Contribution Guidelines

We appreciate all forms of project contributions to MADlib® including bug reports, providing help to new users, documentation, or code patches.

Before You Start:

- You should have some idea about the project and its purpose. If you don't, start here: http://madlib.incubator.apache.org/
- You should be familiar with the project contents. If you are not, look at below links:
  - Documentation and existing modules: http://madlib.incubator.apache.org/docs/latest/
  - Ideas for contribution: Ideas for Contribution
- You should have some knowledge on relational databases, SQL, Python, C/C++, and the analytical method you want to implement.

Step 1: Make friends with GitHub

- Create an account on GitHub.com.
- If you are not familiar with this version control software follow bootcamp guides on help.github.com to gain some confidence.

Step 2: Find your task

- Create an account on MADlib JIRA and review the open issues (new module, feature request or bug fix).
- Check again the Ideas for Contribution page.
- Get in touch with us on MADlib Dev Forum
  - Tell us what you'd like to work on OR,
  - We can discuss the most pressing needs and suggest a task.
- Once we have a consensus we'll open a JIRA ticket (bug, task, or new feature) to track the progress.

Step 3: Fork MADlib® project

- Go to GitHub MADlib repo and fork the project using the FORK button (top right).
  - This step will create a "tracked" copy of apache/incubator-madlib repo under *yourGitHubAccount*/incubator-madlib.
  - Need some help with forking? Check here

Step 4: Develop away...

... by using your own fork of the MADlib repository. You can follow the Quick Start Guide for Developers to see an example of a new module.

Make sure you have all the module components in place: Module Anatomy

And remember about:

- Proper SQL API: SQL API Guide
- C++ coding practices: The C++ Abstraction Layer
- Documentation instructions: Documentation Guide (Doxygen)

Step 5: Send pull-request

Done with coding? Follow this to get your code checked-in:

- Test build/install/execution on your side.
- Create a pull-request from your forked repository.
  - Use different topic branches to separate your commits into few pull-requests.
  - If this sounds like Greek: read more about pull-requests here: http://help.github.com/send-pull-requests/.
  - Use an informative commit message. See this post for a good set of guidelines. Example below.
    - `[Line 1] Module Name: Main purpose in less than 50 chars`
    - `[Line 2] (blank)`
    - `[Line 3] Jira: MADLIB-12345`
    - `[Line 4+] Description`
- Update the corresponding MADlib JIRA ticket
  - Mark the issue "For Review"
  - Add a comment pointing to your pull request.
  - If you know who would be the best reviewer assign the JIRA to that person. Otherwise don't worry.
- Code comments are easy to add on Github. JIRA better serves as a high level log tracker.
- Once the code is merged by a Committer:
  - Github pull request closes automagically
  - JIRA must be resolved manually, this will be done by the Committer who merges the pull request.