LightWAVE: Waveform and Annotation Viewing and Editing in a Web Browser

George B Moody

Massachusetts Institute of Technology, Cambridge, Massachusetts, USA

Abstract

This paper describes LightWAVE, recently-developed open-source software for viewing ECGs and other physiologic waveforms and associated annotations (event markers). It supports efficient interactive creation and modification of annotations, capabilities that are essential for building new collections of physiologic signals and time series for research.

LightWAVE is constructed of components that interact in simple ways, making it straightforward to enhance or replace any of them. The back end (server) is a common gateway interface (CGI) application written in C for speed and efficiency. It retrieves data from its data repository (PhysioNet's open-access PhysioBank archives by default, or any set of files or web pages structured as in PhysioBank) and delivers them in response to requests generated by the front end. The front end (client) is a web application written in JavaScript. It runs within any modern web browser and does not require installation on the user's computer, tablet, or phone. Finally, LightWAVE's scribe is a tiny CGI application written in Perl, which records the user's edits in annotation files.

LightWAVE's data repository, back end, and front end can be located on the same computer or on separate computers. The data repository may be split across multiple computers. For compatibility with the standard browser security model, the front end and the scribe must be loaded from the same domain.

1. Introduction

Since the author and his colleagues established the NIH-sponsored PhysioNet resource [1, 2] in 1999, it has provided researchers, clinicians, educators, and students with free access to large, diverse, and growing collections of physiologic signals and time series for research and education, via its web site (http://physionet.org/) and mirrors run by volunteers world-wide. PhysioNet's open-access data archive, PhysioBank, provides more than 50 data collections that include recorded electrocardiogram (ECG), electroencephalogram (EEG), electromyogram (EMG), invasive and noninvasive blood pressure, respiration, blood gas, and other signals and time series, ranging in duration from minutes to weeks. Most of these recordings are accompanied by detailed annotations that mark the locations of individual events (such as heart beats) and other features of interest.

In order to locate and examine data that may be relevant to their studies, PhysioNet's users need to view physiologic waveform and their associated annotations. PhysioNet itself, and many members of its community of users, also create and revise annotations in the course of creating new data collections and curating existing ones. Since the inception of PhysioNet, we have relied for these capabilities on WAVE[3] (Waveform and Annotation Viewer and Editor), an X11/XView application that we created on SunOS in 1989 and later ported to Solaris, GNU/Linux, Mac OS X, and MS-Windows. WAVE has become increasingly expensive to maintain as evolving X11 technology introduces incompatibilities with the XView library that provides WAVE's GUI. A replacement built on modern standards has been clearly needed.

The author has recently designed a lightweight successor, LightWAVE (http://physionet.org/lightwave/), that provides WAVE's features and more while running in any modern web browser without requiring installation in the user's computer, tablet, or smart phone. As for all software from PhysioNet, this application is open-source software that can be freely used, studied, and adapted by its users.

2. Architecture

LightWAVE consists of a user interface (client), a primary server that fetches and transmits data requested by the client, and a scribe (a secondary server that records data transmitted by the client). These components communicate via the user's web browser and the LightWAVE host's web (HTTP) server.

2.1. Server

The LightWAVE server runs as a CGI application within a web (HTTP) server. The web server collects user requests (typically made using the LightWAVE client running in a web browser on another computer) and forwards them to the LightWAVE server. The LightWAVE server parses the requests, obtains the requested data from local storage or from a data repository such as PhysioNet, constructs appropriate responses, and passes them back to the web server. The web server then relays the responses (typically after compression) to the users.

The server is written in C for speed and efficiency. It uses the libcgi library [4] to provide standard CGI functions, including parsing the URL query strings passed from the client. Five query types and six parameters are recognized by the server (see table 1).

Table 1. Query parameters and types.

db	Database name (e.g., `edb`)
record	Record name (e.g., `e0103`)
signal	Signal name (e.g., `V4`) or number
annotator	Annotator name (e.g., `atr`)
t0	Time from start to first sample of interest (H:M:S)
dt	Duration of interval of interest (seconds)
dblist	Get the list of databases
rlist	Get the list of records within **db**
alist	Get the list of annotators within **db**
info	Get signal names, gains, frequencies ... for **db/record**
fetch	Get data specified by **signal**, **annotator**, **t0**, and **dt** from record **db/record**

Access to the data repositories is via the WFDB (Waveform Database) library [5], which provides uniform access to data files in many supported formats. The library was originally written to read and write local files only, but read-only access to http:// and ftp:// URLs was added by M. Dakin in 1999 using libwww [6], and in 2005 by B. Moody using libcurl [7], incorporating support for reading https:// URLs as well. The use of the WFDB library allows the LightWAVE server to fetch data from multiple sources, which may include both local files and remote web servers.

The LightWAVE server returns data in JSON/JSONP format [8] in response to well-formed requests from any client (not limited to the standard LightWAVE client). It has been tested successfully using the Apache web server on Fedora 64-bit Linux and Ubuntu 32-bit Linux; it does not have explicit system dependencies and should be usable with Apache and other servers that support the Common Gateway Interface standard on other platforms.

2.2. Client

The LightWAVE client runs within the user's web browser, communicating with the LightWAVE server to obtain raw data that the client formats and presents to the user. From the data, the client generates SVG code rendered by the browser to produce high-quality graphical output (see figure 1).

The client is a web application written using HTML5, Javascript, JQuery, and JQuery UI. Annotations can also be created and edited using mouse and touchscreen interfaces. Edit logs are saved in persistent HTML5 local storage to preserve them across sessions. Edits can be exported as PhysioBank-compatible annotation files by transmitting edit logs to LightWAVE's scribe.

Browser and platform compatibility

The LightWAVE client is platform-independent, and runs in any web browser that supports HTML5, JavaScript, and SVG. Chrome, Internet Explorer (MSIE) 9, and Safari currently provide the best performance because of their notably fast JavaScript engines; Firefox and Opera also work well, as do MSIE 6, 7, and 8 with the Google Chrome Frame (GCF) plugin; see table 2.

Table 2. Earliest known compatible versions of major browsers.

	Android	iOS	Linux	Mac OS X	Windows
Chrome	18	23	7	7	7
Firefox	15		4	4	4
MSIE					9
MSIE+GCF					6
Opera	12		12.11	12.14	12.14
Safari		5.0		5.1	5.1

3. Scribe

The LightWAVE scribe is a secondary server that imports edit logs transmitted from the client and saves PhysioBank-compatible annotation files. The scribe is a Perl script that uses CGI.pm to handle uploads; it invokes *patchann*, an application that uses the WFDB library to read and write annotations.

Edit logs contain the names of the record and of the source and destination annotators. The source annotator is the name of the existing set of annotations, if any, that the user has modified, which *patchann* merges with the edits recorded in the log to create the new destination annotation file. If a set of edits was created from scratch, there is no source annotator named in the edit log, and *patchann* writes the log's contents into the destination annotation file.

Same-origin policies [9] enforced by all modern browsers require that the scribe and the client must be loaded from the same domain.

4. Access to restricted data

PhysioNetWorks is a service of PhysioNet that offers password-protected workspaces to members of the PhysioNet community for works in progress that will be made publicly available via PhysioNet when complete. The

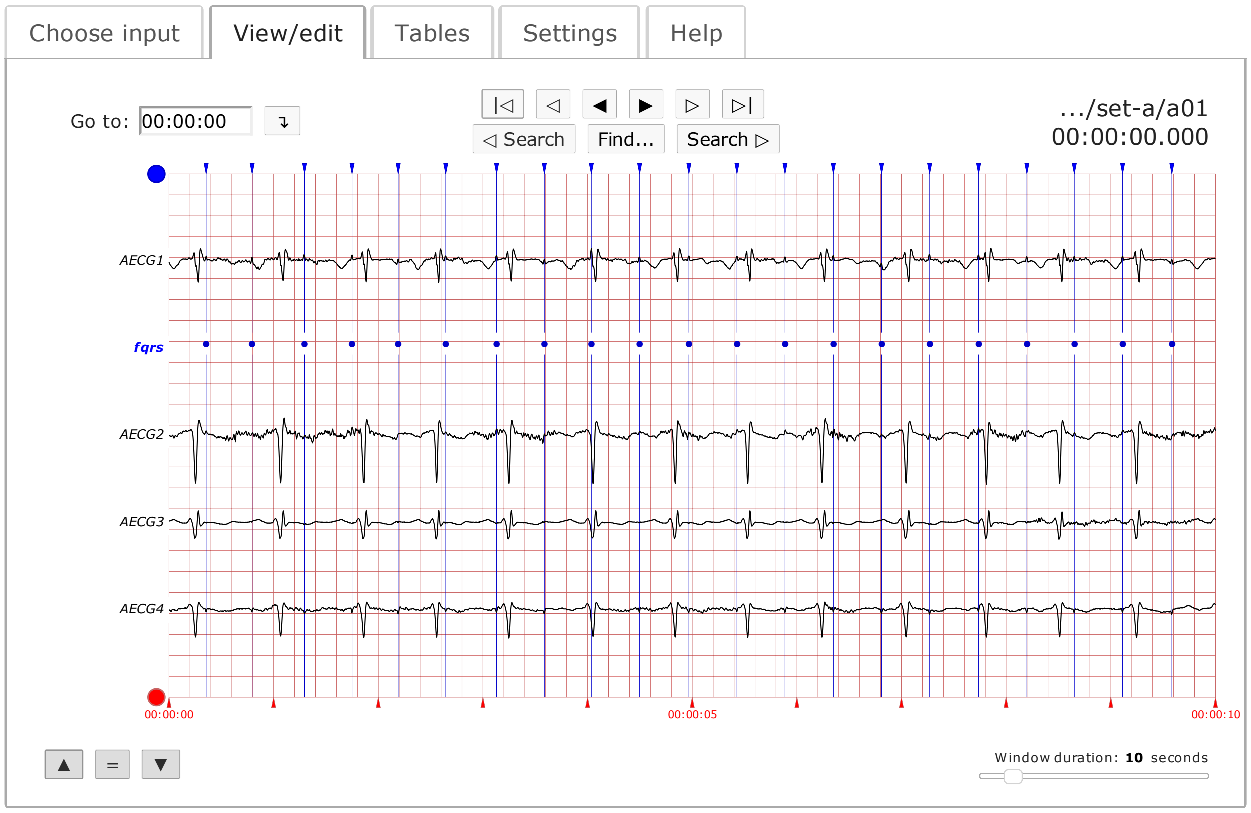

Figure 1. LightWAVE vector screenshot (see text). In an electronic copy of this paper, the figure can be magnified without introducing raster artifacts.

contributor of the data (the *project owner*) controls access to the workspace, and may share it with others, who become members of the project. If the project contains a PhysioBank-compatible data collection, its owner may provide a project-specific LightWAVE client, server, and scribe for the use of project members.

Access to a PhysioNetWorks project is controlled by PhysioNet's Apache web server, which permits files belonging to the project, including the project-specific LightWAVE server, to be accessed by project members only.

The data files that can be read by the LightWAVE server are restricted to those that can be found in its WFDB path (an ordered list of locations to be searched for input files). The public LightWAVE server's WFDB path, which is set by code within its *setrepos* function, does not include any locations within PhysioNetWorks. In a project-specific server, *setrepos* inserts the project's data repository in the WFDB path, thus allowing the server to read the project data.

Any LightWAVE client may set its *server* variable to connect to a project-specific LightWAVE server (although the connection will be successful only if the user has been authenticated as a member of the project). For convenience, the value of *server* in a project-specific client is preset to point to the appropriate server. Similarly, the *scribe* variable specifies where the LightWAVE client transmits any annotations created by the user, and in the project-specific client the value of *scribe* is preset so that the user's annotations are saved in a location accessible to the user and the project owner only.

This flexibility allows either the project owner or any member to create a custom LightWAVE client able to access the project. It is not necessary to restrict access to the client, because access control is enforced by the web server. A custom client may use the project-specific scribe if it is loaded from PhysioNetWorks; otherwise, it is necessary to use a scribe hosted on the same domain as the client, and to upload the saved annotations to the project in order to share them.

5. Discussion and conclusions

A large baseline set of useful features for a waveform and annotation viewer and editor emerged from over 20 years of experience with WAVE. Developing a replacement for WAVE offered the opportunity to revisit design decisions that can be made differently given current technology.

Functionally, LightWAVE and WAVE are quite similar; architecturally, they are very different. WAVE is a compiled application that communicates with the user via an X11 server running on the user's computer. Usually, WAVE runs on the same computer as the X11 server. WAVE accesses data, which may be stored locally or on a remote web server such as PhysioNet, via the WFDB and curl libraries. Customizing WAVE requires a good working knowledge of C, X11, XView, and the WFDB library, and a substantial (though free) software development toolkit.

In contrast, LightWAVE consists of a front end (client) that runs in the user's web browser, communicating with a pair of back end (server) applications that run as components of a web server. Usually, the front end and back end run on different computers. The back end accesses data via the WFDB and curl libraries, in the same way that WAVE does. Customizing LightWAVE requires only a text editor and a knowledge of Javascript.

Although LightWAVE has been described above in the context of PhysioNet and PhysioNetWorks, it should be noted that LightWAVE can be used completely independently if desired. Complete installation guides for LightWAVE's components are included in its on-line help. The server protocol is designed to ease development of alternative front ends with the aim of encouraging experimentation, and at least two such alternatives were developed by others during LightWAVE's beta testing period.

LightWAVE's on-line help also includes a list of projects for extensions and enhancements, most of which can be accomplished within the client. Currently, these projects include:

- Extensions to the client to allow its use as an off-line web application, with features for preloading and caching data while on-line for later use offline
- A client optimized for smart phones and small tablets, with high-resolution displays and low-resolution touch input
- Data import and export plugins
- Interactive filter design plugin
- Data analysis plugins, such as a beat detector
- An alternative client, to function as a multichannel programmable analog signal generator for simulating realistic inputs to medical instruments

Details of these and other projects are available in LightWAVE's on-line help.

Acknowledgments

This work was supported by NIH/NIGMS grant R01-GM104987. Thanks to Ken Pierce, Diego Tognola, Tingting Zhu, and Roger Mark for contributions, feedback, and suggestions. Many other friends suggested improvements in WAVE that are reflected in the design of LightWAVE; thanks to Penny Ford-Carleton, Ted Clancy, Kevin Clark, Leon Glass, Scott Greenwald, Farzin Guilak, Jeff Hausdorff, Isaac Henry, Yuhei Ichimaru, David Israel, Franc Jager, Joe Mietus, Rama Mukkamala, Sheila Ryan, Kambiz Soroushian, Alessandro Taddei, and Andy Wieckiewicz.

References

[1] Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PC, Mark RG, Mietus JE, Moody GB, Peng CK, Stanley HE. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. Circulation 2000 (June 13);101(23):e215–e220. Circulation Electronic Pages: http://circ.ahajournals.org/cgi/content/full/101/23/e215 PMID:1085218; doi: 10.1161/01.CIR.101.23.e215.

[2] Moody GB, Mark RG, Goldberger AL. PhysioNet: Physiologic signals, time series, and related open source software for basic, clinical, and applied research. In Proc 33rd Ann Internat Conf IEEE EMBS. 2011; 8327–8330.

[3] Moody GB, Mark RG. The MIT-BIH Arrhythmia Database on CD-ROM and software for use with it. In Computers in Cardiology 1990. Los Alamitos: IEEE Computer Society Press, 1990; 185–188.

[4] Steil R. A C library to build CGI applications. https://github.com/rafaelsteil/libcgi/. Accessed 1-Sept-2013.

[5] Moody GB. WFDB Programmer's Guide. http://physionet.org/physiotools/wpg/, 2013. Accessed 1-Sept-2013.

[6] Moody GB, Dakin M, Mark RG. Web-enabled physiologic signal processing and analysis. In Proc World Cong on Med Phys and Biomed Eng. Sydney, Australia; 2003.

[7] Stenberg D. libcurl - the multiprotocol transfer library. http://curl.haxx.se/libcurl/, 2013. Accessed 1-Sept-2013.

[8] Crockford D. Introducing JSON. http://www.json.org/. Accessed 1-Sept-2013.

[9] Zalewski M. Browser security handbook, part 2. http://code.google.com/p/browsersec/wiki/Part2. Accessed 1-Sept-2013.

Address for correspondence:

George B Moody
MIT E25-505A, Cambridge, MA 02139 USA
george@mit.edu