Contributing to Apache Droids

Table of contents

1 Introduction

The Droids Project is an Open Source volunteer project released under a very liberal license. This means there are many ways to contribute to the project - either with direct participation (coding, documenting, answering questions, proposing ideas, reporting bugs, suggesting bug-fixes, etc..) or by resource donations (staff time, conference presentations, publicity, software) and even general hardware/money donations via the ASF.

To begin with, we suggest you to subscribe to the Droids mailing lists (follow the link for information on how to subscribe and to access the mail list archives). Listen-in for a while, to hear how others make contributions.

You can get your local working copy of the latest and greatest code (which you find in the Droids module in the Labs Subversion code repository). Review the todo list and the issue tracker, choose a task. Perhaps you have noticed something that needs patching, or have a new feature to contribute. Make the changes, do the testing, generate a patch, and discuss on the dev mailing list. (Do not worry - the process is easy and explained below.)

Document writers are usually the most wanted people so if you like to help but you're not familiar with the innermost technical details, don't worry: we have work for you!

2 Help Wanted Here

We would be glad to have extra help in any of the following areas:

- Assisting to improve documentation.
- Testing Droids (especially its less-frequently-used features) on various configurations and reporting back.
- New samples, droids and plugins to concisely describe and demonstrate features. Such samples can also enable automated testing.
- Debugging - producing reproduceable test cases and/or finding causes of bugs. Some known bugs are informally listed on To Do, and some are recorded as issues (see explanation below).
- Providing new use-cases and requirements. If you think that Droids does not quite meet your needs then tell us about it.
- Specifying/analysing/designing new features - and beyond. If you wish to get further involved with this, please join the `labs` mailing list, install and try out Droids and read some of the mail archives. You should have a reasonable fluency in Java technologies, some Spring and Ant skills, and a basic understanding of the Droids architecture - don't just say "it should have XYZ" without reading anything first - because chances are, somebody has already thought of that feature!)
- ... and there is just one other thing - don't forget to tell everyone who asks, how great Droids is! The more people that know about and start to use Droids, the larger the pool of potential contributors will be.

3 Procedure for reporting bugs and issues and enhancement suggestions

If you think that you have found a bug or you have a suggestion for improvement, then please discuss it on one of the mailing lists. However, please check our Issue Tracker first as it may be already reported.

The Apache Droids Issue Tracker collates our known issues. Obviously not every issue is listed there. Some issues have been discussed on the mailing list but do not yet have an issue recorded.

The Roadmap is the best way to get an overview. The Unscheduled list also needs regular review, and committers will schedule some of those for the next release.

When creating a new issue, please provide a concise Summary Title and a short Description. Add further information as Comments and include links to the mail archives. The normal procedure is to discuss the issue on the mailing list and then add relevant notes to the issue tracker, otherwise it becomes cluttered.

4 SVN Usage

An overview of how to use Subversion (SVN) to participate in Droids development. Do not be afraid - you cannot accidently destroy the actual code repository, because you are working with a local copy as an anonymous user. Therefore, you do not have the system permissions to change anything. You can only update your local repository and compare your revisions with the real repository. The Installing Droids document explains.

5 Procedure for Raising Development Issues

There are two methods for discussing development and submitting patches. So that everyone can be productive, it is important to know which method is appropriate for a certain situation and how to go about it without confusion. This section explains when to use the developer mailing list and the issue tracker.

Research your topic thoroughly before beginning to discuss a new development issue. Search and browse through the email archives - your issue may have been discussed before. Prepare your post clearly and concisely.

Most issues will be discovered, resolved, and then patched quickly via the developer mailing list. Larger issues, and ones that are not yet fully understood or are hard to solve, are destined for the issue tracker.

Experienced developers use the issue tracker directly, as they are very sure when they have found a bug and when not. However, less experienced users should first discuss it on the user or developer mailing list (as appropriate). Impatient people always enter everything into the issue tracker without caring if it is a bug of Droids or their own installation/configuration mistake - please do not do this.

As a rule-of-thumb, discuss an issue on the developers mailing list first to work out any details. After it is confirmed to be worthwhile, and you are clear about it, then submit the bug description or patch via Bug Tracking.

Perhaps you do not get any answer on your first reply, so just post it again until you get one. (But please not every hour - allow a few days for the list to deal with it.) Do not be impatient - remember that the whole world is busy, not just you. Bear in mind that other countries will have holidays at different times to your country and that they are in different time zones. You might also consider rewriting your initial posting - perhaps it was not clear enough and the readers eyes glazed over.

6 Coding Guidelines

We use coding standards so that Droids source code is easy to understand, to maintain, and to enhance.

We are using the commit-then-review approach. Still, **nothing** untested should be committed. First test then commit!!!

> **Note:**
> We do not use the `@author` tag in java code and the corresponding tags in other format. We think this helps feel more like community development.

- Only produce code conforming to the Java code conventions, as they are set down on the page CodeConvTOC.doc.html
- Indentation is 2 spaces for any type of file (being java or xml or ...). No tab stops.

7 How to prepare and contribute patches

If you use the current development version of Droids via Subversion, then do '`svn update; svn status`' to see what files that you have changed. Do '`svn diff > mypatch.txt`' to make a patch which includes every change. To make a patch for a specific file, do `svn diff src/documentation/content/xdocs/install.xml > install.xml.diff`'. It is better to prepare the patch from the $DROIDS_HOME directory so that it contains a definite path to the document. However, be careful that the patch does not contain other work-in-progress.

For more information about working with SVN, see Version Control with Subversion - the opensource SVN book.

> **Note:**
> Please send all contributions via our issue tracker. and specify the Droids version or svn version of the source.

It is always a good idea to check the Droids issue tracker before diving in.

8 How to revert changes in SVN

Check out http://svnbook.red-bean.com/en/1.0/ch04s04.html#svn-ch-4-sect-4.2 for instructions on how to revert (roll back) changes to svn.

9 Contribution Notes and Tips

This is a collection of tips for contributing to the project in a manner that is productive for all parties.

- See general ASF Tips for email contributors
- Every contribution is worthwhile. Even if the ensuing discussion proves it to be off-beam, then it may jog ideas for other people.
- Use sensible and concise email subject headings. Search engines, and humans trying to browse a voluminous list, will respond favourably to a descriptive title.
- Start new threads with new Subject for new topics, rather than reusing the previous Subject line.
- Keep each topic focused. If some new topic arises then start a new discussion. This leaves the original topic to continue uncluttered.
- Whenever you decide to start a new topic, then start with a fresh new email message window. Do not use the "Reply to" button, because threaded mail-readers get confused (they utilise the `In-reply-to` header). If so, then your new topic will get lost in the previous thread and go unanswered.
- Prepend your email subject line with a marker when that is appropriate, e.g. `[Proposal]`, `[RT]` (Random Thought which quickly blossom into research topics :-), `[STATUS]` (development status of a certain facility).
- When making changes to XML documentation, or any XML document for that matter, use a validating XML editor.
- Remember that most people are participating in development on a volunteer basis and in their "spare time". These enthusiasts will attempt to respond to issues. It may take a little while to get your answers.
- Research your topic thoroughly before beginning to discuss a new development issue. Search and browse through the email archives - your issue may have been discussed before. Do not just perceive a problem and then rush out with a question - instead, delve.
- Try to at least offer a partial solution and not just a problem statement.
- Take the time to clearly explain your issue and write a concise email message. Less confusion facilitates fast and complete resolution.
- Do not bother to send an email reply that simply says "thanks". When the issue is resolved, that is the finish - end of thread. Reduce clutter.
- You would usually do any development work against the trunk of SVN.
- When sending a patch, you usually do not need to worry about which SVN branch it should be applied to. The maintainers of the repository will decide.

- Keep all project-related discussion on the mailing list. It is much better to utilise the wider audience, rather than to break off into private discussion groups. You never know who else will have the answer to your issues, and anyway other people are interested in the outcome.
- Become familiar with the mailing lists. As you browse and search, you will see the way other people do things. Follow the leading examples.