EPJ Web of Conferences **245**, 08027 (2020) https://doi.org/10.1051/epjconf/202024508027
CHEP 2019

EVE-7 and FireworksWeb: The next generation event visualization tools for ROOT and CMS

Alja Mrak Tadel[1,*], *Matevz* Tadel[1,**], *Avi* Yagil[1], *Dmytro* Kovalskyi[2], and *Sergey* Linev[3] for the CMS collaboration and the ROOT project

[1]UC San Diego, La Jolla, CA, USA 92093
[2]MIT, Cambridge, MA, USA 02139
[3]GSI, Darmstadt, Germany 64291

Abstract. The CMS experiment supports and contributes to the development of the next-generation Event Visualization Environment (EVE) of the ROOT framework with the intention of superseding Fireworks, the physics analysis oriented event display of CMS, with a new server-web client implementation. EVE-7 is a rewrite of EVE for the ROOT-7 era, using modern C++ and relying on ROOT's built-in http server for communication with GUI clients. Part of EVE-7 is also implemented in JavaScript and uses OpenUI5, JSROOT, and Three.js as its foundation libraries. While some of the advanced features of EVE have not yet been ported to EVE-7, the existing code-base can be used for building of demonstrator applications serving as technology preview. FireworksWeb is currently at the stage of a minimal application built around EVE-7. Several advanced Fireworks features have been ported into EVE-7 in an experiment-independent manner, relying heavily on Cling, the C++ interpreter of ROOT: dynamic table views, handling of physics object collections, and filtering of objects within physics collections.

1 Introduction

During the last decade Event Visualization Environment (EVE) of ROOT [1, 2] was rather successful in satisfying the visualization needs of several high-energy physics (HEP) experiments, for example, ALICE, Belle2, CMS, HyperK, ILC, JUNO, NA-62, and T2K. Additionally, EVE has also been used by several smaller experiments in neutrino, nuclear, and medical physics. Development of the precursor to EVE started in 2005 for the needs of the ALICE experiment, guided by the performance considerations and object selection requirements needed to display simulated heavy-ion collision data. In 2007 it was split into the experiment independent part, which became part of ROOT (module EVE), and the ALICE specific part, AliEVE. ROOT OpenGL interface (RootGL)[3] was co-developed at the same time to support advanced EVE features and to provide performance optimizations required for visualization of LHC data.

In 2007 the CMS experiment chose EVE as the basis for their physics-analysis oriented event display, Fireworks [4, 5]. Prototype development was done in 2008/09 and included

*e-mail: amraktadel@ucsd.edu
**e-mail: mtadel@ucsd.edu

several contributions to core EVE visualization objects and algorithms. During the next two years, CMS invested into an intense, five developer effort that lead to support for operation within full CMSSW framework, implementation of geometry description browser, and implementation of pop-up detailed views for all reconstruction objects. The defining characteristic of Fireworks has always been its stringent requirement that all displayed data is a faithful representation of CMS Event Data Model objects and that the data should be accessed in the same way in the event-display as in an analysis algorithm. Since 2011 EVE, RootGL, and Fireworks are in maintenance mode, without practically any changes to their core elements.

This paper describes the ongoing projects EVE-7 and FireworksWeb aimed at modernization of visualization packages of ROOT and CMS. Section 2 presents the main motivations for this change and describes basic components and status of both projects. Section 3 reviews the plans for their further development.

2 EVE-7 and FireworksWeb

From the design and underlying technology perspective, EVE and RootGL are now 15 years old; ROOT Graphical User Interface (GUI) used by EVE is even older. Further, diversification of the form factor of personal computing devices and availability of production-grade user interface and 3D graphics libraries makes for a compelling argument for ROOT to abandon the old graphical interfaces. However, due to the limited developer resources for the ROOT project, it is clearly impossible to implement device specific user interfaces. In order to minimize the cost associated with the GUI engines, the choice has been made to adopt web-based technologies which are platform independent by its own nature. Additionally, this mode of operation seamlessly supports remote interactive graphics and visualization services.

EVE-7 and FireworksWeb are rewrites of EVE and Fireworks that are aiming to achieve the same goal for event visualization as well as to provide an opportunity to review and modernize the code with features available in modern `C++` [6]. Most of the functionality and features are expected to remain supported. However, with Cling, the new `C++` interpreter in ROOT, and `C++` support for lambda functions, it became possible to move several advanced features from Fireworks into the new EVE-7. The most important one is support for physics collections and physics objects that allows EVE-7 to interface to experiment-specific data-formats and provide backward navigation from graphical representations to physics objects.

Design of EVE-7 was also informed by the HEP Software Foundation Community White Paper on Visualization[7] and in particular by its guidelines for further development pertaining to the usage of community-supported data-formats, serving of geometry and event-data through services, and usage of client-server architecture for event-data visualization.

2.1 EVE-7 Components

EVE-7 architecture is using the server-client paradigm. Server is implemented in C++, as ROOT module `eve7`. Class `REveManager` is the entry point holding a list of top-level EVE directories or scenes, each holding a hierarchy of EVE objects. EVE objects support serialization into JSON format with additional binary data buffers used for transport of low-level rendering data. Object data is served through `RWebWindow` class and ROOT's built-in CivetWeb web server.

Existence of `C++` server is crucial for the main goal of EVE-7 and FireworksWeb: to visualize data as it is seen by analysis and reconstruction algorithms. User-provided `C++` expressions specifying filter expressions and table view contents get called on actual physics data objects, giving exactly the same results as would be obtained during physics processing.

The client side is implemented in JavaScript. JSRoot[8, 9] is used for interfacing to basic ROOT system, like color mapping, graphics attributes and basic 3D primitives. In the future, JSRoot will also provide tree-view widget, file dialogs, and geometry viewer. OpenUI5 [10], the standard Web-GUI for ROOT, is used for GUI components. Three.js[11] is used for 3D rendering.

Client commands and requests are sent to the server as C++ method calls targeting EVE objects. The final form of the method call is constructed on the server and gets executed via Cling.

Graphical views, table view configuration, selection, etc. are all implemented as EVE objects. This allows the same server-client protocol to be used for administrative operations as for the graphics and UI operations.

2.2 Server-client communication

Communication between server and client is bidirectional and stateful, using the WebSocket protocol. Multiple client connections are supported: this is required to be able to show different views in different browser tabs or windows but can also be used for connections from different client machines, by different users. Each client subscribes only to EVE scenes that are being shown in its window. Visual feedback for marking selected objects and objects under the mouse pointer (highlight) are synchronized across all clients and implemented in the client as an object outline or halo.

Physics object data is sent only when a new event is loaded. Detailed low-level visualization structures holding vertex, normal, and color data are only sent for objects that pass active filters. This can drastically reduce the volume of data that needs to be transferred during event transitions. Within an event, only objects that get changed as a result of user actions are streamed and updated. Geometry used for detector outlines or as visual containers for displaying detector hits are only transferred when they are first needed.

As an example, in the current implementation the payload sent to a client for an event with 1,000 tracks (3D + 2 projected views) is O(1 MB) spread over 6 messages.

2.3 Current status

At the moment, development is focused on the preparation of a FireworksWeb prototype with limited functionality. First production release is expected before the LHC Run 3[1] and will support at least the physics-analysis and event-scanning use case. Event-scanning refers to the practice of detailed investigation of some number of events belonging to certain category and ensuring that performance of trigger systems and reconstruction software is satisfactory. This is particularly important following any upgrades of detector, trigger, and DAQ systems.

EVE-7 components are developed as needed and currently EVE-7 is at the level of a technology demonstrator. An important part of the EVE-7 development is the tracking of its resource usage and performance. In particular, data representation and network transport protocol need to be optimized in a way that balances workloads on server and clients as well as provides reasonable response latency. Already supported features include:

- visual objects: point-sets, line-sets, tracks, ellipsoid, jets, and all `TGeo` shapes, including boolean / CSG shapes;

[1]LHC Run 3 is scheduled for 2022 – 2024 with a moderate increase in integrated luminosity per-year compared to Run 2. All LHC detectors are undergoing minor detector upgrades as well as changes to their trigger and DAQ systems to allow for higher data recording rates.

- support for interfacing to externally existing physics collections (e.g., vectors of reconstructed tracks or muons) and physics items (e.g., an individual track or muon in aforementioned collections);
- handling of scene changes (in response to user interaction) and scene destruction (typically during event transitions); and
- a selection and highlight mechanism, which works across browsers windows, graphical views and different visual representations.

A screenshot of EVE-7 browser window is shown in figure 1.

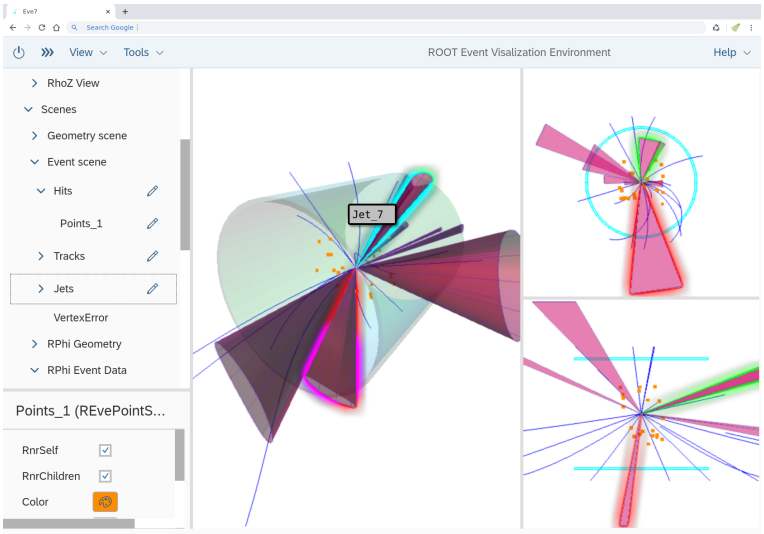

Figure 1. A screenshot of EVE-7 showing object highlight across graphical views.

FireworksWeb uses all existing EVE-7 functionality and already supports the following Fireworks concepts and features:

- a plug-in system for the selection of physics collections;
- physics collection editors (color, visibility, and physics item filter);
- algorithms for creation of visual representations for tracks, PF candidates, jets, MET, electrons, vertices, and muons;
- event navigation through CMS EDM data file;
- uses custom client GUI elements for the display of event information and for event navigation.

Figure 2 shows a screenshot of FireworksWeb and figure 3 shows a table view with CMS EDM track class. Most notable missing features are event filtering, visualization of calorimeter towers, and lego plot[12] showing the event contents as a marked-up $\eta - \varphi$ histogram.

3 Future work

The current version of EVE-7 is being released as part of ROOT 6.20 release in December 2019. At the same time, FireworksWeb technology preview has been releaseed for the CMS members.

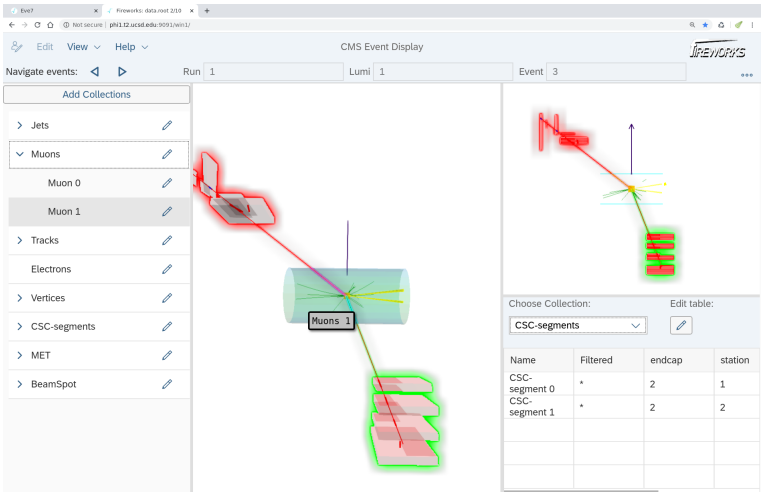

Figure 2. A screenshot of FireworksWeb showing eight physics collections.

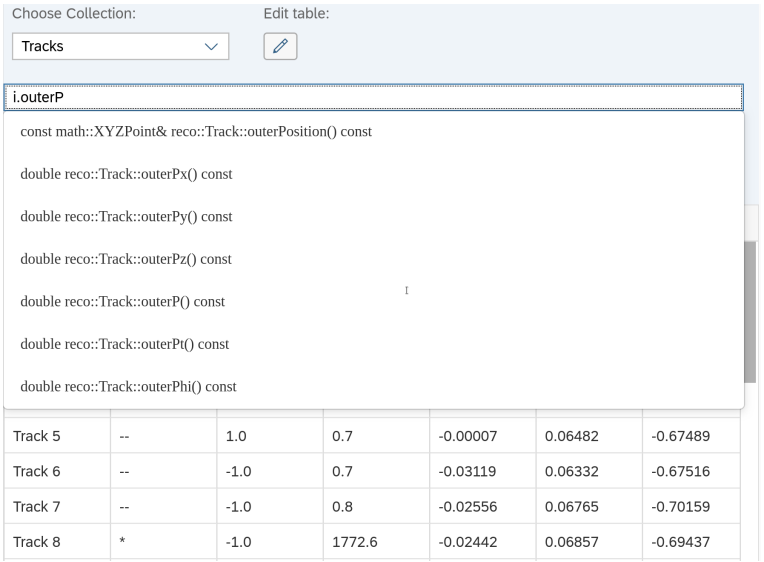

Figure 3. A screenshot of the FireworksWeb graphical user interface showing a table view of CMS data of class type reco::Track.

By the end of 2020 EVE-7 should be ready as replacement for EVE with most functionality ported over. In this time frame, FireworksWeb is expected to be functional for CMS Run3 data-analysis, event scanning, and trigger studies.

Beyond 2020 we expect EVE-7 to undergo further optimizations as well as to see increased user support load. FireworksWeb development will focus on advanced functionality, in particular on running from full CMSSW framework with full access to editing of CMS algorithm parameters. Optimizations for visualization of heavy ion LHC data will likely be required in both EVE-7 and FireworksWeb.

4 Conclusion

EVE-7 and FireworksWeb rewrites are now well underway. FireworksWeb is the driving force for the migration, forcing the initial focus on the most important core elements required for technology investigation as well as for convergence towards a working application in the time-frame of LHC Run3. The development of the ROOT Cling interpreter and the new features introduced by the latest C++ standards allowed for porting of high-level functionality from CMS code-base into ROOT. While the old EVE was actually a toolkit for visualizing HEP data, EVE-7 is striving to provide a framework for the building of comprehensive physics-analysis event display applications. The CMS collaboration is committed to support development and future maintenance of EVE-7 and FireworksWeb.

Acknowledgements

Work presented in this paper has been supported by the US National Science Foundation award #1624356.

References

[1] I. Antcheva *et al.*, *"ROOT: A C++ framework for petabyte data storage, statistical analysis and visualization,"* Comput. Phys. Commun. **180**, 2499 (2009).

[2] M. Tadel, *"EVE: Event visualization environment of the ROOT framework,"* PoS ACAT **08**, 103 (2008).

[3] M. Tadel, *"The new generation of OpenGL support in ROOT,"* J. Phys. Conf. Ser. **119** (2008) 042028.

[4] L. A. T. Bauerdick *et al.*, *"Event display for the visualization of CMS events,"* J. Phys. Conf. Ser. **331**, 072039 (2011).

[5] D. Kovalskyi *et al.*, *"Fireworks: A physics event display for CMS,"* J. Phys. Conf. Ser. **219**, 032014 (2010).

[6] A. M. Tadel, *et al.*, *"Exploring server/web-client event display for CMS,"* EPJ Web Conf. **214**, 05039 (2019).

[7] M. Bellis *et al.*, *"HEP Software Foundation Community White Paper Working Group — Visualization,"* arXiv:1811.10309v1 [physics.comp-ph] (2018).

[8] S. Linev, JSROOT [software], Project repository. https://github.com/root-project/jsroot

[9] B. Bellenot and S. Linev, *"JavaScript ROOT,"* J. Phys. Conf. Ser. **664**, no. 6, 062033 (2015).

[10] P. Muessing, OpenUI5 [software], Project webpage. https://openui5.org

[11] R. Cabello, Three.js [software], Project repository. https://github.com/mrdoob/three.js/

[12] J. D. Bjorken, *"Fractal phase space as a diagnostic tool for high-energy multijet processes,"* Phys. Rev. D **45**, 4077 (1992).