inBloom

Learning Map and
inBloom Index System

*Data Model and Server*

# inBloom Index **Query Editor Developer Guide**

22 Jan 2013

Applied Minds, LLC

1209 Grand Central Ave Glendale CA 91201

# Contents

# Background

## The Learning Map Data Model

The Bill & Melinda Gates Foundation (the Foundation) supports the implementation of the Common Core State Standards for US K-12 education. The Foundation awarded a contract to Applied Minds, LLC (AMI) to develop a Learning Map Data Model (LMDM) with the goal of it becoming a standard for educational technology infrastructure. The Learning Map will provide the organizing framework that maps the relationship between learning objectives, including dependencies and higher level groupings. It will also allow educational media resources, such as courses, books and web content to be linked to learning objectives. Curricula aligned to the standards will exhibit great diversity in highlighting paths through the Learning Map, and a suite of tools will allow authoring and visualization of the Learning Map. We believe that this data model will eventually enable the creation of online learning tools that are more responsive to the individual needs of a student. The LMDM is inspired by and based on the philosophy of the more general Knowledge Web (See: http://edge.org/conversation/aristotle-the-knowledge-web).

## inBloom Technology

The Foundation has, in collaboration with the Carnegie Corporation, initiated an ambitious effort, Shared Learning Infrastructure (SLI), now inBloom Technology, to provide a new technology infrastructure that supports the Common Core Standards and the Foundation's vision, to be implemented by inBloom. AMI has been contracted by inBloom to build an implementation of a Learning Map and Learning Registry System, suitable for third-party software developers to populate content and develop applications.

## InBloom Index Query Editor

As part of this effort, AMI has delivered an inBloom Index Server, formerly called the Learning Registry Index (LRI) Server, with an associated API and Common Core Middleware. This document is intended for software developers, and provides useful information on the Query Editor that AMI delivered to inBloom.

# inBloom Index Query Editor

The InBloom Index Query Editor is a standalone application which can be used to write and debug queries to an InBloom Index Server instance.

The application also provides a simple user interface for navigating through graph data administered by the InBloom Index Server.

## Getting Started

- If you haven't done so already, install Google App Engine.
  - You can download the appropriate package here: `http://code.google.com/p/googleappengine/downloads/list`
- Clone gaesessions from github into lriqueryeditor directory. This provides sesssion support for the Query Editor.
  - `https://github.com/dound/gae-sessions.git`
- Once this is done, go to gae.pth and edit the second line to reflect the path to your GAE folder.
- Make sure to source into the current directory by typing the following in your shell: `source bin/activate`
- If you plan on using InBloom Login, make sure that the values in `configs.py` match the information from your InBloom Application Registration. It may be necessary to register your instance of the InBloom Index Query Editor as a new application.
  - More information can be found here: `http://dev.slcedu.org/docs/adding-your-application-sli/how-application-registration-works-slc`
- **`Optional:`** Set up a link to the `dev_appserver.py` by typing the following into Bash: `ln -s PATH/TO/google_appengine/dev_appserver.py`
- Start the development server by typing `dev_appserver.py . -p <PORT> -a <IP>`, where both the `-p` and `-a` tags are optional. If these tags are ommitted, the Query Editor will run at 127.0.0.1:8080.
- **`Note:`** The Query Editor must be pointed at an LRI Server instance. To do this, click on the cog icon at the top right and make sure that the hostname and port match your InBloom Index Server Instance.

# Features

## Query Editing

The left column of the Query Editor is devoted to writing and submitting queries. Enter the body of the query in the main editor pane. Remember that these queries must be valid JSON. The buttons above the main editor pane allow you to tailor the verb (search, create, update) and object (property, entity) of your query, as well as to add options, such as cache or detail.

Once you've written your query, press Run and the response will appear in the right column.

## Browsable Responses

The Query Editor's left column provides three different views of the InBloom Index Server's response to your query. The default is "Tree" view, which consists of a navigable graphic representation of the server response. Clicking on different properties will collapse or expand sections of the response for easier viewing. Clicking hyperlinked values will run another query, bringing the hyperlinked entity into focus. Use this feature to easy traverse the InBloom Index graph. External URLs (to Learning Resources, for example) will open in a new window.

Responses are also provided in text/raw JSON format and in a compressed "Status" format. The latter can be used to quickly diagnose system issues, such as query speed.

## SLC Login

Users can log into their SLC Accounts from the InBloom Index Query Editor. Doing so will automatically sign all writes to the InBloom Index Server with their SLC UUID.

Logged-out users can sign writes with an admin access token (defined in `lriserver/lri_config.json`).

## Query History

The last four queries are preserved in the application's query history. This history can be accessed by clicking the History button, or by clicking the leftmost panel divider.

## Save and Share Queries

Queries can be saved and shared by clicking the Save Query button. This will archive the present application state (including the state of the tree browser) in the GAE development datastore and provide you a fixed URL. This URL can be opened later and will return the query as it was originally run.