Web Application Development in Perl 6

Gábor Szabó

Web Application Development in Perl 6

Gábor Szabó

This book is for sale at http://leanpub.com/bailador

This version was published on 2017-12-03

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Contents

Preface

This book is a work in process. The planned publishing date is December 2017, but parts of the book will be published and made available to the readers continuously. This means you will see the raw version of the book. Quite likely with a lot of errors.

In addition Bailador is also in constant development and some of its features change. Hopefully converge to better solutions. That means that parts of the book might be out of date by the time you read it.

Expect a lot of typos, grammatical mistakes, and even factual errors. At least for now.

I really hope that by December Bailador will be stable enough that the book can reflect its features for some time.

Anyway, let's get started. There is plenty of things to read and write.

Bug reporting

If you encounter problems with the book, first please verify you are reading the most recent version of the book. Check the fist page of the book you have for "This version was published on DATE" and compare it to the same line of the Sample chapter[1].

If this is the latest version, then report the problem by sending an e-mail to me: Gabor Szabo szabgab@gmail.com with the following information:

1. The publish date of the book you are reading. You can find "This version was published on DATE" on the first internal page of the book.
2. What operating system do you use? (If Linux then which distribution, which version etc.?)
3. What did you do (list the commands and their output)?
4. What did you expected that did not happen?
5. The output of "perl6 -v".
6. The output of "zef info zef".
7. The output of "zef info Bailador".

If you encounter problems with Bailador itself, please visit the GitHub repository of the Bailador project[2]. There you'll find the "Issues". Check if the issue has been already reported. If it has, you are welcome to comment on the issue adding your report. If not, then please open a new issue. Here too, it would be useful to give the information listed above. You can leave the publishing date of the book out, as that is probably not relevant.

[1] https://leanpub.com/bailador

[2] https://github.com/Bailador/Bailador

Changes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

v0.38 2017-12-03

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

v0.37 2017-09-03

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

2017-08-05

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

2017-07-09

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

2017-07-08

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

2017-06-29

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

2017-06-21

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

2017-06-19

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Supporters

The following people have supported the crowd-funding campaign of the book:

Damian Conway,
Meir Guttman,
JJ Merelo Guervos,
Dave Cross,
Mark Allen,
Heince Kurniawan,
Nikos Vaggalis,
Søren Lund,
Kamil Vratny,
Wolfgang B.,
Mike Schienle,
Zoffix Znet,
Tony Edwardson,
Martin Heinsdorf,
Chris Jack,
John Napiorkowski,
Phil Wells,
Austin Kenny,
Paul Johnson,
Tushar Dave,
Veljko Vidovic,
Robert Bond,
Ekkehard (Ekki) Plicht,
David Adler,
Christian Sánchez,
Jonas Linde,
Kieren Diment,
Issac Goldstand,
Jamie Popkin,
Rick Umali,
Csaba Gaspar,
Offer Kaye,
Scott Walters,
Daniel Green,
Tom Browder,
Yary Hluchan,

Dominique Dumont,
Packy Anderson,
Peter Krumins,
Norman Gaywood,
Liz & Wendy Mattijsen & Van Dijk,
Aldrich Thiel,
Laurent Rosenfeld,
Vincent Boerner,
Martin Barth,
Sven Opitz,
Richard Morse,
Manfred Laner,
Enrique Nell,
Steve Roe,
Peter Scott,
Tim Bunce,
Bruce Van Allen,
Stephen Downes,
Borkur Gudjonsson,
Clemens Hintze,
Uri Bruck,
Rouvillain Philippe,
Torben Lund,
Macer Parkinson-Kemp,
Rina Ron,
Ioan Rogers,
Jarkko Haapalainen,
Christoph Flamm,
Emil Nicolaie Perhinschi,
Steffen Winkler,
Paul Voyer,
Xavier Guzman,
Andrew Bach,
Richard Bychowski,
Richard Noble,
JustThisGuy,
Wolfgang,
Bernhard Schmid,
Filipe Dutra,
Xavier L'Hour,
Leora Betesh,
JeanPaul David Tzaban,

David Klann,
Jim Guloien,
Philip Sharman,
Robin Hill,
Michael Houghton,
Brian Gaboury,
David Mills,
Lianna Eeftinck,
Renée Bäcker,
Neil Hainer,
Nathan Waddell,
Mike Messick,
Uri Zackhem,
Chan Wilson,
Rói á Torkilsheyggi,
Hermann Calabria,
Darin McBride,
Dominic Sonntag,
Nick Logan,
Larry Sherwood,
Bruce Gray,
Steve Piner,
Wenjie Sun,
Gabriel Muñoz,
Paul M. Lambert,
Stephen Hall,
Eric de Hont,
Solomon Foster (colomon),
Kristi Anderson,
Ulrich Reining,
John Karr,
Desmond Daignault,
Jonathan Miller,
A. Sinan Unur,
Markus Monderkamp,
Kıvanç Yazan,
Martha Ryan,
Alexey Melezhik,
Kevin Phair,
Vladimir Ivashchenko,
Philip W. Dalrymple,
Barry Kesner,

Doyle Young,
Zvi Kushnaroff,
Russ Brewer,
Shanta McBain,
Miller Hall,
Lara Ortiz de Montel,
Matija Grabnar,
Troy,
Theodore Ivanov,
Tom Legrady,
Anatolie Mazur,
Miroslav Popov,
Barry Keeling,
Piotr Jaskulski,
Leho Larven,
Håkon Hægland,
Axel Schmidt,
J.K. van Achterberg,
Fritz Zaucker,
Paulo Custodio,
Mohammad S Anwar,
Diarmuid Albers,
Joel Maslak,
Greg Cole,
Gordon Banner,
James E Keenan,
Chris Jepeway,
John Keener,
Gerhard Gonter,
Curtis Jewell,
James Carter,
James Bence,
and several more people who wanted to stay Anonymous.

If you have supported the campaign, but your name does not appear here it might be that you marked your support as “Private” or that I made a mistake. In either case contact me by e-mail so I can correct the issue.

Contributors

The following people have contributed to the success of the book by reading it, trying the code, and giving suggestions:

- Martha Ryan
- Moritz Lenz
- Renée Bäcker
- Martin Barth

Hello Perl 6!

In this chapter you'll get some explanation about Rakudo and Perl 6 and see various ways to set up the basic environment for using this new language.

What are Perl 6, Rakudo, and Rakudo Star?

I know it is confusing a bit. I hope my explanation will help to clear it up a bit.

Perl 6 is a programming language.

Some say it is an upgrade of Perl 5. Others say it is a sister language of Perl 5. Yet others say it is a totally different language and should have been named something else.

Let's try to avoid that discussion and focus on how can we use this language to get some job done. Preferably in a fun way.

In Perl 5 the implementation and the specification of the language is the same.
In Python there is a specification of the language and there are multiple implementations based on C, Java, .Net, and even Python itself. These are called CPython, Jython, IronPython, and PyPy respectively. Even though there is a separate name for the C implementation most people say "Python" and actually mean the C implementation of Python.

In the Perl 6 world this is still formulating, but "Perl 6" usually refers to the specification of the language together with a test-suite that confirms a certain compiler/interpreter implements it correctly. We also usually say "Perl 6 code" to something that can be executed with "Perl 6 compile/interpreter".

There are several implementations of the Perl 6 language but as of November 2017 Rakudo is by far the most advanced one.

Although Rakudo is just one of the implementations of Perl 6, the Rakudo executable usually calls itself "perl6" so most people will probably say "Run this code with Perl 6" even though technically what runs the code is Rakudo.

Anyway so
* Rakudo[3] is one of the implementations of the Perl 6 language. It is a compiler/interpreter. One can download the source code of Rakudo and start from there. We can call this the "Rakudo distribution".
* Rakudo Star[4] is another distribution that contains the Rakudo compiler, some documentation, and some additional modules written in Perl 6. The aim of Rakudo Star is to make it easier to get started.

[3] http://rakudo.org/
[4] http://rakudo.org/

* Galaxy[5] is an experiment by yours truly to create another distribution of Rakudo that contains even more modules to have "batteries included". (Think about lots of stars.)

Community

The Perl 6 community congregates around the #perl6 channel on irc.freenode.net. The central web site of Perl 6[6] has a link to that channel. The web site also provides a lot of resources for learning Perl 6.

Installing Perl 6

OK, so I've just explained Perl 6 is the specification so when people talk about "installing Perl 6" they most likely mean installing something that will run Perl 6 code. Which currently means Rakudo or Rakudo Star.

There are several ways to install Rakudo, we'll cover some of the options here.

Before you install it however it is recommended to remove the previous installations you might have. This probably means removing the directory where you have installed Rakudo earlier and if you made changes to the `~/.bashrc` file setting the `PATH` environment variable, then you should comment out that change and open a new terminal window. You might also need to remove the `~/.perl6/` and `~/.zef/` directories.

Rakudo is the most advanced implementation of Perl 6. We are going to use it on the Moar Virtual Machine. You can install the latest release of Rakudo and then all the tools we need. This is easy on Linux and also on Mac OSX. Rakudo is being released once a month. I have not tried it on Windows.

Rakudo Star is the name of the Rakudo distribution. It contains Rakudo, some documentation, and a bunch of extra modules. It comes precompiled for MS Windows and Mac OSX. It is probably easier to get started with this version, but it might be a bit out of date as it is released only once a quarter.

Rakudo Perl 6 is the name I use for the Rakudo. I don't specifically distinguish the compiler and the distribution and add the "Perl 6" there to connect the two names (Rakudo and Perl 6) in the mind of the readers. Some other people use this name to emphasize the name Rakudo. Yet others think that this name creates additional confusion.

MS Windows

If you have MS Windows on your computer you can either use Rakudo Star that provides a binary installer, or a better approach might be to use Vagrant and Linux in a Virtual Box.

[5] https://github.com/szabgab/galaxy

[6] https://perl6.org/

OSX

If you have Mac OSX you can install the binary distribution of Rakudo Star, you can compile Rakudo from source code, or you too can install Vagrant and use Linux in a Virtual Box.

Linux

If you have Linux on your computer then you either compiler Rakudo Star or Rakudo. You could also use Vagrant and another Linux distribution in a Virtual box, but you probably don't need that.

Linux - from your Linux vendor

As of this writing Rakudo is still being developed at lightening speed and Linux distributions probably cannot keep up with that. So for now using Rakudo that comes with the system is probably not the best solution. We'll skip this for now.

Linux - using Rakudobrew

Rakudobrew isn't really recommended for end users, but you really want to use it here are a few additional instructions:

```
1 rakudobrew build-zef
2 zef install TAP::Harness
```

Docker

Docker[7] can be also used to run Perl 6. It will be described later.

Install and set up Rakudo Star

You can download it from the web site of Rakudo[8]. For MS Windows and Mac OSX you can grab a binary installer. For Linux you download the source code in the tar.gz file

Mac OSX

The recommended installation is using the .dmg file. Once you have downloaded the file from Rakudo[9] you launch it. Then drag the Perl6 icon in the Application folder. Open the README file and follow the instructions there.

[7] https://www.docker.com/
[8] http://rakudo.org/
[9] http://rakudo.org/

MS Windows

There is a msi installer of Rakudo Star that you can download from the Rakudo[10] home page.

Linux - build from source

The minimum requirement to compile Rakudo on Linux is 1 Gb free memory, but your operating system and some services will already use some memory, so you probably need at least 2 Gb memory in the machine.

Make sure you have at least 2 Gb memory. e.g. by running the `free` command:

```
$ free -h
              total        used        free      shared  buff/cache   available
Mem:           2.0G         67M        1.7G        3.1M        177M        1.7G
Swap:            0B          0B          0B
```

Install the prerequisites:

On some Debian-based system (e.g. Ubuntu):

```
sudo apt-get -y install gcc make
```

Let's assume your home directory is called `/home/foobar`, you can follow the following instructions:

```
cd /home/foobar

wget https://rakudo.perl6.org/downloads/star/rakudo-star-2017.10.tar.gz
tar xzf rakudo-star-2017.10.tar.gz
cd rakudo-star-2017.10/
perl Configure.pl --backend=moar --gen-moar
make
make install
```

On one of the systems I tried this the `perl Configure` step took 3 minutes. The `make` step took 4 minutes and the `make install` step took 11 minutes as it installed many extra 3rd-party modules.

Assuming you started the whole process in the `/home/foobar` directory, this will install Rakudo in the `/home/foobar/rakudo-star-2017.10/install` directory.

Then you need to add the path to the perl6 executable to the `PATH` environment variable. If you are using Bash as your shell, which is the most common default, you can do the following:

[10] http://rakudo.org/

```
export PATH=$PATH:/home/foobar/rakudo-star-2017.10/install/bin:/home/foobar/raku\
do-star-2017.10/install/share/perl6/site/bin
```

Remember to replace /home/foobar in there with the directory where you started at.

This will configure the `PATH` environment variable in the current shell. To make this permanent, edit the `.bashrc` file in your home directory (`/home/foobar/.bashrc`) and add the above line to the end of the file. That way every time you open a new terminal,
the `perl6` command will be already available.

To verify this you could type:

```
which perl6
```

and you'd get back:

```
/home/foobar/rakudo-star-2017.10/install/bin/perl6
```

and then you can type

```
which zef
```

and you should get back:

```
/home/foobar/rakudo-star-2017.10/install/share/perl6/site/bin/zef
```

Install Rakudo on Linux

The Rakudo Star distribution is released once every 3 month while Rakudo, the compiler is released once a month. In case you are impatient and would like to use one of the intermediate releases of Rakudo, you can follow the instructions below.

Theoretically this should work on Mac OSX as well and with some additional magic on MS Windows as well, but I have not tried those.

Install the prerequisites. On Debian/Ubuntu/Mint:

```
sudo apt-get -y install gcc make git
```

```
cd ~
wget  https://rakudo.perl6.org/downloads/rakudo/rakudo-2017.10.tar.gz
tar xzf rakudo-2017.10.tar.gz
cd rakudo-2017.10/
perl Configure.pl --gen-moar --gen-nqp --backends=moar
make
make install
```

Configure your `PATH` environment variable: edit the `~/.bashrc` file and add the following line at the end, but please be aware that copy and paste may not work, especially on the tildes:

```
export PATH=$PATH:~/rakudo-2017.10/install/bin:~/rakudo-2017.10/install/share/pe\
rl6/site/bin
```

Then reload the .bashrc file by running `source ~/.bashrc`.

Then you can execute `echo $PATH` and verify that the new `$PATH` has your additions at the end.

At this point you can verify that perl6 can be found and it can be executed. Run the following to see if your shell can find perl6:

```
which perl6
```

It should print something like:

```
/home/foobar/rakudo-2017.10/install/bin/perl6
```

Run the following to see Rakudo reporting its version number:

```
perl6 -v
```

Something like this:

```
This is Rakudo version 2017.10 built on MoarVM version 2017.10
implementing Perl 6.c.
```

Installing zef

In [Rakudo Star]{#install-rakudo-star} you don't need the following as it already includes these tools:

Install zef[11], the command line tool that can be used to install Perl 6 modules:

[11] http://modules.perl6.org/dist/zef

```
cd ~
git clone https://github.com/ugexe/zef.git
cd zef
perl6 -Ilib bin/zef install zef
```

You can verify the installation by typing

```
zef info zef
```

You should see something like this, but probably with a higher version number:

```
- Info for: zef
- Identity: zef:ver('0.1.27'):auth('github:ugexe')
- Recommended By: /home/ubuntu/rakudo-2017.10/install/share/perl6/site
Description:         It's like [cpanm] wearing high heels with a tracksuit
License:         Artistic-2.0
Source-url:          git://github.com/ugexe/zef.git
Provides: 35 modules
Support:
#   bugtracker:        https://github.com/ugexe/zef/issues
Depends: 0 items
```

Install TAP::Harness[12] to get the `prove6` command we'll need in the book.

```
zef install --serial TAP::Harness
```

Install Linenoise[13] to provide readline-like features, such as command history, line editing, and tab completion for builtins in the REPL.

```
zef install --serial Linenoise
```

Development environment using Vagrant

VirtualBox allows you to install a new operating system inside your existing operating system without making any changes to your existing operating system. From the point of view of your current operating system it will be seen just as a directory with a few files in it you can delete any time. This allows you to have a full-blown Linux on top of your Windows machine. Then you can use that Linux machine without creating a mess for others in the family who might be using the same machine.

Vagrant is a command line tool that makes it easy to manage VirtualBox instances.

[12] http://modules.perl6.org/dist/TAP::Harness
[13] http://modules.perl6.org/dist/Linenoise

- Download and install the latest version of VirtualBox[14].
- Download and install the latest version of Vagrant[15].
- Create a directory where you'll have your Vagrant configuration file.
- Open the Terminal or Command Prompt and change directory to the one you've just created.
- Type: `vagrant init ubuntu/zesty64`

This means we base our virtual environment on the Zesty64 release of Ubuntu[16]. According to the releases list[17] Zesty is Ubuntu 17.04 released in April 2017.

By the time you are reading this book Ubuntu might have released newer versions of it Operating System. Feel free to use the most recent version. Most likely the rest of the book and all the examples will work there too.

The output of the `vagrant init` command will look like this:

```
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.
```

It will create a file called `Vagrantfile` in the current directory.

We need to make some adjustment to this file. Edit the Vagrantfile with your favorite text editor and add two lines that will map the ports of the guest operating system. The guest operating system being the one we are going to install now and the host operating system that runs on your computer.

```
  config.vm.network "forwarded_port", guest: 5000, host:5000
  config.vm.network "forwarded_port", guest: 3000, host:3000
```

By default Vagrant allocates 1Gb memory to the guest operating system. In most cases this is enough, but if not, you can increase the allocated memory by changing the `vb.memory` line.

Change the line `vb.memory` to `vb.memory = "2048"`.

This is the Vagrant file I have:

[14] https://www.virtualbox.org/
[15] https://www.vagrantup.com/
[16] https://www.ubuntu.com/
[17] https://wiki.ubuntu.com/Releases

code/Vagrantfile

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

# vagrant --version

# Apparently a while ago Mac stopped the port forwarding
# Vagrant Image port forwarding on Mac
# https://www.danpurdy.co.uk/web-development/osx-yosemite-port-forwarding-for-va\
grant/
# http://apple.stackexchange.com/questions/254654/port-forwarding-on-macos-sierra
#
# vagrant plugin install vagrant-triggers


# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure(2) do |config|
  # The most common configuration options are documented and commented below.
  # For a complete reference, please see the online documentation at
  # https://docs.vagrantup.com.

  # Every Vagrant development environment requires a box. You can search for
  # boxes at https://atlas.hashicorp.com/search.
  config.vm.box = "ubuntu/zesty64"   # 2017.04

  # allow x11 forwarding
  config.ssh.forward_x11 = true

  # Disable automatic box update checking. If you disable this, then
  # boxes will only be checked for updates when the user runs
  # `vagrant box outdated`. This is not recommended.
  # config.vm.box_check_update = false

  # Create a forwarded port mapping which allows access to a specific port
  # within the machine from a port on the host machine. In the example below,
  # accessing "localhost:8080" will access port 80 on the guest machine.
  # config.vm.network "forwarded_port", guest: 80, host: 8080

  # No port forwarding on the Mac any more
```

```
42    config.vm.network "forwarded_port", guest: 5000, host:5000
43    config.vm.network "forwarded_port", guest: 3000, host:3000
44    config.vm.network "forwarded_port", guest: 8888, host:8888
45    config.vm.network "forwarded_port", guest: 80, host:8080
46
47    # Create a private network, which allows host-only access to the machine
48    # using a specific IP.
49    # config.vm.network "private_network", ip: "192.168.33.10"
50    #config.vm.network "private_network", type: "dhcp"
51    # error when running vagrant up:
52    #   /sbin/ifdown eth1 2> /dev/null
53    #
54    #   Stdout from the command:
55    #   Stderr from the command:
56    #
57    #   mesg: ttyname failed: Inappropriate ioctl for device
58
59    # Create a public network, which generally matched to bridged network.
60    # Bridged networks make the machine appear as another physical device on
61    # your network.
62    # config.vm.network "public_network"
63
64    # Share an additional folder to the guest VM. The first argument is
65    # the path on the host to the actual folder. The second argument is
66    # the path on the guest to mount the folder. And the optional third
67    # argument is a set of non-required options.
68    # config.vm.synced_folder "../data", "/vagrant_data"
69    config.vm.synced_folder "/Users/gabor/work", "/vagrant"
70
71    # Provider-specific configuration so you can fine-tune various
72    # backing providers for Vagrant. These expose provider-specific options.
73    # Example for VirtualBox:
74    #
75    config.vm.provider "virtualbox" do |vb|
76    #   # Display the VirtualBox GUI when booting the machine
77    #   vb.gui = true
78    #
79    #   # Customize the amount of memory on the VM:
80      vb.memory = "2048"
81    end
82    #
83    # View the documentation for the provider you are using for more
```

```
  # information on available options.

  # Define a Vagrant Push strategy for pushing to Atlas. Other push strategies
  # such as FTP and Heroku are also available. See the documentation at
  # https://docs.vagrantup.com/v2/push/atlas.html for more information.
  # config.push.define "atlas" do |push|
  #   push.app = "YOUR_ATLAS_USERNAME/YOUR_APPLICATION_NAME"
  # end

  # Enable provisioning with a shell script. Additional provisioners such as
  # Puppet, Chef, Ansible, Salt, and Docker are also available. Please see the
  # documentation for more information about their specific syntax and use.
  # config.vm.provision "shell", inline: <<-SHELL
  #   sudo apt-get update
  #   sudo apt-get install -y apache2
  # SHELL
end
```

- Then you can type in `vagrant up`.

This will start the Virtual Machine. The first time you run this, the command will need to download the image and thus it might take some time. Once it is done you get back the prompt of your host operating system.

- Then you can type the command that will take you into the guest operating system where you'll see a prompt like this:

```
ubuntu@ubuntu-zesty:~$
```

Linux, Mac OSX

The command to ssh into the guest operating system:

```
vagrant ssh
```

MS Windows

On MS Windows you can use Putty[18] to connect to the Vagrant based VirtualBox image.

You will need to download both putty.exe and puttygen.exe. You can then follow these instructions[19].

Once you installed the private key, Putty will let you connect to the virtualbox. It will still prompt for a username though, where you need to type in 'vagrant'.

[18] http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

[19] https://www.sitepoint.com/getting-started-vagrant-windows/

Installing Linux environment

At this point you need to follow the instructions on how to set up the development environment on Linux. Once you are done with that you can do all the rest of the tasks using this Virtual Box.

When you stop working on the project, you can leave the guest operating system by typing `exit` that will take you back to the command prompt of the host operating system. You can now shut down the VirtualBox image by typing: `vagrant halt`.

The next time you'd like to use the VirtualBox image you need to `cd` to the same directory and type `vagrant up`. This time it should be much faster as you already have the image locally. Then you can type `vagrant ssh` and it will move you into the guest operating system.

"Hello World" on the command line

Once you have installed Rakudo, you can open a Terminal window and type in the following:

```
1  perl6 -e 'say "Hello World!" '
```

The result should be

```
1  Hello World!
```

On Microsoft Windows you'd open the Command Prompt and you'll use the quotes in the other way around and type:

```
1  perl6 -e "say 'Hello World!' "
```

The result is the same.

This difference is only relevant when we write on the command line.

Writing one-liners can be useful in certain cases, but for web applications we need to create files. So we won't spend time on one-liners, we go to code in files straight away:

"Hello World" in a plain program

We create a file call "hello_world.pl6" with the following content:

code/ch01/hello_world.pl6

```
1 use v6;
2
3 say "Hello World!";
```

You can run this in the Terminal or on the command line by typing `perl6 hello_world.pl`. The result is the same as above.

The `use v6;` indicates that we need Perl to be at a minimum of version 6. It is important to add this as it will make sure no one will run this program with an older version of Perl by mistake.

We could now go on and learn a lot more about the syntax of Perl 6, but let's go ahead and start building a web application. I'll explain the syntax of Perl 6 as we make progress.

Hello, world!

In this chapter you'll get some background information about Bailador and then we'll set up the basic environment for developing web applications.

What is Bailador?

Bailador is a light-weight, route-based web framework.

It provides a way to start building a really simple web site and then lets you gradually implement more complex tasks. It also provides a way to build modern Microservices.

There are a number of similar frameworks in various languages. Probably the first such framework was Ruby Sinatra[20] then other programming languages got similar frameworks: Perl 5 Dancer[21], Python Flask[22], and Express NodeJS[23] just to name a few.

Bailador tries to take the best ideas from these and integrate in a way that fits the mind of the developers.

History of Bailador

Bailador was created by Tadeusz Sośnierz commonly known as Tadzik in 2011. The first commit in the Git repository of the project is dated Mon Jan 10 21:13:34 2011 +0100.

It was created based on the ideas seen in the Perl 5 Dancer[24] web framework which is itself a clone of the Ruby Sinatra[25] framework.

In early 2016 Martin Barth has started to contribute a lot to the project and on Apr 12, 2016 he became the main contributor.

In May 2017, the Bailador repository[26] was moved to its own GitHub organization indicating that it is now a truly mult-developer project.

---

[20] http://www.sinatrarb.com/
[21] http://perldancer.org/
[22] http://flask.pocoo.org/
[23] https://expressjs.com/
[24] http://perldancer.org/
[25] http://www.sinatrarb.com/
[26] https://github.com/Bailador/Bailador/

Contributors of Bailador

In September 2017 there were 37 people listed as contributors of the project. By the time you are reading this the number will surely be higher. In order to avoid publishing an out-of-date list let me just point you to the GitHub repository[27] where you can find the list of contributors with their exact contributions.

Community

In the Bailador project we use Slack. We can be found on the perl6-bailador[28] Slack channel.

The documentation of the Bailador project can be found on the Bailador web site[29].

Install Bailador

Rakudo Star comes with `zef` the command line package installer. (Think cpan, cpanm for Perl 5, pip for Python). After the Rakudo-only installation we have installed zef manually so you should have it in either case. Once you have Rakudo Perl 6 on your system, you can open the Terminal and type in:

```
zef search Bailador
```

to check which version of Bailador is available.

I got the following output:

```
===> Updated cpan mirror: https://raw.githubusercontent.com/ugexe/Perl6-ecosyste\
ms/master/cpan.json
===> Updated p6c mirror: http://ecosystem-api.p6c.org/projects.json
===> Found 5 results
-------------------------------------------------------------------------------
ID|From                              |Package              |Description
-------------------------------------------------------------------------------
1 |Zef::Repository::LocalCache       |Bailador:ver('0.0.10')|A light-weight r...
2 |Zef::Repository::LocalCache       |Bailador:ver('0.0.11')|A light-weight r...
3 |Zef::Repository::LocalCache       |Bailador:ver('0.0.5') |A light-weight r...
4 |Zef::Repository::LocalCache       |Bailador:ver('0.0.6') |A light-weight r...
5 |Zef::Repository::Ecosystems<cpan>|Bailador:ver('0.0.10')|A light-weight r...
-------------------------------------------------------------------------------
```

[27] https://github.com/Bailador/Bailador/

[28] http://perl6-bailador.slack.com/

[29] http://bailador.net/

Then you can install Bailador by typing:

```
zef install --serial Bailador
```

The output will mention the "missing dependencies". `zef` is going to install all those dependencies and finally it will install Bailador.

It will install the latest version available on CPAN.

The `--serial` option will ensure that prerequisites of Bailador are installed as they are encountered even if some of them or Bailador itself fails to install. If that happens, if Bailador is not properly installed, you can try the above command again. Sometimes this happens.

When you install Bailador it also installs a command-line tool called `bailador`.

Once the installation is done, you can type:

```
bailador help
```

to verify that it was installed and to get the command line options. The output will look similar to this:

```
Usage:
  bailador [--name=<Str>] new
  bailador version
  bailador [-w=<Str>] [--config=<Str>] watch <app>
  bailador [--config=<Str>] easy <app>
  bailador [--config=<Str>] routes <app>
  bailador [--config=<Str>] ogre <app>
  bailador [--config=<Str>] tiny <app>
```

You can also run:

```
bailador version
```

This will print the version number of Bailador. I got:

```
Bailador 0.0.11
```

You can run

```
zef info Bailador
```

to see what `zef` thinks about the installed version of Bailador. I got the following output:

```
- Info for: Bailador
- Identity: Bailador:ver('0.0.11')
- Recommended By: /home/ubuntu/rakudo-star-2017.10/install/share/perl6/site
Description:    A light-weight route-based web application framework for Perl 6
License:    MIT
Source-url:  git://github.com/Bailador/Bailador.git
Provides: 37 modules
Support:
#   source: git://github.com/Bailador/Bailador.git
#   bugtracker: https://github.com/Bailador/Bailador/issues
Depends: 18 items
```

"Hello World" with Bailador

Our first Bailador-based web application is very simple:

code/ch01/hello_bailador.pl6

```
use v6;
use Bailador;

get '/' => sub {
    "Hello Bailador!";
}

baile;
```

Create a file called `hello_bailador.pl6` with the above content (Beware, copy-paste from an e-book usually messes up some of the characters. Especially the quotes, and spaces. You might be better off typing in the content.)

Run it on the command line by typing `bailador easy hello_bailador.pl6` It will immediately print something that looks like this:

```
Entering the development dance floor: http://127.0.0.1:3000
[2017-05-26T15:55:32Z] Started HTTP server.
```

and it will wait for you to use your regular browser and visit the address printed on the screen.

Troubleshooting

If you get a nasty error message on the terminal that has the following line in it:

```
Failed to listen: address already in use
```

that means there is already another application using the same port. It might be that you ran the same Bailador application twice or you have some other application using port 3000. You can either try to locate that other process and kill it, or you can tell Bailador to use a different port. You can do the latter by setting the `BAILADOR` environment variable to `port:PORT` where the PORT is the actual port number you'd like to use.

On Linux you'd write the following:

```
BAILADOR=port:5000 bailador easy hello_bailador.pl6
```

If your browser does not find the page and you are running Bailador in Vagrant image while your browser is on your host operating system, you'll need to tell Bailador to listen on all the hosts, not only localhost. You can do that by setting `BAILADOR=host:0.0.0.0`

Stop the current process by hitting `Ctrl-C` and then run it again.

On Linux run:

```
BAILADOR=host:0.0.0.0 bailador easy hello_bailador.pl6
```

If you'd like to combine the two options you can do so with the following command:

```
BAILADOR=host:0.0.0.0,port:5000 bailador easy hello_bailador.pl6
```

Hello World with skeleton

The `bailador` command line tool also allows you to create a skeleton application. Open your terminal window in a directory where you'd like to create your application. Type in

```
bailador new --name=App-Name
```

Instead of 'App-Name' you can use any other name you'd like to use. The only limitation is that the name should consist of letters and numbers, optionally separated by a dash here and there.

This will print the following:

```
Generating App-Name
App-Name/t/app.t
App-Name/bin/app.pl6
App-Name/views/index.html
```

The command will create a directory called App-Name with all the necessary files to run your first Bailador application. Currently this means a file called `app.pl6` in the `bin` subdirectory that contains a single route, the directory `views` that contains a single template called `index.html`, and the directory `t` that contains the first test case.

You can then change directory to the project directory and run the `app.pl6` file:

```
cd App-Name
bailador easy bin/app.pl6
```

The output will look like this:

```
Attempting to boot up the app
# Entering the dance floor: http://127.0.0.1:3000
! [2017-06-23T03:45:59Z] Started HTTP server.
```

This will launch the built-in web server and will tell you to browse to http://127.0.0.1:3000/

As previously mentioned , if you are in a Virtualbox image, you probably should run:

```
BAILADOR=host:0.0.0.0 bailador easy bin/app.pl6
```

Once the server is running you can use your browser on your computer to visit the given URL.

Testing the “application” so far

As you are building an application you will need a way to make sure it works properly. In many companies this is the task of the QA department, but if you want to reduce the back and forth between you and the QA department then you will make sure you yourself can test your application. In addition if you this is an open source project or a small start up then you don’t have a QA department.
So you have to check you application yourself.

Doing that manually might be fun once or twice, but as you make progress with you application you won’t want to check manually if the whole application still works. It is both very time consuming and very boring. So you’d be much better off writing some more code that will verify that the application works properly and also that if the user misbehaves the application can still handle that.

The best and easiest known way to do this is to write automated tests. Some call these unit-tests. Others call them integration-tests. Yet others say that these are regression tests. Let's not get hang up with the name and just write some test.

Luckily the skeleton of the application already contains the first test that verifies the skeleton itself.

In order to run it stop the server by pressing `Ctr-C` in the window where you ran `bailador easy ...`. You will get back the prompt of your terminal.

At this point you can run the unit-tests that were created when the skeleton of this application was created. Run this command:

```
$ prove6
```

The output will look like this:

```
t/app.t .. ok
All tests successful.
Files=1, Tests=1,  2 wallclock secs
Result: PASS
```

This indicates that the skeleton came with one test file that had one unit-test in it and that successfully passed.

Later we'll see how are these tests written and you will learn how to add your own tests.

Adding to Git

Writing web applications is fun, but losing your changes is not. Using version control is essential. There are many open source version control systems around, Git seems to be the most popular among open source developers. Together with GitHub as a cloud-based hosting service it used by millions of people.

Learning Git and GitHub

I'll show you the necessary steps to set up your project on GitHub, but it is probably a good idea to learn to use Git and GitHub more thoroughly.

GitHub has excellent explanations here[30] both for setting up the git client and using GitHub itself.

There are also some free video courses teaching Git and GitHub:

[30] https://help.github.com/categories/setup/

- Version Control with Git[31]
- GitHub and Collaboration[32]
- How to Use Git and GitHub[33]

Jessica Lord has an excellent and free learning tool called GitIt[34].

I also have the beginning of a book on Collaborative Development using Git and GitHub[35].

I strongly recommend you spend the necessary time to learn Git and GitHub.

Install Git Client

First you need to make sure you have Git installed. On Debian/Ubuntu you can install the git client by typing

```
1 sudo apt-get -y install git
```

Configure Git Client

Then you need to configure two basic things in your git client that will identify you:

```
1 git config --global user.name "Foo Bar"
2 git config --global user.email "email@example.com"
```

You probably want to have your real name there and your real e-mail address, but read the section about privacy in the free sample of my Git book[36] first.

Once you have configured the Git client, you can start using it locally. `cd` into the directory of your project. (e.g. in the App-Name directory if you used the above example to create a skeleton using the bailador command.)
Type in

```
1 git init
```

This will initialize the current directory as a Git repository. Technically it will create a subdirectory called `.git` and put the "Git database" in there. You can look around in that directory, but normally you don't need to do anything inside that directory. The commands of `git` use those files.

Then you can add the first version of your code to the repository:

[31] https://www.udacity.com/course/version-control-with-git--ud123
[32] https://www.udacity.com/course/github-collaboration--ud456
[33] https://classroom.udacity.com/courses/ud775
[34] https://github.com/jlord/git-it-electron
[35] https://leanpub.com/collab-dev-git/
[36] https://leanpub.com/collab-dev-git/

```
git add .
git commit -m "First version"
```

From now on every once in a while, after you made some interesting changes to your application and after you verified that it is working, you can add a new version to your local git repository. In order to do this, go to the root directory of your project (App-Name in our example) and then type:

```
git add .
git commit -m "Some explanation about the change"
```

Adding to GitHub

Having version control locally is much better than not having it at all, but having a copy of your Git repository on GitHub as well has some additional advantages. As a minimum it is a backup. It also makes it easier for you to have a copy of this repository on another machine. e.g. the server where you deploy your code. It also makes it much easier to collaborate with other people.

GitHub allows you to have public repositories free of charge. This is great for open source projects and when you are just playing with the code. GitHub also has a paid version where your code is private to you and to the people you give access to the code. Most of my code is public, but I have several projects in private repositories as well. For example the content of this book is maintained in
a private repository.

Create a user on GitHub[37]

Create a repository on GitHub. If your project locally is called App-Name, then probably the best idea is to call it App-Name on GitHub as well. (If you are logged in to GitHub then there is a + sign in the top right corner of the GitHub site next to your picture. Clicking on it you'll see a menu that has several options, including "New repository". Click that.

TBD.

Setting up Continuous Integration using Travis-CI

When there are several people working on a project it is a good idea to frequently merge the work of all the people and frequently check if they all work together properly. In order for this to work smoothly we need to be able to set up a working environment of our application programmatically. (Without manual intervention.) We also need to be able to run test programs that verify that for given input, or given series of inputs, we get the expected results.

In the extreme case we run all the test on every commit, or at least every time someone pushes code out to our central version control system.

[37] https://github.com/

If we use Git and GitHub then we can use Travis-CI[38], the Travis Continuous Integration service. It is hosted services provided free for projects with public GitHub repository and you can get a paid version for your private GitHub repositories as well.

In our example we have a public GitHub repository we wish to hook up to Travis-CI.

First visit Travis-CI[39] log in there with your credential from GitHub.
This will allow Travis to synchronize the list of your public repositories.

Then click on your name (in the top-right corner) and click on Accounts. That should lead you to your profile page. Mine is https://travis-ci.org/profile/szabgab

There you will see the list of your repositories.

Travis synchronizes this list every few hours, but you can also initiate a sync operation with the "Sync account" button.

Once you see the repository you'd like to integrate click on the grey X flipping the switch to show a v. This enabled Travis-CI for that specific repository.

Now you need to add the Travis configuration file to the repository and push it out.

There is a guide for Perl 6 on Travis-CI[40] you can delve into later. For now you need to create a file called `.travis.yml` (It starts with a dot.)
and have the basic configuration added.

code/ch01/.travis.yml

```
branches:
  except:
    - gh-pages
language: perl6
sudo: false
perl6:
  - latest
install:
  - rakudobrew build-zef
  - zef install Test::META
  - zef install Path::Iterator
  - zef --depsonly install .
env:
  - AUTHOR_TESTING=1
```

Add that file to git, commit the changes and push it out to GitHub.
Within a few seconds Travis-CI will notice the new revision and start to build your project and run your tests using the default test command:

[38] https://travis-ci.org/
[39] https://travis-ci.org/
[40] https://docs.travis-ci.com/user/languages/perl6/

```
PERL6LIB=lib prove -v -r --exec=perl6 t/
```

If something fails you'll get an e-mail notifications. Then you can tweak your code or your test till they pass again.

Auto reloading Bailador when files are changed

As you make changes to your code you'll want to keep looking at the new version of your application. If you ran your application using `bailador easy bin/app.pl6` then you'll have to stop it by pressing Ctrl-C and launch it again for every change.

Lucky for us `bailador` comes with a command `watch` that can automatically reload your application when files change. Instead of `easy` use the `watch` mode to launch your application:

```
bailador watch bin/app.pl6
```

During development you'll have a terminal window running this, another window open with your editor, and a browser open on http://127.0.0.1:3000/

Errors in the code

Let's now see what happens if there is an exception in your code. We will add a new route to our application called `/die` that will throw an exception using the `die` function of Perl 6.

In order to do this we go back to our `hello_bailador.pl6` example. I've created a copy of that file now called `exception.pl6` to make it easier to import in the book, but you can feel free to try this with the same file called `hello_bailador.pl6`.

Change the file adding the following route:

```
get '/die' => sub {
    die 'This is an exception so you can see how it is handled';
    "hello world"
}
```

to have the following code:

code/ch01/exception.pl6

```
use v6;
use Bailador;

get '/' => sub {
    "Hello Bailador!";
}

get '/die' => sub {
    die 'This is an exception so you can see how it is handled';
    return "hello world";
}

baile;
```

Restart the application if necessary and visit `http://127.0.0.1:3000/die` you will see the following error message:

```
Internal Server Error
```

That does not tell you much. If this was a real application that would give you this error you would not be very happy.

If you visit the terminal window you'll see the actual exception:

```
Entering the development dance floor: http://127.0.0.1:3000
[2017-05-30T08:00:10Z] Started HTTP server.
[2017-05-30T08:00:28Z] GET / HTTP/1.1
[2017-05-30T08:00:32Z] GET /die HTTP/1.1
This is an exception so you can see how it is handled
  in sub  at manuscript/code/ch01/exception.pl6 line 9
  in block  at /Applications/Rakudo/share/perl6/site/sources/3B84EEC2F55AC291DAB\
BAD9B46EDE308E526A3DE (Bailador::Route) line 59
  in method recurse-on-routes at /Applications/Rakudo/share/perl6/site/sources/3\
B84EEC2F55AC291DABBAD9B46EDE308E526A3DE (Bailador::Route) line 56
  in method dispatch at /Applications/Rakudo/share/perl6/site/sources/F4258CEFD1\
CCA136ED68D2D056880D8AE47F1964 (Bailador::App) line 157
  in block  at /Applications/Rakudo/share/perl6/site/sources/F4258CEFD1CCA136ED6\
8D2D056880D8AE47F1964 (Bailador::App) line 147
```

That's helpful, but that requires an extra switching between windows and it does not provide you with any additional details.

You can turn on `development` mode to see the error messages in the browser window.

Development mode

Enabling development mode can be achieved on the command line:

```
1 BAILADOR=mode:development  bailador watch exception.pl6
```

Visiting `http://127.0.0.1:3000/die` again you'll see a page with the error message and lots of other details.

TBD: Include a screenshot once it looks better.

Deployment

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Configuring Nginx as a reverse proxy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Configuring Apache as a reverse proxy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Using Docker

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Deploying on Ubuntu

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Database access

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Database access from Perl 6

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Command line program to access the database

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

CRUD (Create, Read, Update, Delete) from the command line

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Configuring MySQL

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Configuring PostgreSQL

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Use SQLite

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

INSERT

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

SELECT

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

User Input

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Echo with Bailador

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Counter that stores the count in a text file.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Processing forms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Input validation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Counter that stores the count in a database.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Chapter 5 - Sessions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Counter

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Cookies

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Session management

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Login/Logout

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Chapter 7 - API and Ajax

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Traditional web sites and Single Page Applications

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Backend programming

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Front-end programming

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Full stack developer

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

API and client-server programming

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

AJAX - interaction between the client and the server.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Data Serialization - Marshalling

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

JSON - JavaScript Object Notation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Creating an API that returns JSON

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Manually testing it

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Testing the JSON route

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Client script

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Echo

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Calculator

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Client side in plain JavaScript

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Client side in JQuery

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

RESTful API

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Simple calculator using Ajax with JQuery

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Split the application into files

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Chapter 8 - Routing

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Setting up simple routes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Routes with parameters

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Routes with wildcards

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Responding to GET request

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Responding to POST requests

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Request and Response objects

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Chapter 9 - Templating system

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Mobile friendly - Make the HTML mobile friendly.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Template::Mojo, the Templating System of Bailador

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Passing data to the templates

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Chapter 10 - Handling errors while serving pages

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Handling 404 errors

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Handling 500 errors

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Handling 30x redirects

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Setting the HTTP header manually

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Debugging

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Chapter 11 - Serving static files

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Serving static files such as images, css, JavaScript files

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Generating robots.txt

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Generating sitemap.xml

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Generating RSS and Atom feed

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

NoSQL

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Using Memcached

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Installing Memcached

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Memcached set, get, and delete

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Memcached counters using incr

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Memcached decr

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Memcached Expiration time

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Memcached cleaning keys using flush-all

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Memcached stats

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Using MongoDB

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Installing MongoDB on Ubuntu Linux

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Check MongoDB manually

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Insert document into MongoDB

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Find documents in MongoDB

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Fully annotated version of insert

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Fully annotated version of find

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Inspect manually and delete the MongoDB database

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Using Redis

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Applications

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Personal Bookmark manager

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

URL shortener

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

TODO list

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Blog engine

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

A job board

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Real estate manager

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Flight ticket vendor for low-cost flights

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Banner ads rotator

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Conference organizer

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Gradual development

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

One-page Static site served by plain web server

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

One-page Static site served by Bailador

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Multi-page Static site served by Bailador

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Add special page

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Cookbook

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Sending e-mail

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Hashing Passwords and verifying them

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Encrypt

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Verify a password

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Enhanced encryption

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Registration process with double-opt-in

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Change Password (when logged in)

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Reset forgotten password process

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

IP based access restriction (2-step verification with e-mail)

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Secure initial configuration of web application

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Uploading files (e.g. images)

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Rate limit

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Appendix

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

What files were installed by a module

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.

Index

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/bailador.