after 0 seconds
1 gem installed
```

Az alap Rails rendszerünk összeállt, menetközben szükségünk lesz további gemek telepítésére. Ilyen az `execjs` JavaScriptek Ruby-ba ágyzását teszi lehetővé, a `therubyracer` pedig egy szerver oldali JavaScript interpreter, vagy a `turbolinks` API nélkül a Rails már nem hajlandó elindulni, noha az opcionális.

2. Rails alkalmazás létrehozása

A Rails használatához a `rails` szkriptet használjuk, amely Debian/Ubuntu Linux és 2.1-es `ruby` esetén a `/usr/local/bin/` könyvárba kerül.

A `-d` kapcsoló különös jelentőséggel bír számunkra, ezzel adhatjuk meg a használni kívánt adatbáziskezelő típusát, ami lehet `mysql`, `oracle`, `postgresql`, `sqlite3`, `frontbase` vagy `ibm_db`. A gyakorlatok keretében ezek közül a

`sqlite`-ot mint alapértelmezett adatbáziskezelőt és a `mysql`-t fogjuk használni. A `-B` kapcsoló mellőzi a Rails alkalmazásunk Ruby függőségeinek feloldását, amelyet azonban később pótolnunk kell. Javascript API-nak a `jquery` tökéletesen meg fog felelni számunkra, így azt nem módosítjuk.

```
kovacsg@debian:~# rails
Usage:
  rails new APP_PATH [options]

Options:
  -r, [--ruby=PATH]                                # Path to the Ruby
      binary of your choice
                                                   # Default: /usr/bin
                                                     /ruby
  -m, [--template=TEMPLATE]                        # Path to some
      application template (can be a filesystem path or URL)
      [--skip-gemfile], [--no-skip-gemfile]        # Don't create a
          Gemfile
  -B, [--skip-bundle], [--no-skip-bundle]          # Don't run bundle
      install
  -G, [--skip-git], [--no-skip-git]                # Skip .gitignore
      file
      [--skip-keeps], [--no-skip-keeps]            # Skip source
          control .keep files
  -O, [--skip-active-record], [--no-skip-active-record]  # Skip Active
      Record files
  -S, [--skip-sprockets], [--no-skip-sprockets]    # Skip Sprockets
      files
      [--skip-spring], [--no-skip-spring]          # Don't install
          Spring application preloader
  -d, [--database=DATABASE]                        # Preconfigure for
      selected database (options: mysql/oracle/postgresql/sqlite3/frontbase/
      ibm_db/sqlserver/jdbcmysql/jdbcsqlite3/jdbcpostgresql/jdbc)
                                                   # Default: sqlite3
  -j, [--javascript=JAVASCRIPT]                    # Preconfigure for
      selected JavaScript library
                                                   # Default: jquery
  -J, [--skip-javascript], [--no-skip-javascript]  # Skip JavaScript
      files
      [--dev], [--no-dev]                          # Setup the
          application with Gemfile pointing to your Rails checkout
      [--edge], [--no-edge]                        # Setup the
          application with Gemfile pointing to Rails repository
      [--skip-turbolinks], [--no-skip-turbolinks]  # Skip turbolinks
          gem
  -T, [--skip-test-unit], [--no-skip-test-unit]    # Skip Test::Unit
      files
      [--rc=RC]                                    # Path to file
          containing extra configuration options for rails command
      [--no-rc], [--no-no-rc]                      # Skip loading of
          extra configuration options from .railsrc file

Runtime options:
  -f, [--force]                    # Overwrite files that already exist
  -p, [--pretend], [--no-pretend]  # Run but do not make any changes
  -q, [--quiet], [--no-quiet]      # Suppress status output
  -s, [--skip], [--no-skip]        # Skip files that already exist

Rails options:
  -h, [--help], [--no-help]        # Show this help message and quit
```

```
  -v, [--version], [--no-version]  # Show Rails version number and quit

Description:
    The 'rails new' command creates a new Rails application with a default
    directory structure and configuration at the path you specify.

    You can specify extra command-line arguments to be used every time
    'rails new' runs in the .railsrc configuration file in your home
        directory.

    Note that the arguments specified in the .railsrc file don't affect the
    defaults values shown above in this help message.

Example:
    rails new ~/Code/Ruby/weblog

    This generates a skeletal Rails installation in ~/Code/Ruby/weblog.
    See the README in the newly created application to get going.
```

A Rails keretrendszerben a `rails` parancs az az univerzális eszköz mellyel többek között új komponenseket hozhatunk létre és elindíthatjuk a beépített webszervert. Egy új Rails alkalmazást a `rails` parancsnak `new` opciót megadva hozhatunk létre, a második argumentum az alkalmazás neve. A parancsot kiadva a konzolon láthatjuk az automatikusan generált fájlokat. Az egyes könyvtárak értelmezéséről előadáson esett szó, e gyakorlat keretében megnézzük az egyes fájlok szerepét és tartalmát.

A parancs végén automatikusan lefutna a Rails keretrendszer egy másik parancsa, a `bundle`, amely az `install` opció határása összeszedi a gemek közül azokat, amelyekre az alkalmazásunknak szüksége lesz, azonban ezt a `-B` kapcsolóval letiltjuk, mert testre akarjuk szabni a keretrendszerünket.

```
root@debian:/home/kovacsg# rails new gyakorlat -B
      create
      create  README.rdoc
      create  Rakefile
      create  config.ru
      create  .gitignore
      create  Gemfile
      create  app
      create  app/assets/javascripts/application.js
      create  app/assets/stylesheets/application.css
      create  app/controllers/application_controller.rb
      create  app/helpers/application_helper.rb
      create  app/views/layouts/application.html.erb
      create  app/assets/images/.keep
      create  app/mailers/.keep
      create  app/models/.keep
      create  app/controllers/concerns/.keep
      create  app/models/concerns/.keep
      create  bin
      create  bin/bundle
      create  bin/rails
      create  bin/rake
      create  bin/setup
      create  config
      create  config/routes.rb
```

```
      create  config/application.rb
      create  config/environment.rb
      create  config/secrets.yml
      create  config/environments
      create  config/environments/development.rb
      create  config/environments/production.rb
      create  config/environments/test.rb
      create  config/initializers
      create  config/initializers/assets.rb
      create  config/initializers/backtrace_silencers.rb
      create  config/initializers/cookies_serializer.rb
      create  config/initializers/filter_parameter_logging.rb
      create  config/initializers/inflections.rb
      create  config/initializers/mime_types.rb
      create  config/initializers/session_store.rb
      create  config/initializers/wrap_parameters.rb
      create  config/locales
      create  config/locales/en.yml
      create  config/boot.rb
      create  config/database.yml
      create  db
      create  db/seeds.rb
      create  lib
      create  lib/tasks
      create  lib/tasks/.keep
      create  lib/assets
      create  lib/assets/.keep
      create  log
      create  log/.keep
      create  public
      create  public/404.html
      create  public/422.html
      create  public/500.html
      create  public/favicon.ico
      create  public/robots.txt
      create  test/fixtures
      create  test/fixtures/.keep
      create  test/controllers
      create  test/controllers/.keep
      create  test/mailers
      create  test/mailers/.keep
      create  test/models
      create  test/models/.keep
      create  test/helpers
      create  test/helpers/.keep
      create  test/integration
      create  test/integration/.keep
      create  test/test_helper.rb
      create  tmp/cache
      create  tmp/cache/assets
      create  vendor/assets/javascripts
      create  vendor/assets/javascripts/.keep
      create  vendor/assets/stylesheets
      create  vendor/assets/stylesheets/.keep
```

Az `app` könyvtár fogja tartalmazni az általunk készített Ruby és beágyazott Ruby kódot tartalmazó HTML forrásokat, amelyeket a MVC minta alapján struktúrál a Rails, amelyeket a `test` könyvtárban elhelyezett teszt osztályokkal ellenőrzünk. A `bin` könyvtár elérhetővé teszi számunkra a `rails`,

a `rake` és a `bundle` parancsokat, amelyeket a Rails alkalmazásunk menedzsmentjét fogjuk megvalósítani. A `config` könyvtár a Rails alkalmazásunk konfigurációs beállításait tartalmazza. A `db` könyvtár az aktuális adatbázis sémát, az összes eddig adatbázis séma migrációt és sqlite adatbáziskezelő esetén szerializált formában magát az adatbázis tartalmazza. A `doc` a dokumentációk gyűjtőhelye, a `lib` és `vendor` könytvárak mások által készített Ruby, illetve Rails függvénykönyvtárakat tartalmazhatnak. A `public` könyvtár a beépített webszerver területe, az összes ott bekövetkezett esemény a `log` könyvtárban található az aktuális Rails környezetnek megfelelő állományban kerülnek naplózásra, a webszerver `tmp` könyvárban helyezheti el az átmeneti fájljait, mint például session azonosítókat, sütiket.

A `Gemfile` és `Rakefile` a Rails alkalmazásunk által használt Ruby függvénykönytárakat specifikálja, amelyek vagy a telepített Ruby és Rails környezeten vagy a már említett `lib` és `vendor` könytvárakban érhetők el.

Az első dolgunk a `Gemfile` módosítása. Először kikommentezzük az első sort, hogy a Rails a Ruby függvénykönyvtárakat a helyi, frissen telepített helyen keresse, és ne a megadott távoli szerveren. Ez azért szükséges, hogy a rendszer egy rögzített verziójú keretrendszer mellett tudhassuk fejleszeni, és az ne frissüljön a `Gemfile` minden módosításakor. Egy esetleges frissítés fejlesztés közben elronthatja a teljes addigi munkánkat! Praktikusan a `rails`, az adatbáziskezelő adaptere és a `jquery-rails`-en kívül a többi, a fájlban szereplő függőségre nincs szükségünk. A beágyazott webszerverünk futtatásához szükségünk lesz szerver oldali JavaScript értelmezőre, ezért a fájlba felvesszük a következő gem függőségeket. A `Gemfile` minden további módosítása után futtatnunk kell majd a `bundle install` vagy `bundle update` parancsot.

```
gem 'therubyracer', platforms: :ruby
gem 'execjs'
```

A konzolon kiadott `bundle install` paranccsal telepíthetjük az összes az esetlegesen hiányzó vagy nem megfelelő verziójú Ruby függvénykönyvtárat, a `bundle update` paranccsal pedig az éppen használni kívánt verziót tudunk betölteni az egyes gemekből. A `bundle` parancshoz a `--path` kapcsolót és egy fájlrendszeri útvonalat hozzáfűzve elérhetjük, hogy garantáltan mindig a saját Ruby környezetünkkel dolgozhassunk, és ne ütközzünk az időközben bekövetkető rendszerfrissítések okozta inkompatibilitás áldozatául. Ha a Ruby csomagokat minden felhasználó számára elérhetővé akarjuk tenni, meg kell adnunk az adminisztrátor jelszavát.

```
kovacsg@debian:~# cd gyakorlat
kovacsg@debian:~/gyakorlat# bundle install
Fetching gem metadata from https://rubygems.org/.....
```

```
Fetching version metadata from https://rubygems.org/...
Fetching dependency metadata from https://rubygems.org/..
Resolving dependencies.................


Your user account isn't allowed to install to the system Rubygems.
You can cancel this installation and run:

    bundle install --path vendor/bundle

to install the gems into ./vendor/bundle/, or you can enter your password
and install the bundled gems to Rubygems using sudo.

Password:
Installing rake 10.4.2
Using i18n 0.7.0
Installing json 1.8.3 with native extensions
Using minitest 5.8.1
Using thread_safe 0.3.5
Using tzinfo 1.2.2
Using activesupport 4.2.4
Using builder 3.2.2
Using erubis 2.7.0
Installing mini_portile 0.6.2
Installing nokogiri 1.6.6.2 with native extensions
Using rails-deprecated_sanitizer 1.0.3
Using rails-dom-testing 1.0.7
Installing loofah 2.0.3
Using rails-html-sanitizer 1.0.2
Using actionview 4.2.4
Using rack 1.6.4
Using rack-test 0.6.3
Using actionpack 4.2.4
Using globalid 0.3.6
Using activejob 4.2.4
Using mime-types 2.6.2
Using mail 2.6.3
Using actionmailer 4.2.4
Using activemodel 4.2.4
Using arel 6.0.3
Using activerecord 4.2.4
Installing debug_inspector 0.0.2 with native extensions
Installing binding_of_caller 0.7.2 with native extensions
Using bundler 1.10.6
Installing byebug 6.0.2 with native extensions
Installing coffee-script-source 1.9.1.1
Installing execjs 2.6.0
Installing coffee-script 2.4.1
Using thor 0.19.1
Using railties 4.2.4
Installing coffee-rails 4.1.0
Installing multi_json 1.11.2
Installing jbuilder 2.3.2
Installing jquery-rails 4.0.5
Installing libv8 3.16.14.11
Using sprockets 3.3.5
Installing sprockets-rails 2.3.3
Using rails 4.2.4
Using rdoc 4.2.0
Installing ref 2.0.0
Installing sass 3.4.18
Installing tilt 2.0.1
```

```
Installing sass-rails 5.0.4
Installing sdoc 0.4.1
Installing spring 1.4.0
Using sqlite3 1.3.10
Installing therubyracer 0.12.2 with native extensions
Installing turbolinks 2.5.3
Installing uglifier 2.7.2
Installing web-console 2.2.1
Bundle complete! 14 Gemfile dependencies, 56 gems now installed.
Use `bundle show [gemname]` to see where a bundled gem is installed.
```

A Rails rendszerünk ezután az alábbi Ruby függvénykönyvtárakból áll:

```
root@debian:/home/kovacsg#gem list

*** LOCAL GEMS ***

actionmailer (4.2.4)
actionpack (4.2.4)
actionview (4.2.4)
activejob (4.2.4)
activemodel (4.2.4)
activerecord (4.2.4)
activesupport (4.2.4)
arel (6.0.3)
bigdecimal (1.2.4)
binding_of_caller (0.7.2)
builder (3.2.2)
bundler (1.10.6)
byebug (6.0.2)
coffee-rails (4.1.0)
coffee-script (2.4.1)
coffee-script-source (1.9.1.1)
crass (1.0.2)
debug_inspector (0.0.2)
erubis (2.7.0)
execjs (2.6.0)
globalid (0.3.6)
i18n (0.7.0)
io-console (0.4.2)
jbuilder (2.3.2)
jquery-rails (4.0.5)
json (1.8.3, 1.8.1)
libv8 (3.16.14.11 x86_64-linux)
loofah (2.1.0.rc1, 2.0.3)
mail (2.6.3)
mime-types (2.6.2)
mini_portile (0.7.0.rc4, 0.6.2)
minitest (5.8.1, 4.7.5)
multi_json (1.11.2)
mysql2 (0.4.1)
nokogiri (1.6.7.rc3, 1.6.6.2)
passenger (5.0.20)
psych (2.0.5)
rack (1.6.4)
rack-test (0.6.3)
rails (4.2.4)
rails-deprecated_sanitizer (1.0.3)
rails-dom-testing (1.0.7)
rails-html-sanitizer (1.0.2)
railties (4.2.4)
rake (10.4.2, 10.1.0)
```

```
rdoc (4.2.0, 4.1.0)
ref (2.0.0)
sass (3.4.18)
sass-rails (5.0.4)
sdoc (0.4.1)
spring (1.4.0)
sprockets (3.3.5)
sprockets-rails (3.0.0.beta2, 2.3.3)
sqlite3 (1.3.10)
test-unit (2.1.5.0)
therubyracer (0.12.2)
thor (0.19.1)
thread_safe (0.3.5)
tilt (2.0.1)
turbolinks (2.5.3)
tzinfo (1.2.2)
uglifier (2.7.2)
web-console (2.2.1)
```

A `rails` parancs gyakran használt argumentuma a `console` vagy röviden `c`, amivel az első gyakorlatról megismert interaktív Ruby értelmezőt indíthatunk, melyben a Rails alkalmazásunk környezeti beállításai inicializálásra kerültek.

```
kovacsg@debian:~/gyakorlat#rails console
Loading development environment (Rails 4.2.4)
irb(main):001:0>
```

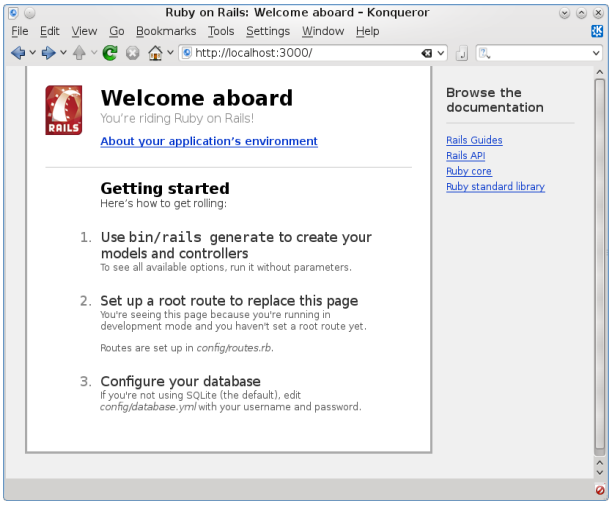
3. Web- és adatbázisszerver kapcsolat beállítása

A Rails keretrendszer webszerverek számára állít elő dinamikus tartalmat adatbázisbeli adatok alapján. A gyakorlat során kétféle webszervert és adatbáziskezelőt vizsgáltunk meg.

A beépített webszervert a `rails server` vagy röviden `s` opciójával indíthatjuk el a Rails alkalmazásunk tetszőleges könyvtárából. Ha a szervert háttéralkalmazásként kívánjuk futtatni, akkor a `-d` kapcsolót is hozzá kell fűznünk a parancssorhoz. Ezután a szerver elérhető a `http://localhost:3000` webcímen, lásd 1. ábra.

```
kovacsg@debian:~/gyakorlat# rails server
=> Booting WEBrick
=> Rails 4.2.4 application starting in development on http://localhost:3000
=> Run `rails server -h` for more startup options
=> Ctrl-C to shutdown server
[2015-10-01 09:35:49] INFO  WEBrick 1.3.1
[2015-10-01 09:35:49] INFO  ruby 2.1.5 (2014-11-13) [x86_64-linux-gnu]
[2015-10-01 09:35:49] INFO  WEBrick::HTTPServer#start: pid=7822 port=3000
```

A Rails a már említett Passenger plugin segítségével illeszthető Apache2 webszerverhez. Ehhez az alábbi kódrészletben mutatott parancsot kell kiadnunk. A parancs lefordítja, majd telepíti az Apache webszerver Rails (és



1. ábra. A beágyazott webszerver elérése

egyéb további) modulját, a konzolra kiírja az Apache2 konfigurációs állományába beírandó Rails specifikus három sort, és a Rails alkalmazásunkhoz való hozzáféréshez szükséges VirtualHost beállításokat. A Passenger alapértelmezés szerint a Rails éles környezetével működik együtt, ha ezt módosítani szerenténk, akkor a `RailsEnv development` sort kell elhelyeznünk az Apache konfigurációs állományába a fejlesztői környezet eléréséhez.

```
passenger-install-apache2-module
```

A Rails kényelmesebb hozzáférése végett módosíthatjuk a helyi gépen doménnévtáblát (Windowson `C:\windows\system32\drivers\etc\hosts`, Linuxon `/etc/hosts`) a következő bejegyzész hozzáadva, amely minden, a `gyakorlat.com`, illetve `www.gyakorlat.com` címre küldött kérést a helyi gépen kezeltet le.

```
127.0.0.1       gyakorlat.com www.gyakorlat.com
```

Az Apache2 modul elérhetővé tételére az Apache2 konfigurációs könyvtárában létre kell hoznunk a modul beállításait. Szerencsére a telepítő parancs a helyes beállításokat kiírja a konzolra. A fájl a `/etc/apache2/mods-enabled/rails.load` néven hozzuk létre, a tartalmaz az alábbi konfigurációrészlethez hasolnó.

```
LoadModule passenger_module /var/lib/gems/2.1.0/gems/passenger-5.0.20/
    buildout/apache2/mod_passenger.so
  <IfModule mod_passenger.c>
    PassengerRoot /var/lib/gems/2.1.0/gems/passenger-5.0.20
    PassengerDefaultRuby /usr/bin/ruby2.1
  </IfModule>
```

Az Apache2 virtuális hoszt beállításait szintén konfigurációs állományok között kell elhelyeznünk például `/etc/apache2/sites-available/gyakorlat.conf` néven, majd az `a2ensite` paranccsal konzolon engedélyezzük a gyakorlat oldalt[4]. Ennek tartalma a telepítő parancs alapján a következő lehet:

```
RackEnv development

#<VirtualHost *:80>
<VirtualHost www.gyakorlat.com:80>

  ServerName www.gyakorlat.com
  ServerAdmin admin@gyakorlat.com
  DocumentRoot /home/kovacsg/gyakorlat/public
  ServerSignature On

  CustomLog /var/log/apache2/gyakorlat_access.log combined
  ErrorLog /var/log/apache2/gyakorlat_error.log
  LogLevel info

  <Directory />
    Options FollowSymLinks
    AllowOverride None
  </Directory>
  <Directory /home/kovacsg/gyakorlat/public/>
    Require all granted
    Options Indexes FollowSymLinks
    AllowOverride None
    Order allow,deny
    Allow from all
    Options -MultiViews
  </Directory>

</VirtualHost>
```

Mivel módosítottuk az Apache2 beállításait, újra kell töltenünk annak konfigurációs állományait.

```
root@debian:~# /etc/init.d/apache2 reload
[ ok ] Reloading web server config: apache2.
```

Ezután egy böngészőbe beírva a `www.gyakorlat.com` címet a Rails alkalmazásunk fog megjelenni! A Rails alkalmazás módosítása után az Apache-beli frissítése a `tmp/restart.txt` fájl időpecsétjének módosításával lehetséges (mentés, létrehozás, `touch` stb.)

Mivel a Rails alkalmazás létrehozásakor nem adtunk meg, hogy melyik adatbáziskezelőt kívánjuk használni, ezért az alapértelmezett SQLite konfigurációs paraméterei jelennek meg a `database.yml`-ben mindhárom definiált környezethez, vagyis a fejlesztési (`development`), tesztelési (`test`) és éles (`production`). Az egyes környezetek adapter opciója határozza meg az adatbáziskezelő típusát és a használható paraméterkészletet. Az SQLite

[4] Alternatív megoldás: ln -s /etc/apache2/sites-available/gyakorlat.conf /etc/apache2/sites-enabled

az adatait fájlokban tárolja, amelyek a Rails alkalmazás `db` könyvtárában találhatók, és a környezet nevével azonosítottak e fájl szerint.

```
# SQLite version 3.x
#   gem install sqlite3
#
#   Ensure the SQLite 3 gem is defined in your Gemfile
#   gem 'sqlite3'
#
default: &default
  adapter: sqlite3
  pool: 5
  timeout: 5000

development:
  <<: *default
  database: db/development.sqlite3

# Warning: The database defined as "test" will be erased and
# re-generated from your development database when you run "rake".
# Do not set this db to the same as development or production.
test:
  <<: *default
  database: db/test.sqlite3

production:
  <<: *default
  database: db/production.sqlite3
```

MySQL esetén (`rails new gyakorlat -d mysql`) a `mysql2` adaptert használjuk[5]. A különbség a `Gemfile`-ban és a `database.yml`-ben jelentkezik. Fontos opció az alapértelmezett karakterkódolás megadása, ami, ha lehet hagyjunk változatlanul utf8 értéken. A `database` opció a séma nevét tartalmazza értékként. A felhasználónév és jelszó megadása kötelező. A kapcsolat lehet `socket` vagy `host` és `port` alapon megadott.

Távoli gépen található MySQL esetén szükséges a Rails felhasználó hozzáférési jogosultságainak beállítása, amelyet a MySQL `mysql` adatbázisának `user` táblájában kell megtennünk. A Rails szerverének IP címére engedélyeznünk kell a Rails adatbázis-konfigurációs állományában megadott felhasználónévvel és jelszóval azonosított felhasználó számára a select, update, insert, delete, create, alter, drop és index jogosultságokat.

Konzolos adatbáziskapcsolatot a `rails db` paranccsal tudunk előhozni.

```
kovacsg@debian:~/gyakorlat/db# rails db
SQLite version 3.7.13
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> .schema
sqlite>
```

[5]A Rails 4.2.4-es változatához a következő függőség kell: gem 'mysql2', ' > 0.3.18' Ne feledkezzünk el a bundle update parancs kiadásáról sem.

Az adatbáziskapcsolatnak a szerver elindítása előtt léteznie kell. Ez SQLite esetén nem probléma, hiszen a fejlesztői adatbázisfájl automatikusan létrejön, MySQL esetén viszont a sémákat explicite létre kell hoznunk. Ebben a `rake` parancs lesz segítségünkre, amelyet a Rails keretrendszerhez kapcsolódó műveletekre használunk. A műveletek listája alább látható.

```
kovacsg@debian:~/gyakorlat/bin$rake -T
rake about                              # List versions of all Rails
    frameworks and the environment
rake assets:clean[keep]                 # Remove old compiled assets
rake assets:clobber                     # Remove compiled assets
rake assets:environment                 # Load asset compile environment
rake assets:precompile                  # Compile all the assets named in
    config.assets.precompile
rake cache_digests:dependencies         # Lookup first-level dependencies
    for TEMPLATE (like messages/show or comments/_comment.html)
rake cache_digests:nested_dependencies  # Lookup nested dependencies for
    TEMPLATE (like messages/show or comments/_comment.html)
rake db:create                          # Creates the database from
    DATABASE_URL or config/database.yml for the current RAILS_ENV (use db:
    create:all to create all databases in the config)
rake db:drop                            # Drops the database from
    DATABASE_URL or config/database.yml for the current RAILS_ENV (use db:
    drop:all to drop all databases in the config)
rake db:fixtures:load                   # Load fixtures into the current
    environment's database
rake db:migrate                         # Migrate the database (options:
    VERSION=x, VERBOSE=false, SCOPE=blog)
rake db:migrate:status                  # Display status of migrations
rake db:rollback                        # Rolls the schema back to the
    previous version (specify steps w/ STEP=n)
rake db:schema:cache:clear              # Clear a db/schema_cache.dump file
rake db:schema:cache:dump               # Create a db/schema_cache.dump file
rake db:schema:dump                     # Create a db/schema.rb file that is
     portable against any DB supported by AR
rake db:schema:load                     # Load a schema.rb file into the
    database
rake db:seed                            # Load the seed data from db/seeds.
    rb
rake db:setup                           # Create the database, load the
    schema, and initialize with the seed data (use db:reset to also drop the
     database first)
rake db:structure:dump                  # Dump the database structure to db/
    structure.sql
rake db:structure:load                  # Recreate the databases from the
    structure.sql file
rake db:version                         # Retrieves the current schema
    version number
rake doc:app                            # Generate docs for the app -- also
    available doc:rails, doc:guides (options: TEMPLATE=/rdoc-template.rb,
    TITLE="Custom Title")
rake log:clear                          # Truncates all *.log files in log/
    to zero bytes (specify which logs with LOGS=test,development)
rake middleware                         # Prints out your Rack middleware
    stack
rake notes                              # Enumerate all annotations (use
    notes:optimize, :fixme, :todo for focus)
rake notes:custom                       # Enumerate a custom annotation,
    specify with ANNOTATION=CUSTOM
rake rails:template                     # Applies the template supplied by
```

```
    LOCATION=(/path/to/template) or URL
rake rails:update                    # Update configs and some other
    initially generated files (or use just update:configs or update:bin)
rake routes                          # Print out all defined routes in
    match order, with names
rake secret                          # Generate a cryptographically
    secure secret key (this is typically used to generate a secret for
    cookie sessions)
rake stats                           # Report code statistics (KLOCs, etc
    ) from the application or engine
rake test                            # Runs all tests in test folder
rake test:all                        # Run tests quickly by merging all
    types and not resetting db
rake test:all:db                     # Run tests quickly, but also reset
    db
rake test:db                         # Run tests quickly, but also reset
    db
rake time:zones:all                  # Displays all time zones, also
    available: time:zones:us, time:zones:local -- filter with OFFSET
    parameter, e.g., OFFSET=-6
rake tmp:clear                       # Clear session, cache, and socket
    files from tmp/ (narrow w/ tmp:sessions:clear, tmp:cache:clear, tmp:
    sockets:clear)
rake tmp:create                      # Creates tmp directories for
    sessions, cache, sockets, and pids
```

Az adatbázis létrehozása következő paranccsal történhet meg, amely a development és a test környezetekhez hozza létre egy üres sémát.

```
kovacsg@debian:~/gyakorlat$ rake db:create
```

4. Bevezetés a Rails használatába

Az adatbáziskapcsolat-leíró mellett a másik fontos konfigurációs fájlunk a `routes.rb`. Ez azt adja meg, hogy milyen struktúrájú legyen az URL, amivel elérjük a Rails alkalmazásunk egyes funkcióit. A Rails kontrollerek létrehozásakor automatikusan írja ezt a fájl, de megadhatunk egy általános mintát is a HTTP kérések URI-ainak útvonal részére, az alábbi kódrészlet a legláltalánosabb beállítást tartalmazza. A webszerver IP címe után a Controller osztály neve (`:controller`), majd a Controller osztály egy metódusa (`:action`), majd egy adatbázis azonosító (`:id`), és végül formázási útmutató következik, például `.html` vagy `.xml`. Az utolsó három megadása opcionális.

```
Gyakorlat::Application.routes.draw do
  match ':controller(/:action(/:id(.:format)))'
end
```

Nézzük meg, hogy miként tudunk dinamikus tartalmat létrehozni Railsszel. A példák a [1] könyvből valók.

Hozzunk létre egy új controllert a `rails` parancs `generate` opciójával. A második argumentum (`controller`) azt mondja meg, hogy egy új controllert

hozunk létre, a harmadik a controller nevét. A negyedik és minden további paraméter a controllerben definiál akciókat. E parancs négy Ruby forrásfájlt és egy könyvárat hoz létre az akcióknak megfelelő weboldalak, view-k számára. A controller nevének megfelelő controller osztályt (`say_controller.rb`), helper osztályt, illetve ezek funkcionális és egységtesztjéhez használható osztályokat.

```
kovacsg@debian:~/gyakorlat$ rails generate
Usage: rails generate GENERATOR [args] [options]

General options:
  -h, [--help]     # Print generator's options and usage
  -p, [--pretend]  # Run but do not make any changes
  -f, [--force]    # Overwrite files that already exist
  -s, [--skip]     # Skip files that already exist
  -q, [--quiet]    # Suppress status output

Please choose a generator below.

Rails:
  assets
  controller
  generator
  helper
  integration_test
  jbuilder
  job
  mailer
  migration
  model
  resource
  scaffold
  scaffold_controller
  task

Coffee:
  coffee:assets

Js:
  js:assets

TestUnit:
  test_unit:generator
  test_unit:job
  test_unit:plugin
kovacsg@debian:~/gyakorlat/tmp$ rails generate controller say hello
      create  app/controllers/say_controller.rb
       route  get 'say/hello'
      invoke  erb
      create    app/views/say
      create    app/views/say/hello.html.erb
      invoke  test_unit
      create    test/controllers/say_controller_test.rb
      invoke  helper
      create    app/helpers/say_helper.rb
      invoke    test_unit
      invoke  assets
      invoke    coffee
      create      app/assets/javascripts/say.coffee
```

```
      invoke    scss
      create      app/assets/stylesheets/say.scss
```

Bármilyen tartalom megjelenítéséhez a `routes.rb` alapján a view könyvtárban kell elhelyeznünk az akciónak megfelelő néven egy beágyazott Ruby kódot tartalmazó HTML fájlt (`.rhtml` vagy `.html.erb`).

Az alkalmazás keretét az `app/views/layouts/application.html.erb` fájl definiálja, amely a HTML dokumentum törzs helyén egy beágyazott `yield` parancsot tartalmaz, amely átadja a vezérlést az akció HTML-ének, ami jelen esetben `hello` lesz.

Ez alapján nézzük meg a szokásos Hello, world alkalmazást ezúttal Railsben. Az `app/views/say` könyvtárban létrehozunk egy `hello.html.erb` nevű view-t, amely a `say` controller `hello` akciójához kötődik. Az eredményt a `http://localhost:3000/say/hello` linken ellenőrizhetjük.

```
<h1>Hello, world!</h1>
```

Ez dinamikussá tehetjük az aktuális idő kiírásával.

```
<%= Time.now %>
```

Mivel a nézetbe nem illik logikát rakni, csak a megjelenítendő értéket, ezért áttesszük az idő lekérdezését a kontrollerbe, annak is az akciónak megfelelő metódusába, a `hello`-ba

```
class SayController < ApplicationController
  def hello
    @time=Time.now
  end
end
```

A nézetben pedig csak hivatkozunk a Controller példányváltozóira.

```
<%= @time %>
```

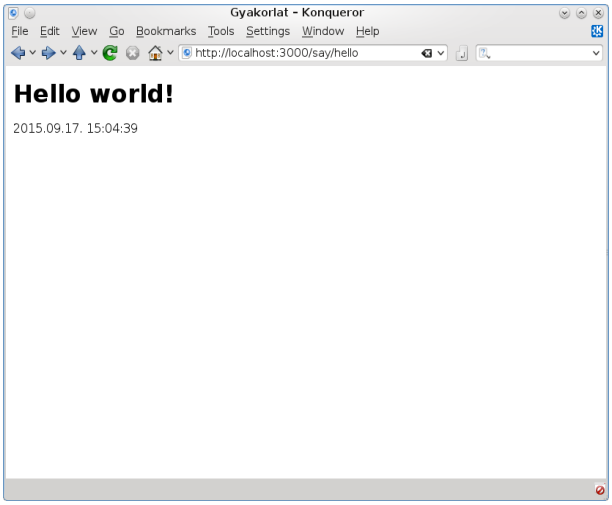
A nézet többnyelvűsítését a `/config/locales/` könyvárban lévő YML fájlokkal érhetjük el. Definiáljuk a hello fordítását és egy időformátumot.

```
en:
  hello: "Hello world!"

  time:
    formats:
      default: '%Y.%m.%d.'
      datetime: '%Y.%m.%d. %H:%M:%
```

A nézetünkben (`app/views/say/hello.html.erb`) pedig használjuk a fordítást végző `t`, és a lokalizációt végző `l` függvényeket.

```
<h1><%= t :hello %></h1>
<p><%= l @time, format: :datetime %></p>
```



2. ábra. A hello akció megjelenítve

A controllernek és akciónak megfelelő megjelenített oldalt a 2. ábra mutatja.

Az oldal forrását megnézve felismerjük benne a layout által nyújtott keretet és a View beágyazott kódját.

```
<!DOCTYPE html>
<html>
<head>
  <title>Gyakorlat</title>
</head>
<body>

<h1>Hello, world!</h1>
<p>2015-09-17 15:04:39</p>


</body>
</html>
```

A Rails MVC filozófiájának harmadik eleme a modell, amelyet szintén a `rails` parancs `generate` opciójával hozhatunk létre. A harmadik argumentum a modell oszlály neve, amely a konvenció alapján egy egyes számban megadó és a szavakat `_` szimbólummal összefűző string. Ennek többes számú változatával jön létre az az adatbázisban egy tábla. A parancs kiadása négy fájlt hoz létre: egy adatbázis migrációs Ruby szkriptet, egy `ActiveRecord::Base` leszármazottat a modell osztályok közé, egy egységteszteket tartalmazó osztályt és egy tesztadatokat tartalmazó YAML fájlt.

```
kovacsg@debian:~/gyakorlat# rails generate model User username:string
    password:string email:string birthdate:datetime
      invoke  active_record
      create    db/migrate/20150917131714_create_users.rb
```

```
      create    app/models/user.rb
      invoke    test_unit
      create      test/models/user_test.rb
      create      test/fixtures/users.yml
```

Az adatbázis-migrációs szkriptben az adatmodell változtatásait adjuk meg. Az előző modellgeneráló szkript egy olyan táblát hozna létre, amelyben egy azonosító és két időpecsét attribútum mellett egy `username`, egy `password`, egy `email` azonosítójú string típusú és egy `birthdate` nevű dátum típusú attribútum szerepelne. Jelszó attribútumot elfelejtettünk megadni, de ez nem fog problémát okozni, mert a sémánk verziókezelt.

```
class CreateUsers < ActiveRecord::Migration
  def change
    create_table :users do |t|
      t.string :username
      t.string :password
      t.string :email
      t.datetime :birthdate

      t.timestamps null: false
    end
  end
end
```

Ezután elvégezhetjük a tábla struktúrájának módosítását, amiben a `rake` parancs nyújt segítséget. A `rake db:create` létrehozza a táblákat, ha még nem tettük volna meg, a `rake db:migrate` módotítja a séma struktúráját.

```
ovacsg@debian:~/gyakorlat$ rake db:create
kovacsg@debian:~/gyakorlat/db/migrate$ rake db:migrate
(in /home/kovacsg/gyakorlat)
== 20150917131714 CreateUsers: migrating
   ===================================
-- create_table(:users)
   -> 0.0801s
== 20150917131714 CreateUsers: migrated (0.0803s)
   ==========================
```

Az adatbáziskezelővel a `gyakorlat_development` adatbázist kiválasztva ezután ellenőrizhetjük, hogy a táblánk valóban létrejött. Alább ennek ellenőrzése látható a két adatbáziskezelő konzolján.

```
root@debian:~# mysql -u root
mysql> use gyakorlat_development;
Database changed
mysql> show tables;
+-----------------------------------+
| Tables_in_gyakorlat_development |
+-----------------------------------+
| schema_migrations                 |
| users                             |
+-----------------------------------+
2 rows in set (0.00 sec)

mysql> describe users;
+-------------+--------------+------+-----+---------+----------------+
```

```
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| id         | int(11)      | NO   | PRI | NULL    | auto_increment |
| username   | varchar(255) | YES  |     | NULL    |                |
| password   | varchar(255) | YES  |     | NULL    |                |
| email      | varchar(255) | YES  |     | NULL    |                |
| birthdate  | datetime     | YES  |     | NULL    |                |
| created_at | datetime     | NO   |     | NULL    |                |
| updated_at | datetime     | NO   |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
7 rows in set (0.01 sec)
```

Hozzuk létre a szótanuló modellünket és a hozzá tartozó kontrollert egy paranccsal. A `Word` modellünkben legyenek string típusú `word`, `description`, `lang` és `pronounce` attribútumok.

```
kovacsg@debian:~/gyakorlat# rails g scaffold Word word:string description:
    text lang:string pronounce:string
      invoke  active_record
      create    db/migrate/20150917130728_create_words.rb
      create    app/models/word.rb
      invoke    test_unit
      create      test/models/word_test.rb
      create      test/fixtures/words.yml
      invoke  resource_route
       route    resources :words
      invoke  scaffold_controller
      create    app/controllers/words_controller.rb
      invoke    erb
      create      app/views/words
      create      app/views/words/index.html.erb
      create      app/views/words/edit.html.erb
      create      app/views/words/show.html.erb
      create      app/views/words/new.html.erb
      create      app/views/words/_form.html.erb
      invoke    test_unit
      create      test/controllers/words_controller_test.rb
      invoke    helper
      create      app/helpers/words_helper.rb
      invoke    test_unit
      invoke    jbuilder
      create      app/views/words/index.json.jbuilder
      create      app/views/words/show.json.jbuilder
      invoke  assets
      invoke    coffee
      create      app/assets/javascripts/words.coffee
      invoke    scss
      create      app/assets/stylesheets/words.scss
      invoke  scss
      create    app/assets/stylesheets/scaffolds.scss
```

Hajtsuk végre a migrációt.

```
kovacsg@dev:~/gyakorlat/app/views$ rake db:migrate
(in /home/kovacsg/gyakorlat)
== 20150917130728 CreateWords: migrating
   ===============================================
-- create_table(:words)
   -> 0.0010s
== 20150917130728 CreateWords: migrated (0.0011s)
   ======================================
```

Majd a böngészőben nyissuk meg a szavak nézetet (`http://localhost:3000/words`), és hozzunk létre egy új szót. Nézzük meg, hogy létrejött-e a rekord az adatbázisban[6][7], nyissuk meg az adatbázis konzolt:

```
kovacsg@dev:~/gyakorlat/app/views$ rails db
SQLite version 3.7.13
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> select * from words;
sqlite> .schema
CREATE TABLE "schema_migrations" ("version" varchar NOT NULL);
CREATE TABLE "words" ("id" INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, "word
    " varchar, "description" text, "lang" varchar, "pronounce" varchar, "
    created_at" datetime NOT NULL, "updated_at" datetime NOT NULL);
CREATE UNIQUE INDEX "unique_schema_migrations" ON "schema_migrations" ("
    version");
sqlite> kovacsg@dev:~/gyakorlat/app/views$ rails db
SQLite version 3.7.13
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> select * from words;
1|||hu||2015-09-17 13:10:45.774187|2015-09-17 13:10:45.774187
```

Nézzük meg, miként tudunk ehhez Rubyból hozzáférni. Nyissunk egy konzolt:

```
kovacsg@dev:~/gyakorlat/db$ rails console
Loading development environment (Rails 4.2.4)
irb(main):001:0> Word.all
  Word Load (2.7ms)  SELECT "words".* FROM "words"
=> #<ActiveRecord::Relation [#<Word id: 1, word: "", description: "", lang:
    "hu", pronounce: "", created_at: "2015-09-17 13:10:45", updated_at:
    "2015-09-17 13:10:45">]>
irb(main):002:0> Word.all[0]
  Word Load (0.3ms)  SELECT "words".* FROM "words"
=> #<Word id: 1, word: "", description: "", lang: "hu", pronounce: "",
    created_at: "2015-09-17 13:10:45", updated_at: "2015-09-17 13:10:45">
```

Hivatkozások

[1] Sam Ruby, Dave Thomas, and David Heinemeier Hansson et al. *Agile Web Development with Rails*, volume Third Edition. The Pragmatic Bookshelf, 2009 Mar.

[6] A második gyakorlaton SQLite adatbáziskezelőt használtunk.

[7] A gyakorlaton jelen lévő hallgatók igen beszédes szót javasoltak, ezért a string attribútum értéke üres.