Forrestbot Web Interface

Table of contents

This webapp implements a website staging application for sites built with Apache Forrest. The webapp hosts HTML for a number of managed websites. The user can regenerate these sites at will. Once the user is satisfied with the site, they may 'deploy' the site, making it live.

1 Build

To build the webapp from source, you need Apache Maven installed. From a command prompt, go to the `forrest/tools/forrestbot/webapp` directory and execute "`maven war`".

2 Requirements

- Forrest
- A servlet container such as Apache Tomcat
- Directories for:
  - forrestbot buildfiles
  - source and working files
  - log files
  - built websites (this is not their deployed location)
- A webserver to view the sites in the build directory

The server does not need local access to the sites' source nor final deployed location. The buildfiles configure the transfer of those files.

3 Installation

> **Note:**
> Most of this configuration is related to the OpenSymphony OSUser authentication layer.

1. Extract the war to an empty directory
2. Modify welcome.local.html
3. Modify WEB-INF/classes/osuser.xml if you want to use an authenticator other than XML files
4. Modify projects.xml, groups.xml, and users.xml (all in WEB-INF/classes) according to your needs
5. Optionally modify logging via WEB-INF/classes/log4j.properties
6. Modify WEB-INF/classes/settings.properties according to the server's specific setup
7. Deploy the webapp in Tomcat or some other servlet container (you may want to repackage the war file)
8. Make sure tomcat has permissions to write in forrestbot's build, work, and log directories.

Optional: set up "cron jobs" to execute forrestbot. This is completely independent of the forrestbot web interface, but is often useful. A suggested practice is to build the sites regularly and deploy them as desired through the web interface.

4 FAQ

4.1 The build and/or deploy commands don't seem to work

> **Warning:**
> The only environment variable available is FORREST_HOME, which is set by forrest-home in the settings.properties file. Use a wrapper script (see below) if you need other environment variables.

Currently forrestbot only logs the forrest part of it's execution, not the whole thing. And the web interface starts a forrestbot process and doesn't watch it's progress directly.

Set debug-exec=true in settings.properties and make sure log4j.properties logs at the DEBUG level. This will log all the thread output.

Then look at the debugging output for the command executed and the working directory used. Log into the server as the use the Tomcat server uses and cd to the working directory and then execute the command. This will fairly accurately simulate what the web interface does.

4.2 Can I do anything with this besides just running the forrestbot?

Sure. Create a wrapper script and specify it as forrest-exec in settings.properties. Here's an example:

```
#!/bin/bash

# get env vars
. /home/user/.profile

# group writable for easier sharing of files with others who run forrestbot
umask g+w

# you can preprocess something here
xmlfile=$2
if [ "${xmlfile##*.}" = "xml" ]               # everything after last .
then
    target=$3
    if [ "$target" != "deploy" ]               # don't run preprocess task on a deploy
    then
        projectTarget=${xmlfile%.*}              # everything before last .
        # do something special for $projectTarget here
    fi
else
        echo "Syntax: forrest_wrapper.sh -f myapp.xml [build|deploy|...]"
        exit
fi

# run forrest(bot) with all the parameters
forrest $*

# you can postprocess something here
```

5 TODO Wishlist

- seperate authorization of 'build' and 'deploy'
- log everything, not just the build
- view old log files
- ability to reload all config files on the fly
- put date at top of viewlog_body page