An Application Programming Interface (API) for Programmable Access to Animal QTLdb

Zhi-Liang Hu*, Carissa A. Park, James E. Koltes, Eric Fritz-Waters and James M. Reecy
Department of Animal Science and Center for Integrated Animal Genomics, Iowa State University, 2255 Killdeer Hall, Ames, IA 50011

Abstract

The development of the Animal QTLdb has been successful in terms of its growing content and numbers of utilities, users, and citations by journal papers. The success of the Animal QTLdb is partly attributable to the web, which made it possible for many remote users to access the database simultaneously through interfaces designed to serve various pre-defined purposes. We have strived to continue to develop the web interfaces by implementing new functions, query options, and online analysis tools, to assist users with hypothetical and requested types of data query. To date, we have developed more than 70 user options implemented in 20+ CGI programs. Since the development of these tools was driven partly by our motivations and partly by user demands, and they are implemented on the server side, the users who alter the way data are queried and summarized is limited. Recently we have started to develop a programmable access (application programming interface, API) platform for users to write their own scripts to query data. This interface allows users to script their own logics and query combinations, and possibly set up a cron job to make the queries automatic. To showcase, we have implemented a half dozen API portals to solicit accessible scriptable access to the database. The advantage of this platform is that users can modify and run their scripts at any time, obtain data or data summary, and flag certain data with their own filters, directly into a biofocused or other preformatted format.

Framework and Implementation

The Animal QTLdb API is implemented following REST (REpresentational State Transfer) architecture. The REST is an intercomputer framework designed to be scriptable, reliable, and automated data request and transport, and is one of the most widely used architectural styles to build web services. Because it uses coordinated constraints on components, connectors, and data elements for distributed information transportation through well-known HTTP protocol (Figure 2), it is relatively straightforward to program client-side applications. We have implemented a server-side API platform using Perl/CGI. It serves the client-side API program calls through Apache web server with data queried from the backend MySQL database.

Scheme and Usage

We designed the Animal QTLdb API to follow human intuition in the way one would go about looking into a database, in three logical steps: (1) “info/Help”/query for an overall idea of what the database has and how the data are organized, or how to use the API platform; (2) keywords “query” for targeted information; and (3) data “fetch” to obtain the data located in the above steps. These logical functions are implemented in three server-side programs called “info”, “query”, and “fetch”. In Box 1 is shown what’s required and how client-side API programs can be structured.

Table 1. Programs and parameters used in the QTLdb API. (a) Basic programs and parameters; (b) Additional parameters.

Discussion

The Animal QTLdb API is designed for structured data to be queried and transmitted in an unambiguous manner. We adhere to XML for reliable data transfers and parses on the client side. Although it’s possible to support other data formats, such as JSON, YAML, and CDV, for the time being, we leave it to client-side programs to convert to or from a needed data format.

References