|  |
| --- |
| **GLIMMPSE: File Service API** |

Contents

[**GLIMMPSE: File Service API** 1](#_Toc292963110)

[1 Introduction 1](#_Toc292963111)

[2 The File Service 1](#_Toc292963112)

[3 File Uploads 2](#_Toc292963113)

[4 Save 3](#_Toc292963114)

# Introduction

The GLIMMPSE software system provides a web based user interface to estimate power and sample size for the general linear multivariate model (GLMM) with or without a baseline covariate. The GLIMMPSE system consists of five main components:

* GLIMMPSE.com user interface - a Javascript/GWT front-end user interface
* **Power Service** - Java web service which processes power/sample size requests
* Chart Service - Java web service which produces graphs. For the GLIMMPSE system, this service produces power curves.
* File Service - Java web service providing upload/save functionality for study design information
* JavaStatistics library - low level library containing routines for computing GLMM power

This document describes the API for the File Service. It assumes basic familiarity with [HTTP](http://www.w3.org/Protocols/), [HTML forms](http://www.w3schools.com/html/html_forms.asp), and the GLMM. For additional background on GLMM power calculations when controlling for a baseline covariate, please see Glueck and Muller.

# The File Service

The File Service component of the GLIMMPSE system is a Java web service which provides file save and upload capabilities for the GLIMMPSE user interface. The service itself is quite generic, and basically echoes data or the contents of an uploaded file back to the caller with appropriate headers to initiate browser "save as" or "submit complete" events.

The GLIMMPSE user interface calls this service to allow users to save/upload study design information, and to save power results as a CSV file which can be used with other spreadsheet or statistical software packages. The service is implemented using the [Restlet Framework](http://www.restlet.org/).

Requests to the file service are received via an [HTML form](http://www.w3schools.com/html/html_forms.asp) submission from the GLIMMPSE user interface. The File Service was developed and tested for use within the [Apache Tomcat Server (v 6.0)](http://tomcat.apache.org/).

# File Uploads

File uploads requests echo the contents of an uploaded file back to the GLIMMPSE user interface, or other calling application. File upload requests are sent from a [Google Web Toolkit (GWT) "FileUpload" component](http://google-web-toolkit.googlecode.com/svn/javadoc/1.5/com/google/gwt/user/client/ui/FileUpload.html) within the GLIMMPSE user interface, and parsed in the File Service with the [RestletFileUpload extension](http://www.restlet.org/documentation/1.0/ext/org/restlet/ext/fileupload/RestletFileUpload.html).

The RestletFileUpload extension depends on the [Apache Commons FileUpload library](http://commons.apache.org/fileupload/), and complies with [RFC 1867, "Form-based File Upload in HTML"](http://www.faqs.org/rfcs/rfc1867.html). Under this standard, file upload requests are submitted using the POST method and a content type of "multipart/form-data". The RestletFileUpload component then parses the uploaded file data into a list of FileItem objects.

File uploads are requested through the following URI

POST /file/upload

The entity body should be multipart/form-data encoded with the following structure:

-----------------------------21358143628469

Content-Disposition: form-data; name="file"; filename="study.xml"

Content-Type: text/xml

*… contents of study.xml file …*

-----------------------------21358143628469--

where

* **name** - the name of the file upload form element, which should always be called "file"
* **filename** - the name of the file which was uploaded

The resulting file contents are returned to the caller as media type text/html. In the GLIMMPSE user interface, the file contents are processed when an ["SubmitCompleteEvent"](http://google-web-toolkit.googlecode.com/svn/javadoc/1.6/com/google/gwt/user/client/ui/FormPanel.SubmitCompleteEvent.html) is received.

# Save

To save data from the client browser, the data is sent to the File Service via a form submission. The contents of the "data" field in the form submission are echoed back to the caller with the following headers attached:

Content-type: application/force-download

Content-disposition: attachment; filename=*filename*

These headers trigger the client's browser to display the save dialog. The user can then open the file or download it to their local machine.

Saving of data is requested through the following URI:

POST /file/saveas/

The entity body is form encoded with the following format:

filename=*filename*&data=*data*

where

* **filename** - the default filename displayed in the browser's "save as" dialog box
* **data** - contents of the file to be saved

Author: Sarah Kreidler

Last Updated: May 12, 2011