# WikiGraph

Mark Jordan, Jeremy Lenz, Robert McClure,
Austin Nakamura, Michael Rush, Khanh Tran,
Thomas Van Doren

## Alpha Release Documentation

Draft 1

7 February 2011

## Access to Web Application

The WikiGraph flash client may be accessed at the following URL:

http://wikigraph.cs.washington.edu/

Many of the URL endpoints in the WikiGraph SDS are not active (or even available) at this time. Please bear with the team as these are completed. Some sample available inputs include “Ruby” and “Ruby_Programming”.

## Source Control and Build Instructions

Information about cloning the repos, branches, and pushing commits can be found on the WikiGraph Wiki:

http://goo.gl/Bieaz.

There are two branches, dev and release. Dev contains the bleeding edge of WikiGraph source code (all features being developed). Release contains a stable release version, in this case the version that was deployed for alpha.

A one-step build script is in the repository in the build directory. It is complemented by a deploy script. Use:

```
./build.sh <branch>
./deploy.sh <branch>
```

The default branch is dev. ‘release’ is the only other option.

The build script will clone the reposity, checkout the dev branch (or release branch if specified in the parameter), tar and zip the needed files into the build directory, and delete the clone. The deploy script will unzip archives in a local build/<branch> directory into a tmp folder, then ask the user for a valid cseid to login to cubist (the user will then have to provide valid cubist login credentials), login to cubist, securely copy the files into the appropriate directories, and then remove the tmp directory.

The deployment and build are split up so that a developer may use the build script and then deploy locally. Eventually (before beta), the entire build will be continuous. The Hudson CI service will be configures and used to keep a constant build of all branches (release and dev).

## Bug Tracking System Instructions

Bug tracking is done by the issue feature of google code, which can be accessed by clicking on the ‘issues’ tab of the wiki-map google code main:

http://code.google.com/p/wiki-map-uw-cse/

Bug tracking can be accessed by both users and developers for bug reporting.

Under the issues tab, a list of the current bugs is displayed in a summarized table format. If you would like to see more information regarding a specific bug, such as the thread of comments and changes, click anywhere on the row of the bug. To view an older bug, or search for a specific name or type bug, enter a search category and search term in the search controls above the table.

To report a bug, click ‘New issue’ on the left side of the page and enter the relevant information about the bug. Several templates specify where and what kind of bug is to be reported. A short text field allows the person reporting the bug to describe briefly what the bug is about. The main description of the bug is described in a large text box with a general outline of what things to report.

After submission, a bug may be modified by clicking on the row in the table identifying the bug. This will bring up several control options, including adding comments, changing status, changing owners, or adding labels. Depending on your account privileges, a subset of these options may be available.

## Database Access Instructions

The current database is stored on cubist and uses MySQL. It can be accessed by following these steps:

Create an SSH connection to cubist.
(This step requires you to have a University of Washington Computer Science account to perform)
This can be done with the ssh executable, which these directions assume is installed and can be found in your system path. To connect, run the following command:

```
ssh -l <your_user_name> cubist.cs.washington.edu
```

replacing `<your_user_name>` with your UW CSE username. The system will prompt for your password, and you should enter your UW CSE password.

Run the MySQL command:

```
mysql -u rmcclur -p
```

When prompted for a password, enter:

```
d5RbiX3z
```

Next, select the database with the following command:

```
use rmcclur_test;
```

Now you can enter SQL commands to access the data. For example, the command:

```
SELECT * FROM page WHERE page_title = ‘Ruby_Programming’;
```

will show you the page information for the page titled Ruby_Programming.

```
SELECT pl_to FROM pagelinks WHERE pl_from = 844;
```

will show you the page IDs of pages that the Ruby_Programming (page ID of 844) links to.