

# Gyro essentials: Installing and setting up a new project

This tutorial shows how to install Gyro and to set up a new project using the `systemupdate` module. This guide is for \*nix systems only.

## Downloading and installing Gyro

First download and extract a Gyro release or check it out from the repository. Downloads are available under <http://www.gyro-php.org/downloads/>, for how to access the repository see <http://code.google.com/p/gyro-php/source/checkout>.

Once you extracted the Gyro files, there is a directory called `gyro`, containing three subdirectories `core`, `install`, and `modules`. This directory is referred to as the Gyro root directory.

Gyro is now installed on your system.

## Setting up a new project

### Creating directory structure

Gyro supports creating new projects with a set of scripts and the `systemupdate` module.

First, the application's directory must be created. There is a shell script to do this, which takes two parameters:

- The path to the project's directory, absolute or relative. Missing directories will be created.
- The path to the Gyro root directory. This must be absolute!

You invoke the script like this:

```
1 {Gyro root}/install/makedirs.sh path/to/your/project {Gyro root}
```

The script will create a couple of directories and copy some basic files required for a Gyro project. Right beneath your project directory you will find the following folders:

- `app`. This is where your application code resides. It is called the application root. There are a couple of subdirectories here for `model`, `view`, `controller`, `behaviour`, and the `web` root.
- `data`. Used for keeping project related data, like specs, documentation, and such.
- `tmp`. Temporary files are stored here, like logs, and compiled views. The webserver must have write access to this directory.

If you use version control: Now is a good time to initially upload your project into your repository.

### Configuring webserver and database

For the next steps, we need the webserver and the database to be ready, so

- Create a virtual host for your project and point it to `app/www` beneath your project directory.
- Create a database and - if you like - a database user for your project.

Additionally, Gyro-PHP depends on the PEAR classes `Mail` and `Mail_Mime`, so these must be installed.

```
1 sudo pear install Mail
  sudo pear install Mail_Mime
```

Now copy `app/config.php.example` to `app/config.php` and adjust the following properties:

```

/**
2  * The domain of the app, excluding 'http://'.
  */
4  define('APP_URL_DOMAIN', 'fill in!');

6  /**
  * DB Constants
8  */
  define('APP_DB_TYPE', 'mysql');
10 define('APP_DB_NAME', 'fill in!');
  define('APP_DB_USER', 'fill in!');
12 define('APP_DB_PWD', 'fill in!');
  define('APP_DB_HOST', 'localhost');
14 /**
  * Mail related
16 */
  define('APP_MAIL_SENDER', 'fill in!');
18 define('APP_MAIL_ADMIN', 'fill in!');
  define('APP_MAIL_SUPPORT', 'fill in!');

```

`APP_URL_DOMAIN` contains your application's host, like "www.gyro-php.org". Note there's no leading `http://` and no ending slash.

The `APP_DB_*` constants contain the name, user, password, and host of your database. Currently only "mysql" is fully supported as type, so don't change this.

The `APP_MAIL_*` constants contain mail addresses:

- `APP_MAIL_SENDER`: The mail address to be used as sender by default, when sending mail.
- `APP_MAIL_ADMIN`: The admin's mail address. This mail gets system mails send.
- `APP_MAIL_SUPPORT`: Mail for contacts, feedback, and everything that involves real people.

## More configuration options

Gyro allows several instances of your project to run, like a development version, a staging distribution, and a live system. These are called installations. Therefore, configuration is split in two files:

- `app/config.php`: This is for local settings, like the database, the host, and the Gyro path.
- `app/constants.php`: This contains configuration settings that are the same for all installations, like the title of the application, the language, encoding, and so on.

Take a look at `app/constants.php` and change it to fit your needs, too.

If you use version control, it is a good idea to exclude `app/config.php` from your repository.

## Running `systemupdate` to install tables

While everything now is configured properly, the Gyro core tables are still missing in the database. Fortunately, we can let the `systemupdate` module do all the required steps.

To use `systemupdate`, we also need the Gyro console. Open `app/modules.php` in your favorite text editor and make sure the modules `console`, `systemupdate` and `staticmainpage` are enabled, that is these three lines are not commented out:

```

1 Load::enable_module('console');
  Load::enable_module('systemupdate');
3 Load::enable_module('staticmainpage');

```

The `staticmainpage` module isn't really needed, but allows to easily check if everything is running.

If all required modules are enabled, run

```

1 {Gyro root}/modules/console/install/install.sh /path/to/your/project

```

This will copy `run_console.php` to your application root directory. Change to your project directory and run

```
1 php app/run_console.php systemupdate
```

If everything is OK, there should be a couple of success messages being displayed.

If you now point your browser to your project domain, you should see the staicmainpage default message: *"It works, GYRO is ready to be used on your system"*. Congratulations!