

Home

This page last changed on Jan 20, 2006 by [gbevin](#).

Drone is a Java IRC bot built on the [RIFE](#) framework.

Drone has a modular API which makes it possible to easily extend and customize the active feature set. Drone sports a modern web administration interface to handle all common tasks and a public logging section. Installation is done by simply dropping a war in your servlet container or running the standalone distribution.

Features

- Abstracted IRC protocol
- Easy configuration
- Multiple bots can run on multiple servers in multiple channels
- Intuitive module API
- User-friendly web interface
- Ships with useful modules
 - do
 - faq
 - help
 - httperror
 - identify (with nickserv)
 - log
 - ping
 - quit
 - regexp
 - register (with nickserv)
 - say
 - seen
 - version
- Runs inside or outside a servlet container
- Database storage of collected data
- Powerful web search of logged conversations
- Web interface for posting messages to IRC remotely

Try it out now by reading how to [install](#) a binary distribution.

Playing

This page last changed on Jan 07, 2005 by [gbevin](#).

The rest of the documentation of your bot lives in the bot itself. You just have to message it on IRC and ask for help, for instance:

```
/msg Drone help
```

If the nickname of your bot is Drone.

public site

Don't forget to visit the public logging interface from the following url if you ran it as standalone:
<http://localhost:8080/>

You need to adapt this of course if you ran it in your servlet container.

remote IRC messaging

Several users have requested a way to be able to automate the posting of IRC messages through Drone. You can find more information about how to use it [here](#).

Have fun with Drone!

Remote Messaging

This page last changed on Jan 07, 2005 by [gbevin](#).

You can connect to the remote messaging interface through the following url:

<http://localhost:8080/message>

You need to adapt this of course if you run it from your existing servlet container or if you modified the Jetty configuration files of the standalone distribution.

Log in with the credentials of a user that has the `remote` role. If you haven't modified the [users permissions](#), the login will be:

drone

and the password:

password

You'll be presented with the following interface.

You first have to select which **bot** this message will have to be posted with.

The **target** is either a *nickname* or a *channel*, channels are prefixed with a hash (#) character.

You're free to enter any **message** you like.

UWYN Drone


authid 436b99fe2ab143261ebe21afe5ea9fdf

bot

target

message

Drone v1.1 © 2002-2005 [Uwyn](#)



Since most people will use this feature in an automated fashion to for example post commit messages,

it's also possible to access the interface through a REST API that provides everything in a single URL. The syntax is the following (*everything needs to be on one line and without spaces*):

```
http://localhost:8080/message?  
  submission=credentials&  
  login=yourlogin&  
  password=yourpass&  
  submission=sendMessage&  
  bot=yourbot&  
  target=%23channel&  
  message=yourmessage
```

If your message is long, it's probably best to submit the parameters through a `POST` request. The parameters are exactly the same.

Installation

This page last changed on Jan 20, 2006 by [gbevin](#).

Drone comes in two flavours:

- a zip archive that contains everything to get it running both as a web application or as a standalone bot:
[drone-1.4.zip](#)
- a war archive that can be used as a web application with an existing servlet container:
[drone-1.4.war](#)

standalone zip archive

You can just unzip this archive at the directory of your choice, nothing more is required.

war archive

This web application archive has to be deployed in your servlet container. Since the configuration files are inside the archive, it's best to install it as an unpacked web application and edit the files directly. You can of course also unpack the war file, modify the configuration files and create a new war file that can be deployed as usual.

Drone is tested with [Jetty](#) and [Tomcat](#).

If you previously ran **version 1.0** of Drone, you'll have to perform some tasks to [upgrade](#) to the new version.

If you previously ran **version 1.1** of Drone, you'll have to perform some tasks to [upgrade](#) to the new version.

If you previously ran **version 1.2** of Drone, you don't have to do anything upgrade.

If you previously ran **version 1.3** of Drone, you don't have to do anything upgrade.

After the installation you'll probably want to [configure](#) Drone.

Upgrade 1.0 to 1.1

This page last changed on Feb 15, 2005 by [gbevin](#).

configuration file changes

You should make sure that the following files are updated with those of the new version since they contain modules that are required by the search engine:

```
participants.xml
participants_web.xml
scheduler.xml
```

It's also important to add the new `LUCENE_DIR` parameter to the file

```
config.xml
```

Failure to do so will result in run-time errors because the search engine will not know where to store the indices. More information can be found [here](#).

enabling the search index

The major new feature in this release is the web search interface that's built on top of [Lucene](#). This however requires your database structure to be updated and a search index to be created of your existing logs.

To do this, you simply have to go to the following url:

<http://localhost:8080/admin/reindex>

You need to adapt this of course if you run it from your existing servlet container or if you modified the Jetty configuration files of the standalone distribution.

Log in with the credentials of a user that has the `setup` role. If you haven't modified the [users permissions](#), the login will be:

```
setup
```

and the password:

```
password
```

You'll be presented with the following interface, you just have to press the 'Start indexing' button.


UWYN Drone

Re-index the stored logs for the search engine log out

[home](#) [reindex](#) [search](#) [faq](#) [log](#) [seen](#)

You can start the re-indexing of the logs here.

Note that this can take a long time if your logs are large.

Drone v1.1 © 2002-2005 [Uwyn](#) 


Only one indexing operation can be active at the same time, and you'll get a clear status page that indicates the progress as precisely as possible.


UWYN Drone

Re-index the stored logs for the search engine log out

[home](#) [reindex](#) [search](#) [faq](#) [log](#) [seen](#)

The re-indexing of the logs is currently busy.

 **26%** 11'25"
 127246/479915

Drone v1.1 © 2002-2005 [Uwyn](#) 

To upgrade to the latest version, you still have to perform some [additional tasks](#).

Upgrade 1.1 to 1.2

This page last changed on Feb 15, 2005 by [gbevin](#).

configuration file changes

Additional tags have been added to the [bot configuration](#) and the [global configuration](#). They all have sensible default values, so you're not required to updated your files if the new tags are not important for you.

database structure changes

To make it possible for Drone to work on more databases we had to change the database structure a bit, enter the following SQL statements in your SQL console to upgrade your database structure:

```
ALTER TABLE indexfield RENAME COLUMN index TO isIndex;  
ALTER TABLE indexfield RENAME COLUMN store TO isStore;  
ALTER TABLE indexfield RENAME COLUMN token TO isToken;
```

The upgrade is finished!! Start [playing around](#) around with your updated minion!

Contact

This page last changed on Jun 18, 2004 by [gbevin](#).

Don't hesitate to contact me at the following obfuscated email address:

[gbevin at codehaus dot org](mailto:gbevin@codehaus.org)

The mailing lists are still being set up.

Running

This page last changed on Jan 07, 2005 by [gbevin](#).

When you run the bot the first time, it's best to run it with the web interface to be able to install the database structure.

From the war archive

If you're using Drone as a web application in an existing servlet container (the war archive), then all you have to do next is to *start or restart your container*. The web interface will be loaded and the bot will connect to all the servers and channels you've configured.

From the standalone zip file

If you're running Drone from the standalone zip file, than you can use the included scripts in the `bin` directory to start the bot.

The scripts `drone.sh` and `drone.bat` run the bot and the web interface.

The scripts `dronebot.sh` and `dronebot.bat` only run the bot.

Now that your bot is running, [install the database structure](#) from the web interface.

Install Database

This page last changed on Jan 07, 2005 by [gbevin](#).

You can connect to the administration interface through the following url:

<http://localhost:8080/admin>

You need to adapt this of course if you run it from your existing servlet container or if you modified the Jetty configuration files of the standalone distribution.

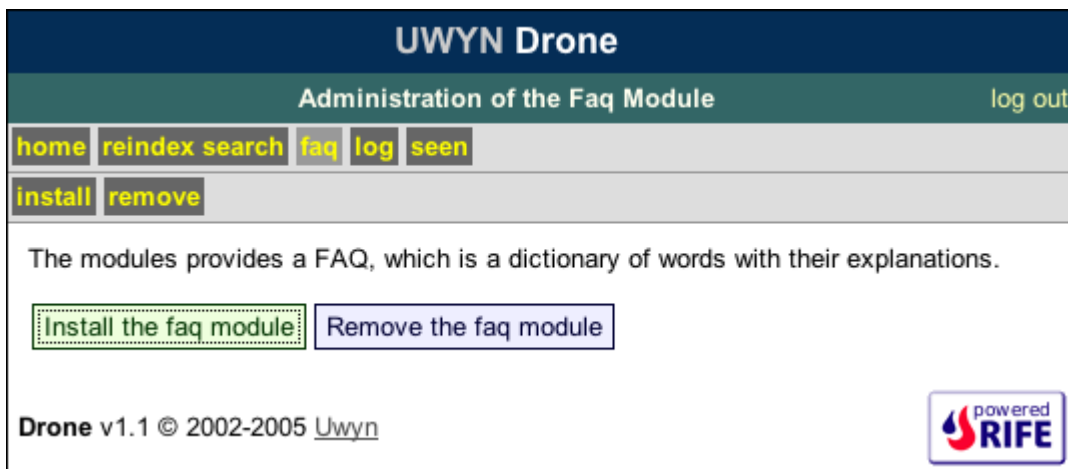
Log in with the credentials of a user that has the `setup` role. If you haven't modified the [users permissions](#), the login will be:

setup

and the password:

password

You'll be presented with the following interface, you just have to visit each module (faq, log, seen) and press the 'Install' button. The corresponding structure will each time be installed.



The screenshot shows the 'UWYN Drone' administration interface for the 'Administration of the Faq Module'. The page has a dark blue header with the title 'UWYN Drone' and a 'log out' link. Below the header is a navigation bar with buttons for 'home', 'reindex', 'search', 'faq', 'log', and 'seen'. A secondary bar contains 'install' and 'remove' buttons. The main content area features a text block: 'The modules provides a FAQ, which is a dictionary of words with their explanations.' Below this text are two buttons: 'Install the faq module' (highlighted with a dashed border) and 'Remove the faq module'. At the bottom left, it says 'Drone v1.1 © 2002-2005 Uwyn' and at the bottom right is a logo for 'powered RIFE'.

Setup is finished!! Start [playing around](#) around with your new minion!

Configuration

This page last changed on Jan 07, 2005 by [gbevin](#).

Drone's configuration is done by editing some XML files.

You'll find these in:

```
config/rep
```

for the standalone distribution,

and in

```
WEB-INF/classes/rep
```

for the war file.

The configuration consists out of different parts:

- [configure the bot](#)
- [configure the database](#)
- [configure global settings](#)
- [configure user permissions](#)

Once everything is configured, you're probably aching to [run the bot](#).

Configuration Bot

This page last changed on Feb 15, 2005 by [gbevin](#).

The most important file is `drone.xml`. It allows you to configure your bot exactly as you want.

Below is a sample configuration file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE drone SYSTEM "/dtd/drone.dtd">

<drone>
  <server name="codehaus">
    <output max="1500" amount="25" interval="1000"/>
    <timeout value="300000"/>
    <charset name="UTF-8"/>
    <address name="irc.codehaus.org" port="6667"/>
  </server>

  <bot name="testbot"
    nick="TestBot" altnick="TestBot_"
    realname="UWYN's Drone Bot (http://drone.codehaus.org/)"
    servername="codehaus">

    <channel name="#drone"/>

    <module classname="com.uwyn.drone.modules.Version"/>
    <module classname="com.uwyn.drone.modules.Ping"/>
    <module classname="com.uwyn.drone.modules.Help"/>
    <module classname="com.uwyn.drone.modules.Faq"/>
    <module classname="com.uwyn.drone.modules.Log"/>
    <module classname="com.uwyn.drone.modules.Seen"/>

  </bot>
</drone>
```

You can use the tag overview below to customize this to your needs.

If you use modules that store information, you'll probably want to [configure the database next](#).

Supported tags

<drone>

This is the root tag and has to be used to start the configuration.

<server>

Declares a new server.

The following attributes are available:

name	the name of the server that will be used later to
-------------	---

	refer to it
--	-------------

<output>

Configures the output rate at which messages can be sent to the server. It's important to tune this well, otherwise the bot might be kicked from the server due to excess flood.

This tag is optional, if it's omitted, sensible default values will apply.

The following attributes are available:

max	The amount of bytes that can be spewed out in one go when the buffer is empty, if more text is available the buffer will be filled and slowly emptied.
interval	The interval at which the buffered text will be emptied.
amount	The amount of bytes that will be sent at each interval

<timeout>

Configures the amount of time Drone waits for data on the server socket. If no data is received in time, Drone pings the server and expects a response. If the response doesn't arrive within the timeout delay, Drone will reconnect to the server.

This tag is optional, if it's omitted, it will default to 5 minutes.

The following attributes are available:

value	The timeout value in milliseconds.
--------------	------------------------------------

<charset>

Configures the character set that Drone will use to decode the data it receives from the IRC server.

This tag is optional, if it's omitted, it will default to UTF-8.

The following attributes are available:

name	The name of the character set.
-------------	--------------------------------

<address>

An address at which the server can be reached. You can add as many `address` tags as you want, they will

be tried in the order of declaration in case some of them might not be reachable.

The following attributes are available:

name	the domain name or ip address of the IRC server
port	the port where it can be reached, this attribute is optional and defaults to 6667 if it's not specified

<bot>

Creates a new bot that will run on an earlier defined server.

You can create as many bots as you like and the shipped modules will separate the information according to the bots that you configured. So if you want to maintain different FAQs for different channels, just create different bots.

The following attributes are available:

name	The unique name of this bot. This name will be used to refer to the bot and to store the information in the database.
nick	The nickname that the bot should use on IRC.
altnick	The alternative nickname that the bot should use in case the primary nickname is taken.
realname	The real name of the bot that will show when people request information about your bot through IRC.
servername	The server name to which the bot will connect. This has to correspond to the name of an earlier defined server.

<channel>

Makes the bot join a channel.

You can define as many channels as you want, the bot will join them all.

The following attributes are available:

name	The name of the channel that has to be joined
-------------	---

<module>

Defines a module that should be activated for this bot.

You can define as many modules as you want, they will be activated in the order you declare them.

The following attributes are available:

classname	The fully quantified class name of a Java class that extends the Module interface
------------------	---

Configuration Database

This page last changed on Feb 15, 2005 by [gbevin](#).

Drone only needs a database if you use modules that need to store information. This setup is thus only needed when you use the `faq`, `log` or `seen` module.

The database is configured as a [RIFE datasource](#) and it is setup in the `datasources.xml` file.

Below is a sample datasource configuration file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE datasources SYSTEM "/dtd/datasources.dtd">
<datasources>
  <datasource name="postgresql">
    <driver>org.postgresql.Driver</driver>
    <url>jdbc:postgresql://localhost:5432/drone</url>
    <user>drone</user>
    <password>password</password>
    <poolsize>10</poolsize>
  </datasource>
</datasources>
```

It should be pretty trivial to configure this. Drone has currently been tested with [PostgreSQL](#), [MySQL](#), [McKoiSQL](#), [Apache Derby](#), [Hsqldb](#) and [DaffodilDB](#).

The JDBC drivers for PostgreSQL and MySQL are included in Drone, to be able to use the other databases you have to download the required jars from their websites and put them in your classpath. Look at the website of your favourite database for installation instructions.

To activate the datasource you have to [configure the global settings](#).

Configuration Global

This page last changed on Feb 03, 2007 by [gbevin](#).

Since Drone runs under the [RIFE](#) framework, all global settings of RIFE apply.

The global configuration can be modified by editing `config.xml`.

Below is an example configuration file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE config SYSTEM "/dtd/config.dtd">
<config>
  <param name="DRONE_DATASOURCE">postgresql</param>
  <param name="DRONE_TIMEZONE">GMT</param>
  <param name="MAX_SEARCH_RESULTS">20</param>
  <param name="LUCENE_DIR">/tmp/drone-lucene</param>
  <param name="DRONE_DEBUG_OUTPUT">>false</param>
</config>
```

As you can see, Drone understands the following specific settings:

DRONE_DATASOURCE

You have to enter the name of your datasource here. Since it's possible to declare several datasources in the `datasources.xml` file, Drone has to know which one to use.

DRONE_TIMEZONE

Contains the name of the timezone that Drone will use to format and parse dates and times.

MAX_SEARCH_RESULTS

Drone is capable of searching logfiles by receiving commands through IRC. The results will be returned to the user as private messages. By entering a number here, you can setup how many results will be returned maximum.

LUCENE_DIR

You have to enter the name of the directory where the search index of the logs will be created. This directory has to be writable by the user that runs the bot.

DRONE_DEBUG_OUTPUT

Turns on debug information that will be printed to the console.

It's recommended to [configure the user permissions](#) before putting your bot online.

Configuration Users

This page last changed on Apr 01, 2005 by [gbevin](#).

The web interface of Drone offers an authenticated administration interface. This interface allows administrators to change the bot's running status and maintainers to install or remove the database structures that are required for the modules.

The users and passwords are setup in the file `users.xml`, for example:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<credentials>
  <user login="drone">
    <password>password</password>
    <role>admin</role>
    <role>remote</role>
  </user>
  <user login="setup">
    <password>password</password>
    <role>admin</role>
    <role>setup</role>
  </user>
</credentials>
```

The tags should be totally intuitive. People with the `admin` rule will not be able to install or remove the database structure and people with the `setup` credentials have full permissions. The `remote` permission allows people to access Drone's web interface that allows you to remotely post messages to IRC.

Now that everything is configured, you're probably aching to [run the bot](#).

Securing your password

Since it's often not recommended to store your password in clear text, you can use RIFE's password encryption methods to encrypt or obfuscate your password before putting them in this file.

The following command on the console will print out the usage help:

```
java -cp lib/rife-0.8.2.jar com.uwyn.rife.tools.StringEncryptor
```

In short, if your password is 'humdiedum' and you want to encrypt it with a SHA algorithm, you do this:

```
java -cp lib/rife-0.8.2.jar com.uwyn.rife.tools.StringEncryptor -e SHA:humdiedum
```

and you get

```
SHA:lZThBSPqkCkq3st9sGshP7+RVwI=
```

You can use this string instead of your password in the `users.xml` file. The supported encryption prefixes are: SHA, MD5 and OBF.

