Firefox OS Web Apps for Science

Raniere Silva and Frédéric Wang

Mozilla MathML Project

June 10, 2014

Abstract

In this document, we describe recent work made by the Mozilla MathML team to help publishing scientific content using Web technologies. We focus on the Firefox OS platform currently being developed by Mozilla that provides a good framework to create mathematical user interfaces on mobile devices.

Contents

Introduction

The Web has become an integral part of our daily life. Some of the reasons for its success are its open nature and the way it enables anyone to get access to knowledge and to create his own projects. The Mozilla community has worked since the early days of the Web to guarantee openness, innovation and opportunity [Moz1].

The Web was created at the CERN to share knowledge between researchers [W3C1]. But although it was invented by scientists, we still have not seen the same positive impact on scientific practice. There are two main reasons that could explain this situation and they are actually related to each other.

The first one is a human problem. Researchers have kept teaching students to write papers for publication in journals and to avoid sharing their detailed results because of competition between academic groups. This means that most scientists ignore how to use tools to publish Web content and do not have the culture of openness and collaboration. Mozilla Science Lab was launched last year to remedy that problem and build educational resources, tools and prototypes for the research community [Moz2].

The second one is more technical. At the beginning of the Web, we had no tools to write scientific documents that could effectively replace the traditional authoring tools for publication on papers. Even if some tools are now available [1] they are still not used during the early years of undergraduate and graduate courses. In science, there is one extra technical problem: we still lack a cross-compatible way to publish mathematics on the Web despite the publication of the MathML standard in 1998 [W3C2]. The Mozilla MathML Project [Moz3] was launched in 1999 and in a few years, the team produced a good MathML implementation in Gecko together with tools to publish mathematics on the Web. MathML finally became part of HTML5 thanks to Mozilla's effort [Moz4], but other Web rendering engines still have limited support or even no support at all [Oreilly1]. This means that scientists either stay outside the Web (e.g. exchange only PDF documents) or rely on some workarounds (PNG images, CSS stylesheet, Javascript polyfills [2]) to publish mathematical content on the Web, with their inherent limitations and issues.

In recent years, the mobile market has grown considerably and more and more people are using mobile devices to access the Web. Mozilla has been working on Firefox OS, an open-source operating system for these mobile devices that relies exclusively on open standards and in particular Web technologies [Moz5]. Thanks to Gecko's good support for MathML and HTML5 in general as well as Mozilla's long experience with community involvement, we now have the opportunity to build a family of scientific Web applications compatible with Firefox OS devices. Some of the early prototypes created by the Mozilla MathML team are presented in this paper.

In a first part, we will review the Web platforms and focus on how the technologies can be used for science. We will present the classical features as well as more recent improvements that have been integrated into Mozilla projects recently such as the Open Type MATH table, WebGL, Web Components or TeXZilla. Some of this work has been made during the crowdfunding project "Mathematics in ebooks" which has also resulted in the creation of a collection of scientific documents using advanced Web technologies [Wang1].

In a second part, we will study how to use these technologies to write Firefox OS Web apps for science. We will give an overview of the general format, which is essentially just an archive of Web pages together with a manifest. We will discuss the big picture for a Math Suite and present three Web apps we started to develop: a math cheat sheet, a note-taking app for math and finally an app to help doing algebraic calculations similar to Epsilonwriter at MathUI'13 [Nicaud2].

The authors have written this paper using collaboration tools like GitHub and all the sources, programs and tools presented here are open. Because PDF format has been requested for submission to the MathUI workshop, we had to provide our demos separately instead of integrating them directly into the document. We try to provide some screenshots in appendix but they will not replace real testing and we strongly invite the reader to test the demos in a Gecko browser. A Web version of that document is also available for online reading.

[1] For example, the authors used git for version control and LaTeXML to produce HTML pages from LaTeX

[2] This is a term used by Web developers to describe code providing a technology that is expected to be supported natively by browsers

1 The Web Platform

1.1 Overview

All the technologies presented in this section are based on Web standards and should be supported by any Web rendering engine. We will particularly be interested in Gecko which has the best native MathML support and is the core of Firefox OS. We will present some of the improvements that have been made by the Mozilla MathML team.

All these technologies should be usable in HTML documents. This obviously includes Web pages but also EPUB ebooks, HTML mails, browser add-ons or Firefox OS Web apps. For example, it is possible to receive and send emails with mathematical equations using Thunderbird or Seamonkey's mail client [Wang2]. In this paper, we will mainly focus on Firefox OS Web apps but one should keep in mind that all the features apply in other contexts too.

1.2 Basic HTML5 Features

The main language is HTML5, which allows to create pages with headers, paragraphs, tables, hyperlinks, etc. The well-known CSS language is used to apply specific style to HTML5 and is powerful enough to produce advanced designs. Finally, DOM/Javascript provides a programming language and enables interactive documents and complex user interfaces. New HTML5 elements gives other possibilities. For example the document `pendulum-20131125` of [Wang1] uses the the `<video>` tag to insert some sequences of a physics lecture.

For scientific documents, we need two other features: creating graphs, schemas, diagrams, etc. and writing mathematical formulas. For the former, simple PNG images might be enough. However, Web rendering engines also support the SVG language to let authors write scalable images using some simple drawing primitives. Many programs are available to generate scientific schemas in SVG formats. Mathematical equations can be viewed as an extension of text layout and thus requires a good integration within HTML as done by Gecko's native MathML. The document demos/1-mathml-in-html.html shows how various font properties apply to MathML text via CSS, the good alignment of inline equations and its scalability.

One of the nice feature introduced some years ago in Gecko is the possibility to integrate MathML equations inside SVG images via the `<foreignObject>` element. People can then create scientific schemas with mathematical equations. We will also use this property in section 1.5 when we introduce `<canvas>`. See demos/2-mathml-in-svg.svg for an example of a SVG schema with MathML equations inside. Again, some authoring tools exist to help generating such schemas such as LaTeXML [LaTeXML].

demos/3-mathml-javascript.html is a small example of an interactive MathML formula. Javascript is used to allow the user to highlight parts of a 3-dimensional determinant and understand each term of the Sarrus' rule. Note that no particular Javascript API is needed, one only modifies the MathML tree via the standard DOM interface and the rendering is automatically updated. This example is taken from [Wang1] which contains many other examples of this type.

To conclude this review, we briefly mention classical Web features like Unicode characters, advanced typographic features (e.g. kerning, ligatures), right-to-left directionality, copy and paste, automatic line breaking and accessibility to users with disabilities. In general, all these features are well handled by Web languages and rendering engines. The two first works well in Gecko's MathML too and right-to-left MathML was also implemented four years ago [Bugzilla534963]. For the fourth one, we have proposed a MathML Copy add-on for Firefox as a temporary solution to [Bugzilla539506]. MathML line breaking [Bugzilla534962] is very important for small screens but to our knowledge, no CSS-compatible implementations exist. Finally, Mozilla's intern Jonathan Wei has started some work on MathML accessibility in Gecko [Bugzilla916419] [Moz7]. We expect to see improvements to these three last points in order to get a complete Web platform for mathematical user interfaces.

1.3 Styling of Mathematics

As seen in section 1.2, one can use CSS the standard way to apply specific styles to mathematical equations. However, perhaps the most important style that scientists and publishers want is the rendering with a given mathematical font. Gecko has always used TeX heuristics to position scripts, fractions, roots, etc. However, there are many parameters that depend on
fontdimen parameters and others on built-in rules in TeX's typesetting engines, which are not exposed to a HTML5 rendering engine *a priori* [Vieth]. Moreover, in order to draw stretchy and large operators the MathML code requires to pick and assembly specific glyphs from math fonts. So far, Gecko only knows a limited set of Unicode constructions and also has a few private per-font tables. This means that without one of the supported fonts installed, the quality of the MathML rendering could be very low.

In order to solve that problem, we have started to implement the OpenType MATH table which is currently undergoing standardization as ISO/IEC CD 14496-22 "Open Font Format" [MPEG1]. We have started to use this table in order to provide a generic support for MATH fonts and a partial support is available in Firefox 31. Note that MathML does not specify a precise way to render mathematical formulas and leave the details to the implementers. Relying on the MATH table will give more control to professional font designers. In a nutshell, the main features are:

1. Some OpenType Tags to provide alternate form of glyphs. For example, we implemented the "ssty" feature to adjust the size of glyphs like primes when they are used as scripts.

2. Font parameters to define precisely the gaps, shifts, kerning, etc. of mathematical objects. Most of them are extensions to the TeX rules so it is easy to integrate them into our MathML rendering engine.

3. Font parameters specific to each glyph. At the moment, we only considered italic corrections.

4. A table to draw stretchy and large operators such as parenthesis, radicals or summation symbols. We have started to use that table for our operator stretching code.

There are various mathematical fonts with an OpenType MATH table available and most of them are distributed with a TeX Live distribution. Windows and Mac systems have respectively Cambria Math and STIX installed by default. We have also experimented with PackageKit auto-installation on Linux [PackageKit], we plan to install math fonts on Firefox OS and tried other another auto-installation method for Android. In any case, fallback downloadable Web fonts can also be used. This means that missing mathematical fonts will become very unlikely in the future.

The page demos/4-mathml-fonts.html shows how to get various font style for a document: Asana, TeX Gyre Bonum, TeX Gyre Pagella, TeX Gyre Termes, Latin Modern, Neo Euler and XITS. You need a browser with Gecko 31 or higher in order to see the appropriate rendering.

1.4 TeXZilla

Mathematical formulas are complex and hence writing mathematics is difficult. Some tools like WYSIWYG editors or handwriting recognition might help. One input method easy to implement is the conversion from a simple syntax like LaTeX into MathML. Even if there are tons of such converters, only a few of them are implemented in Javascript. Moreover the remaining ones try to tweak the output to workaround browser limitations, are not standalone Javascript module usable in add-ons and do not work with Unicode characters or right-to-left mathematics. Hence we decided to write yet another converters to focus on Mozilla's needs.

We thus wrote TeXZilla [Wang3], a LaTeX-to-MathML converter generated with the help of Jison. It relies on a LALR(1) grammar and on the `unicode.xml` file of [W3C3], which contains an extensive list of Unicode characters used in science together with some semantic property and LaTeX commands to produce them. It accepts arbitrary Unicode input as well as right-to-left mathematics, something that is important for the world-wide aspect of the Mozilla community and more specifically Arabic mathematics.

The public API contains a `toMathML` function to convert a LaTeX string into a MathML DOM but also a `toMathMLString` function when the DOM is not available (e.g. in nodejs). There is also a convenient

`getTeXSource` function to extract the LaTeX source from the MathML output and a `toImage` function to embed the MathML output inside a SVG image (see section 1.5).

TeXZilla can work in any CommonJS program [CommonJS], in Web pages, as a command line program or as a Web server. We have submitted a Firefox add-on that has been approved by the reviewers. We have also already integrated it into various other Mozilla tools: in CKEditor for the Mozilla Developer Network, in the Seamonkey/Thunderbird composers or in a Firefox OS web app (see section 2.4). Finally, a prototype `<x-tex>` custom element is also available (see section 1.5). We expect it will become an important library for future scientific applications.

1.5 Advanced HTML5 Features

The `<canvas>` element has been introduced in HTML5 to draw graphics via JavaScript. This element may be more convenient than SVG in situations where you want to generate schemas dynamically. For example demos/5-canvas.html shows the typical graphs associated to a RLC circuit that are automatically updated when the user changes the frequency of the current. It is also possible to use WebGL [WebGL], opening the possibility to draw 3D scientific schemas. For example chemdoodle relies on WebGL to provide 3D representation of chemical structures.

Similarly to SVG, 2D/3D scientific schemas created via the `<canvas>` might require mathematical formulas. This is possible since we saw in section 1.2 that MathML can be inserted in SVG via a `<foreignObject>` and moreover SVG images can be inserted in both the `CanvasRenderingContext2D` and `WebGLRenderingContext`. One can for example use TeXZilla's `toImage` function to generate such SVG images dynamically. For instance, the page demos/6-mathml-in-webgl.html shows an example of a 3D schemas for the magnetic field of a circular current loop. Again, see [Wang1] for more examples.

One of the recent HTML5 features that is currently being implemented in Web rendering engines is Web Components [WebComponents]. The idea is to allow Web developers to create their own custom HTML elements. These elements are made of the so-called shadow tree, which is an "internal" representation from basic elements and CSS style to which we attach some behaviors with DOM events and Javascript. This is typically useful to create some reusable widgets. Mozilla's Brick project is a collection of such elements and contains many demos [Moz6]. Web Components are currently not fully implemented natively in browsers but some polyfill libraries like X-Tag allows to emulate their behavior [Moz6].

Web components can obviously be very helpful to design user inteface for sciencific Web app. For example, the specific interactive drawing of demos/5-canvas.html for RLC circuits could become a generic widget `<x-graph>` to draw scientific graphs with some user interface to configure the curve parameters. As a proof-of-concept, we have used the X-Tag library to create a custom `<x-tex>` tag. The behavior is simply to automatically converts its LaTeX content into MathML using TeXZilla, see demos/7-web-component.html. The Mozilla MathML team is following the developments in Web components and plans to create more custom elements for sciencific web apps, so that Web developers can reuse them.

2 Firefox OS

2.1 Overview

As said before, Firefox OS is the open-source operating system targeting mobile devices that is being developed by Mozilla and based on Web technologies [Moz5]. All the apps in Firefox OS are Web pages, provided by a server or stored on the device, built using only Web technologies and, except when some special Javascript API are required, can also run in a Web browser.

To create an app from a previous Web page you only need a manifest file that stores some metadata about the app and an image to use as icon. The files demos/app/manifest.webapp and demos/app/icon.png, respectively a manifest and an icon, show these extra files needed to create an app for demos/app/index.html, a very basic static HTML page.

Although the Web was invented to share knowledge, today it is also used as a platform to create knowledge (e.g. Wikipedia) and in a near future significant part of the content created on the Web will be done using mobile devices. For science there are some Web authoring tools for publication on papers focus on desktop users but only a small number deals with math and none are optimized for mobile devices.

This lack of tools to deal with math can be addressed by a "Math Suite" that we describe in the next section followed by some of the demos written by the Mozilla MathML community.

2.2 Math Suite

According to Wikipedia, "an office suite [...] is a collection of productivity programs intended to be used by knowledge workers". Similarly, we would like to develop a "Math Suite", that is a collection of software tools intended to be used by someone who makes intensive use of math.

The basic components of a Math Suite are a text processor with math support and a symbolic math programming language, like Mathematica, but the suite could also have:

- copy & paste;
- search engines for math expressions;
- a WYSIWYG math editor;
- markup converters (e.g. LaTeX to MathML and MathML to LaTeX);
- handwriting recognition;
- a 2D drawing tool that helps with, for example, demos/2-mathml-in-svg.svg;
- a 3D drawing tool that helps with, for example, demos/6-mathml-in-webgl.html;
- a scripting tool that helps with, for example, demos/3-mathml-javascript.html;
- accessibility to users with disabilities;
- a virtual keyboard for math
- and more.

Many of the applications, whether or not they are part of the Math Suite, share a set of functions. For example the search engine can be used by a text processor and by an ebook reader. Having these functions available in an open source Javascript library will make easy to develop new Web applications that can be customized into standalone applications to run on desktop and mobile devices. Figure 1 illustrates some of the applications that can compose the Math Suite, the features made available by the Javascript Library and which feature is used by which application.

The authors of this work started writing this Javascript library as small independent modules together with some prototype demos that use them. The prototypes described below were developed for Firefox

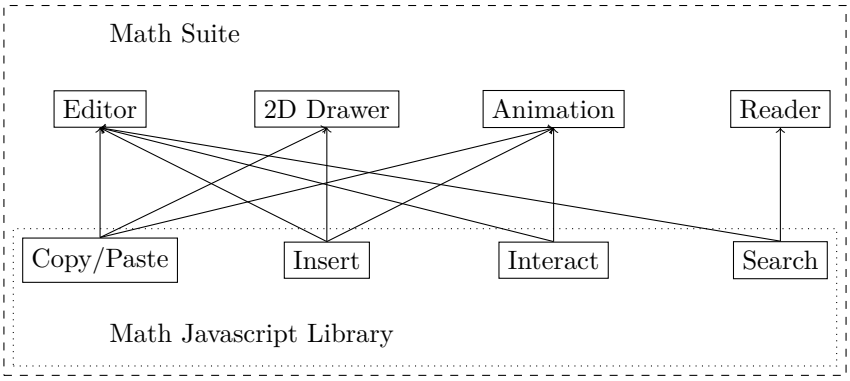


Figure 1: Illustration of Math Suite.

OS but also work with any browser that supports HTML5 with MathML and are available on the Firefox Marketplace.

One of the authors will also implement a virtual keyboard for Firefox OS as part of the 2014 edition of Google Summer of Code [Bugzilla1004057]. So far the most used method to input math on the Web is LaTeX, e.g. Wikipedia and phpBB. Unfortunately typing LaTeX code on a mobile device is hard because to access some useful characters, e.g. ^, _ and \, the user needs to switch the keyboard. This virtual keyboard project is intended to help writing such mathematical expressions faster.

2.3 Math Cheat Sheet

This is a basic app with a collection of common K-12 equations available at http://r-gaia-cs.github.io/math-cheat-sheet/. The equations are organized in sections and a table of contents is provided to help jumping from one section to another.

Although it is a basic app it is a nice way to show that Firefox OS supports MathML and create demand for some features not available right now like copy and paste (only the unstable version of Firefox OS supports copy and paste and the desktop version of Firefox requires the MathML Copy add-on), automatic line breaking and accessibility to users with disabilities.

The math authoring tool used in this app was TeXZilla, see section 1.4, because it allows the conversion of LaTeX into MathML and keeps the original LaTeX source code available. This is useful, for example, to copy an equation and paste it into Wikipedia.

Other nice features that can be added to this app in the future include:

- link to resource like Wikipedia related to the equation,
- search bar to quickly find the desired equation,
- customization of equations, ...

2.4 TeXZilla App

This is a note-taking app for math available at http://r-gaia-cs.github.io/TeXZilla-webapp/. It takes a LaTeX expression as input, shows a preview of it refreshed during input and enables the user to save it.

The conversion from LaTeX to MathML is performed by TeXZilla (see section 1.4) and no internet connection is required. The save functionality uses IndexedDB, "an API for client-side storage of significant amounts of structured data and for high performance searches on this data using indexes" [W3C4].

This app also has some experimental buttons to help the user to enter LaTeX commands like `\frac` and `\sqrt`. These buttons will be removed once the math virtual keyboard mentioned at the end of section 2.2 is available.

Right now, it is only possible to add math expressions but nice features that can be added to this app in the future include adding text around the math expressions and having more than one "piece of paper" for notes.

2.5 DynAlgebra

This is a Dynamic Algebra [Nicaud1], "algebraic calculations on a computer by direct manipulation", app available at http://r-gaia-cs.github.io/DynAlgebra/. The idea of Dynamic Algebra is about twenty years old and could help students to learn formula structure and transformations and help every users to produce step by step calculations. However, most of the previous implementations have not been successful.

The last known implementation of Dynamic Algebra was done in EpsilonWriter [Nicaud2] as a Java application and covers:

internal drag & drop like converting $x + y$ into $y + x$,

external drag & drop like dropping a formula on another formula,

on click calculation like converting $x + x$ into $2x$ by click on + and

rewriting rules like converting x^{-n} into $\frac{1}{x^n}$.

Our app, written in Javascript, is still at a prototype level compared to EpsilonWriter since it only partially covers the "on click calculation" and does not have a WYSIWYG input interface (currently TeXZilla is used). However, it uses MathML as storage format, which makes possible to work in documents not intended to be used for Dynamic Algebra.

Conclusion

In this paper, we have reviewed the HTML5 features and how they could be used to write mathematical user interfaces. In particular, good MathML support is essential to write mathematics in Web apps in a way that is compatible with the classical HTML, CSS, Javascript/DOM and SVG features. We have also presented some of the improvements to mathematical styling in Gecko based on OpenType MATH fonts. We then gave an overview of TeXZilla, a standalone LaTeX-to-MathML converter compatible with Unicode that can easily be integrated in Web apps. Finally, we studied advanced HTML5 features like canvas, WebGL and Web Components and mentioned how they could be used to create 3D-schemas, animations or complex user interfaces. These are suitable to share knowledge in science and help explaining complex concepts more easily, in an interactive way.

In a second part, we explained how these HTML5 features can be used to create Firefox OS Web apps and we provided concrete use cases like a note-taking math app, a math cheat sheet or an interactive app for algebraic manipulations. The idea is to rely on Firefox OS to build a platform for science, not only to consult knowledge but also to generate new knowledge as shown in some of the demos presented in this paper. Although, these apps are still prototypes we expect we will improve them in the future and create a real math suite for mobile platforms. We expect that new contributors could get involved and that the original ideas shared at the MathUI workshop could in turn become real Firefox OS apps.

References

[Bugzilla534962] Improve MathML linebreaking
Mozilla Bug 534962
https://bugzilla.mozilla.org/show_bug.cgi?id=534962

[Bugzilla534963] Mechanisms for controlling the Directionality of layout
Mozilla Bug 534963
https://bugzilla.mozilla.org/show_bug.cgi?id=534963

[Bugzilla539506] Implement clipboard specification
Mozilla Bug 539506
https://bugzilla.mozilla.org/show_bug.cgi?id=539506

[Bugzilla916419] Expose MathML as a hierarchical accessible tree
Mozilla Bug 916419
https://bugzilla.mozilla.org/show_bug.cgi?id=916419

[Bugzilla1004057] Math virtual keyboard
Mozilla Bug 1004057
https://bugzilla.mozilla.org/show_bug.cgi?id=1004057

[CommonJS] CommonJS, "a group with a goal of building up the JavaScript ecosystem for web servers, desktop and command line apps and in the browser".
http://wiki.commonjs.org/wiki/CommonJS

[Moz1] Mozilla's Mission
https://www.mozilla.org/en-US/mission/

[Moz2] Mozilla Science Lab
http://mozillascience.org/

[Moz3] Mozilla MathML Project
https://developer.mozilla.org/en-US/docs/Mozilla/MathML_Project

[Moz4] MathML-in-HTML5
mozilla.dev.tech.mathml
https://groups.google.com/forum/#!topic/mozilla.dev.tech.mathml/eW-tA_3U_Fk

[Moz5] Firefox OS
https://www.mozilla.org/en-US/firefox/os/

[Moz6] Brick
http://mozilla.github.io/brick/

[Moz6] X-Tag
http://x-tags.org/

[Moz7] MathML Accessibility (video on Air Mozilla)
Jonathan Wei
https://air.mozilla.org/mathml-accessability/

[MPEG1] Text of ISO/IEC CD 14496-22 3rd edition "Open Font Format",
6.3.6 MATH – The mathematical typesetting table
http://mpeg.chiariglione.org/standards/mpeg-4/open-font-format/text-isoiec-cd-14496-22-3rd-edition

[Vieth] OpenType math illuminated
Ulrik Vieth
TUGboat, Vol. 30, No. 1. (2009), pp. 22-31
http://www.tug.org/TUGboat/tb30-1/tb94vieth.pdf

[Nicaud1] Algèbre dynamique, glisser - déposer par équivalence,
Jean-François Nicaud and Christian Mercat
Actes des Journées mathématiques de l'Institut Français de l'Éducation (ENS de Lyon).

[Nicaud2] Implementation of Dynamic Algebra in Epsilonwriter,
Jean-François Nicaud and Christophe Viudez
MathUI 2013 - 8th Workshop on Mathematical User Interfaces

[Oreilly1] MathML forges on, O'Reilly blog
Peter Krautzberger
http://programming.oreilly.com/2013/11/mathml-forges-on.html

[PackageKit] What is PackageKit?
http://www.freedesktop.org/software/PackageKit/pk-intro.html

[W3C1] WorldWideWeb: Proposal for a HyperText Project
Tim Berners-Lee, Robert Cailliau
http://www.w3.org/Proposal.html

[W3C2] W3C Math Home
http://www.w3.org/Math/

[W3C3] XML Entity Definitions for Characters
David Carlisle and Patrick Ion
http://www.w3.org/TR/xml-entity-names/

[W3C4] Indexed Database API
Nikunj Mehta, Jonas Sicking, Eliot Graff, Andrei Popescu, Jeremy Orlow and Joshua Bell
http://www.w3.org/TR/IndexedDB/

[Wang1] Mathematics in ebooks
Frédéric Wang
http://www.ulule.com/mathematics-ebooks/

[Wang2] Writing mathematics in emails
Frédéric Wang
http://www.maths-informatique-jeux.com/blog/frederic/?post/2012/11/14/Writing-mathematics-in-emails

[Wang3] TeXZilla 0.9.4 Released
Frédéric Wang
http://www.maths-informatique-jeux.com/blog/frederic/?post/2014/02/25/TeXZilla-0.9.4-Released

[WebComponents] Introduction to Web Components,
Dominic Cooney and Dimitri Glazkov
http://www.w3.org/TR/components-intro/

[WebGL] WebGL Specification
Dean Jackson
https://www.khronos.org/registry/webgl/specs/latest/1.0/

[LaTeXML] LaTeXML 0.8.0 released, converts tikz/pgf to SVG
http://lists.jacobs-university.de/pipermail/project-latexml/2014-May/001773.html

A The Web Platform

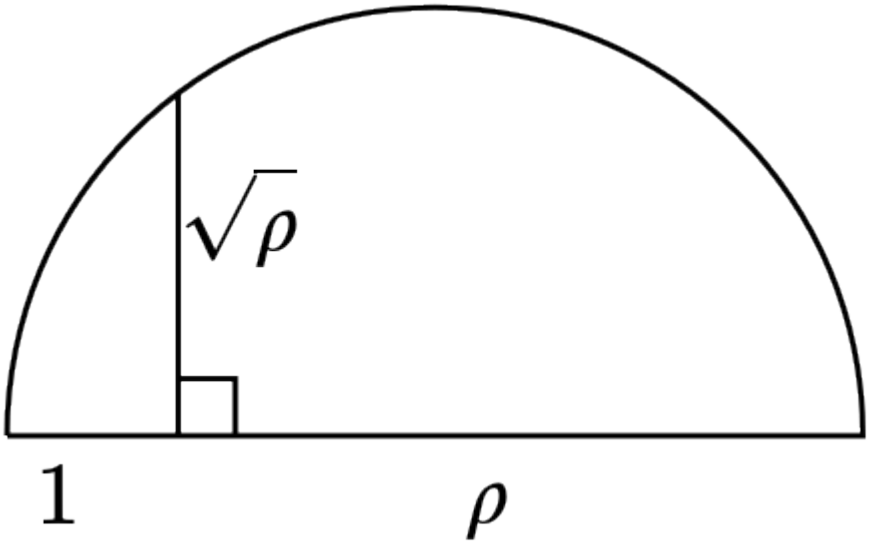
A.1 1-mathml-in-html

The demo shows that MathML integrates well inside HTML page. It can easily be styled like the text (`font-family: Neo Euler, font-size: 64pt, color: white`), scales according to the font-size and has good vertical alignment with respect to the surrounding text.

PNG	blah $\frac{x+y^2}{k+1}$ blah
MathML	blah $\frac{x+y^2}{k+1}$ blah

A.2 2-mathml-in-svg

The demo is an SVG image with MathML formulas embedded via the `<foreignObject>` element.



A.3 3-mathml-javascript

The demo shows how Javascript can be used to produce interactive documents. Clicking on the button will successively highlight different terms of Sarrus' rule. Below is a screenshot of the three first steps.

$$\det\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{31}a_{22}a_{13} - a_{32}a_{23}a_{11} - a_{33}a_{21}a_{12}$$

$$\det\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{31}a_{22}a_{13} - a_{32}a_{23}a_{11} - a_{33}a_{21}a_{12}$$

$$\det\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{31}a_{22}a_{13} - a_{32}a_{23}a_{11} - a_{33}a_{21}a_{12}$$

A.4 4-mathml-fonts

Below are screenshots of the demo page with different fonts in Firefox 31 for Linux. This is simply achieved by using the classical CSS rule `font-family` with optional `@font-face` rules to define a fallback with Web fonts. Gecko 31 is required to get the different fonts used for the drawing of operators (e.g. the integral and sum symbols). Other features of the OpenType MATH table (e.g. gaps or linethickness) are not implemented yet in this version.

Asana

We note that $x \mapsto \Im\left(f(x)\right)$ is the integrand above. We also note that the poles of f are $z^{\pm} = -1 \pm 2i$. By choosing the right contour, one can show that

$$\int_{-\infty}^{+\infty} f(x) dx = -2i\pi \sum_{\substack{z \text{ residue} \\ \Im(z) < 0}} \operatorname{Res}(f, z)$$

and thus

$$\int_{-\infty}^{+\infty} \frac{16 \sin\left(2x + \pi\right)}{5\left(x^2 + 2x + 5\right)^2} dx = \frac{\pi \sin\left(2\right)}{e^4}$$

Latin Modern

We note that $x \mapsto \Im\left(f(x)\right)$ is the integrand above. We also note that the poles of f are $z^{\pm} = -1 \pm 2i$. By choosing the right contour, one can show that

$$\int_{-\infty}^{+\infty} f(x) dx = -2i\pi \sum_{\substack{z \text{ residue} \\ \Im(z) < 0}} \operatorname{Res}(f, z)$$

and thus

$$\int_{-\infty}^{+\infty} \frac{16 \sin\left(2x + \pi\right)}{5(x^2 + 2x + 5)^2} dx = \frac{\pi \sin\left(2\right)}{e^4}$$

Neo Euler

We note that $x \mapsto \Im\left(f(x)\right)$ is the integrand above. We also note that the poles of f are $z^{\pm} = -1 \pm 2i$. By choosing the right contour, one can show that

$$\int_{-\infty}^{+\infty} f(x)dx = -2i\pi \sum_{\substack{z\,\mathrm{residue} \\ \Im(z)<0}} \mathrm{Res}\,(f,z)$$

and thus

$$\int_{-\infty}^{+\infty} \frac{16\sin\left(2x+\pi\right)}{5\left(x^2+2x+5\right)^2}dx = \frac{\pi\sin\left(2\right)}{e^4}$$

TeX Gyre Bonum

We note that $x \mapsto \Im\left(f(x)\right)$ is the integrand above. We also note that the poles of f are $z^{\pm} = -1 \pm 2i$. By choosing the right contour, one can show that

$$\int_{-\infty}^{+\infty} f(x)dx = -2i\pi \sum_{\substack{z\mathrm{residue} \\ \Im(z)<0}} \mathrm{Res}\,(f,z)$$

and thus

$$\int_{-\infty}^{+\infty} \frac{16\sin\left(2x+\pi\right)}{5\left(x^2+2x+5\right)^2}dx = \frac{\pi\sin\left(2\right)}{e^4}$$

TeX Gyre Pagella

We note that $x \mapsto \Im\left(f(x)\right)$ is the integrand above. We also note that the poles of f are $z^{\pm} = -1 \pm 2i$. By choosing the right contour, one can show that

$$\int_{-\infty}^{+\infty} f(x)dx = -2i\pi \sum_{\substack{z\,\mathrm{residue} \\ \Im(z)<0}} \mathrm{Res}\,(f,z)$$

and thus

$$\int_{-\infty}^{+\infty} \frac{16\sin\left(2x+\pi\right)}{5\left(x^2+2x+5\right)^2}dx = \frac{\pi\sin\left(2\right)}{e^4}$$

TeX Gyre Termes

We note that $x \mapsto \Im\left(f(x)\right)$ is the integrand above. We also note that the poles of f are $z^{\pm} = -1 \pm 2i$. By choosing the right contour, one can show that

$$\int_{-\infty}^{+\infty} f(x)dx = -2i\pi \sum_{\substack{z\,\mathrm{residue} \\ \Im(z)<0}} \mathrm{Res}\,(f,z)$$

and thus

$$\int_{-\infty}^{+\infty} \frac{16\sin\left(2x+\pi\right)}{5\left(x^2+2x+5\right)^2}dx = \frac{\pi\sin\left(2\right)}{e^4}$$

XITS

We note that $x \mapsto \Im\left(f(x)\right)$ is the integrand above. We also note that the poles of f are $z^{\pm} = -1 \pm 2i$. By choosing the right contour, one can show that

$$\int_{-\infty}^{+\infty} f(x)dx = -2i\pi \sum_{\substack{z \text{ residue} \\ \Im(z)<0}} \text{Res}(f, z)$$

and thus

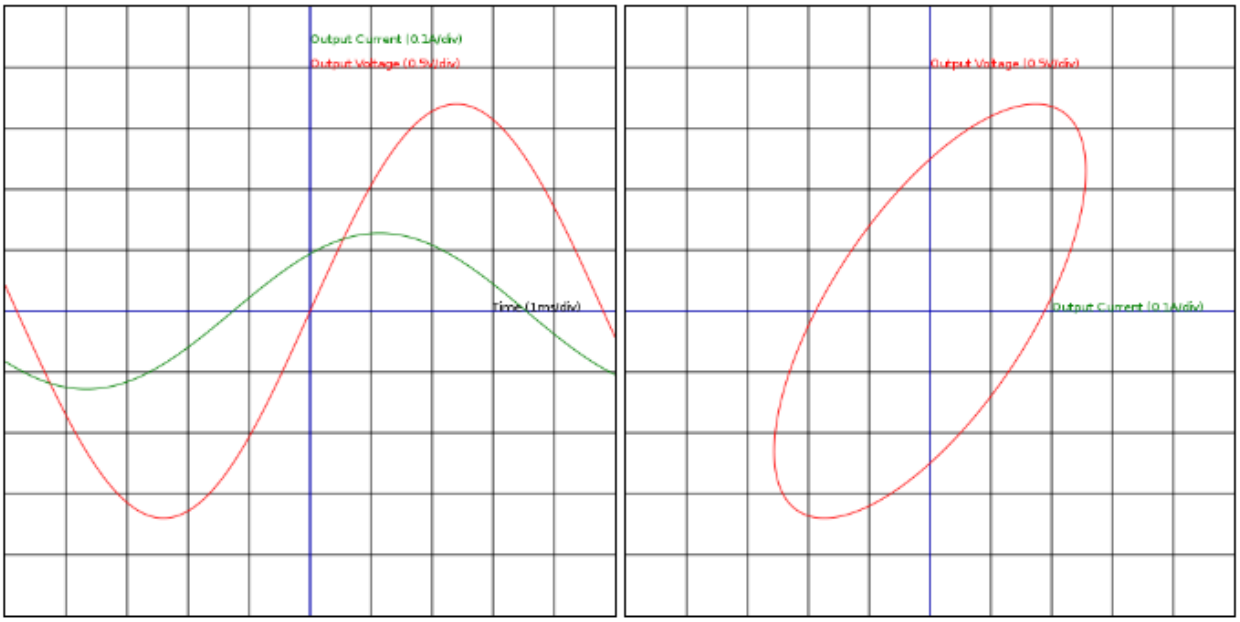
$$\int_{-\infty}^{+\infty} \frac{16\sin\left(2x+\pi\right)}{5\left(x^2+2x+5\right)^2} dx = \frac{\pi\sin(2)}{e^4}$$

A.5 5-canvas

The graphs in this demo are drawn using canvas. They are updated automatically when you modify the current frequency.

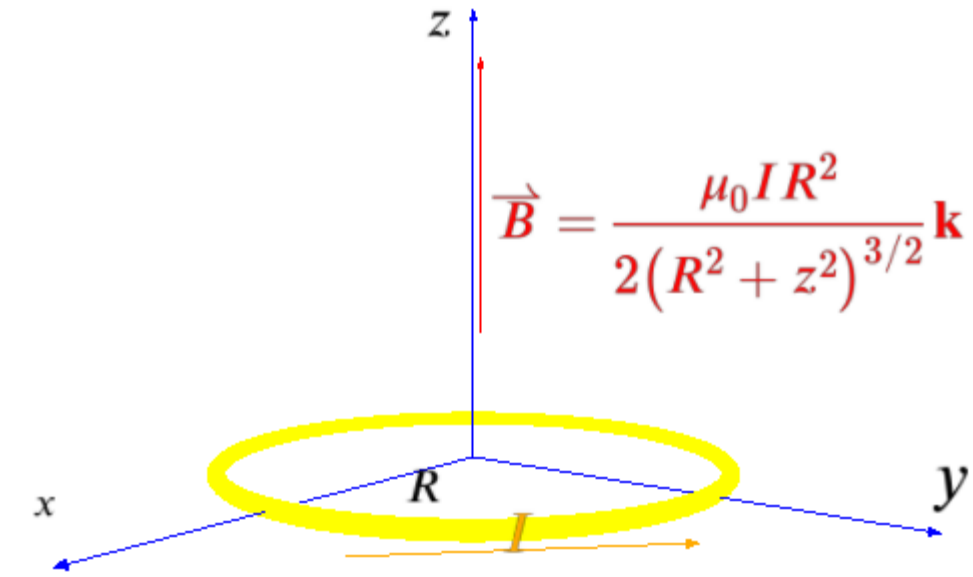
Canvas

f: 104 Hz ; V_0: 1.7 V ; I_0: 0.12812649919130903 A ; V_0/I_0: 13.268137432379897 Ω ; $\frac{V_0 I_0}{2}$: 0.1089

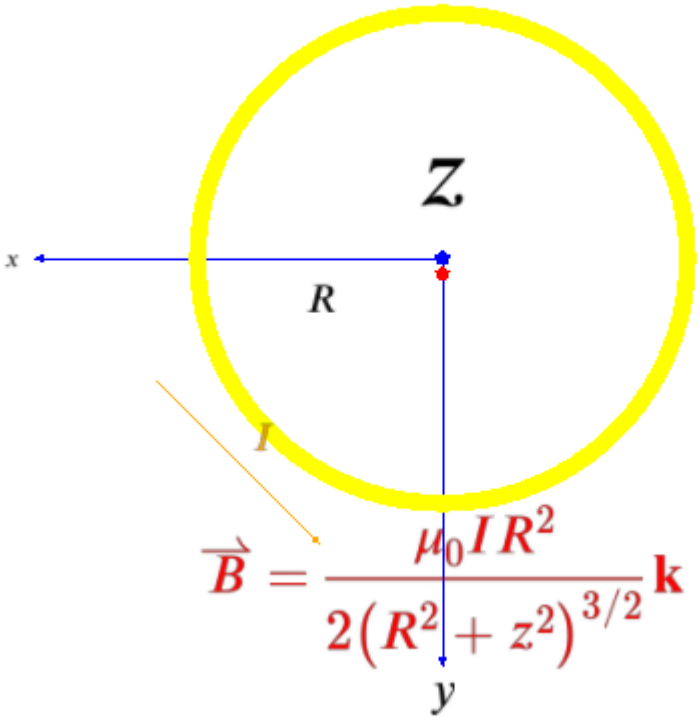


A.6 6-mathml-in-webgl

The 3D schema in this demo is drawn using WebGL and has MathML formulas included inside.



You can interactively move and rotate the schema using the mouse. Here is the same schema seen from above, which more clearly shows the circular loop and its radius.



A.7 7-web-component

The page contains the following code:

```
<x-tex display="block">\Gamma(t) =
```

```
  \lim_{n \to \infty} \frac{n! \; n^t}{t \; (t+1)\cdots(t+n)} =
  \frac{1}{t} \prod_{n=1}^\infty \frac{\left(1+\frac{1}{n}\right)^t}{1+\frac{t}{n}} =
 \frac{e^{-\gamma t}}{t} \prod_{n=1}^\infty \left(1 + \frac{t}{n}\right)^{-1} e^{\frac{t}{n}}
</x-tex>
```

which is automatically converted into MathML and renders approximately like this:

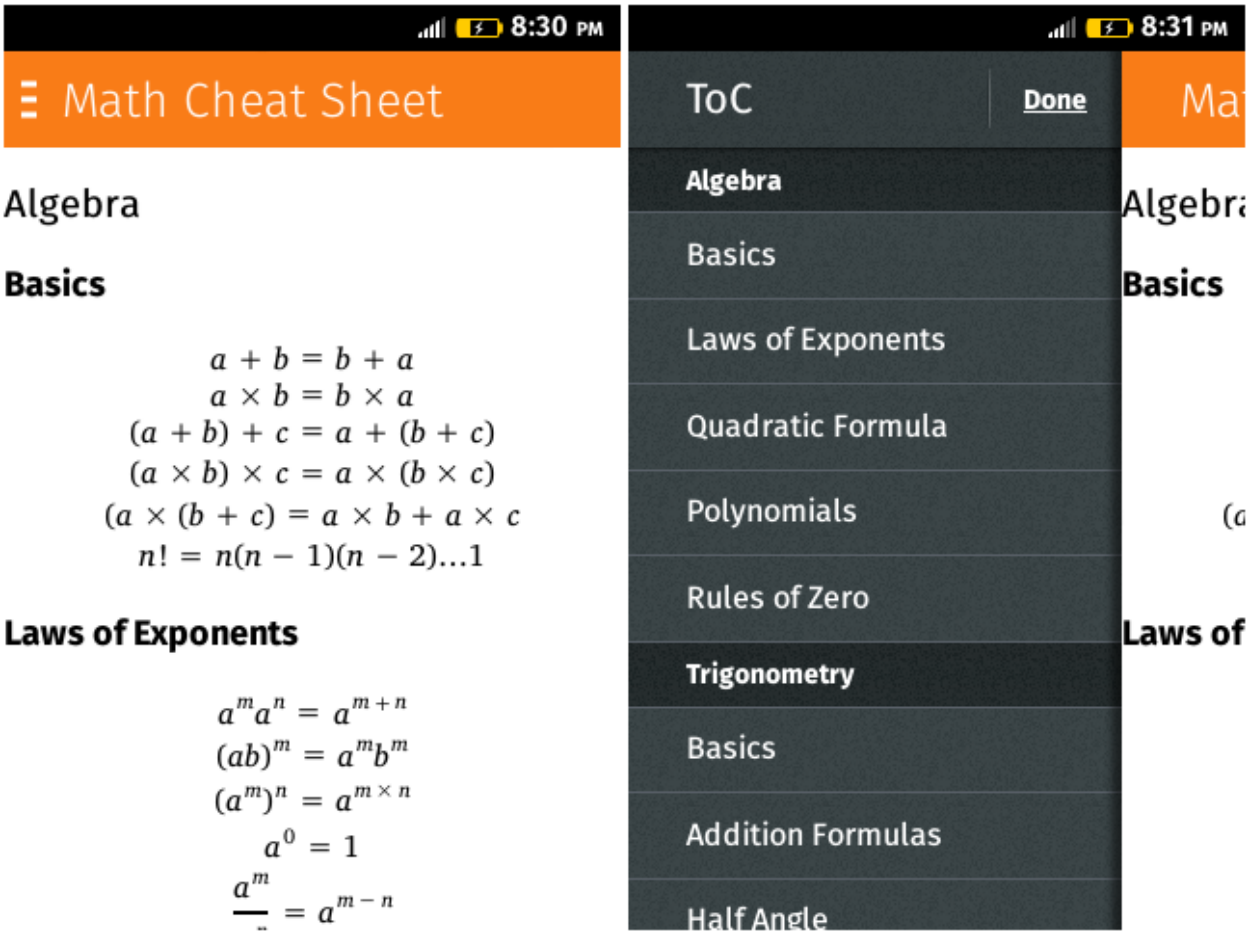
$$\Gamma(t) = \lim_{n\to\infty} \frac{n!\, n^t}{t\,(t+1)\cdots(t+n)} = \frac{1}{t}\prod_{n=1}^{\infty} \frac{\left(1+\frac{1}{n}\right)^t}{1+\frac{t}{n}} = \frac{e^{-\gamma t}}{t}\prod_{n=1}^{\infty}\left(1+\frac{t}{n}\right)^{-1} e^{\frac{t}{n}}$$

B Firefox OS Apps

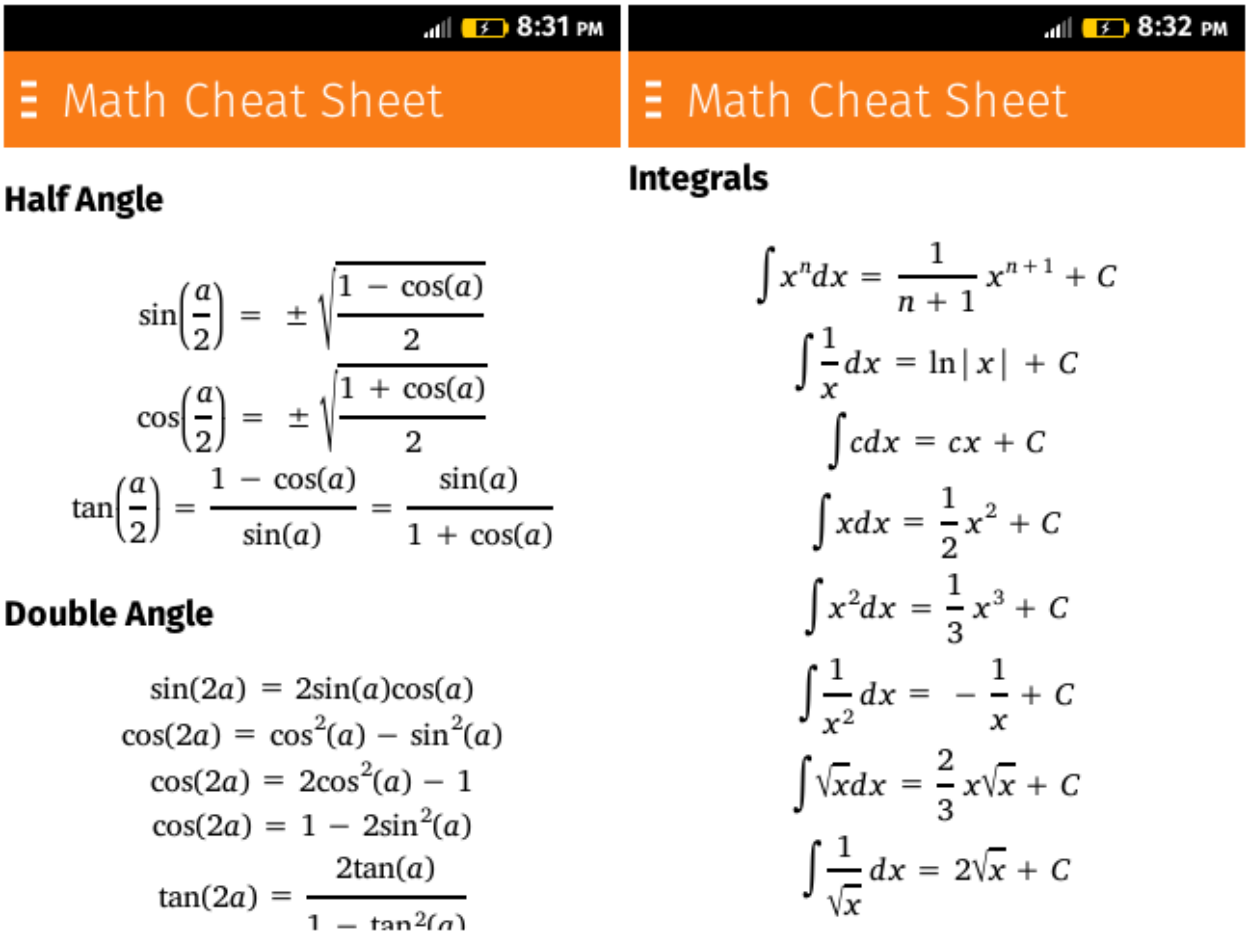
All the apps presented in this paper work in a web browser that supports MathML. The screenshots in this section were taken with one phone running Firefox OS powered by Gecko 31 and can be a little different from what the reader gets when trying the apps with a web browser or Firefox OS device. As discussed in 1.3, Firefox OS does not have pre-installed MathML fonts at the moment and so the rendering of mathematical formulas is of lower quality.

B.1 Math Cheat Sheet

All the equations are presented in display mode and a table of content is available at the top left corner of the app.



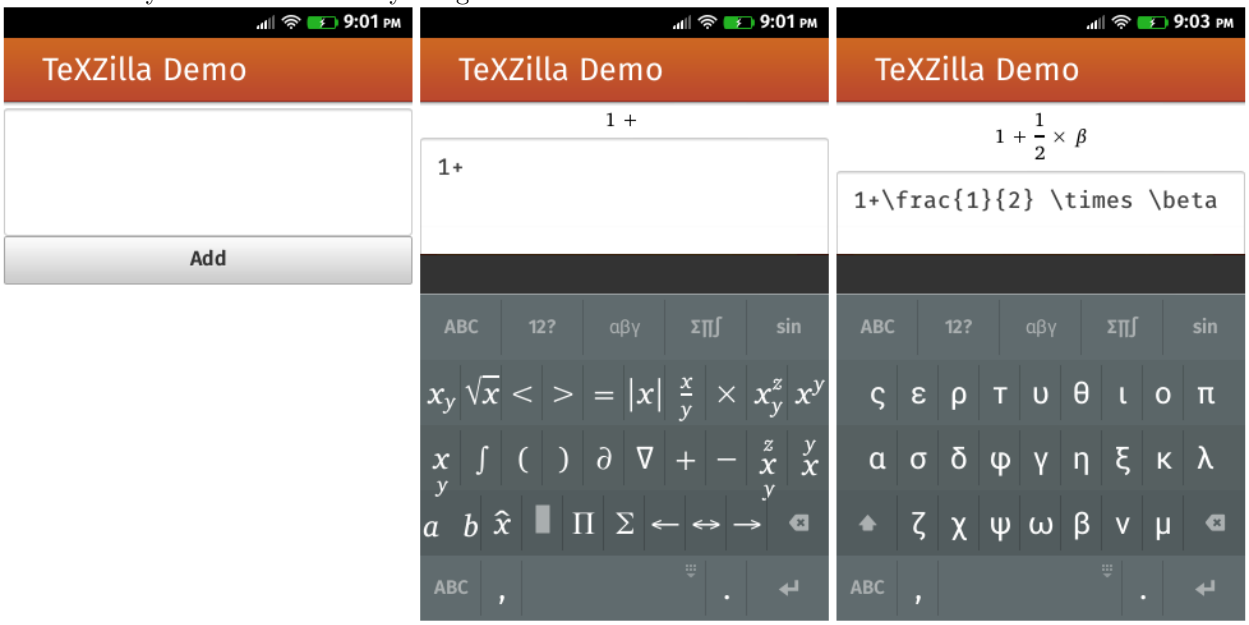
More complex equations are shown below.



B.2 TeXZilla App

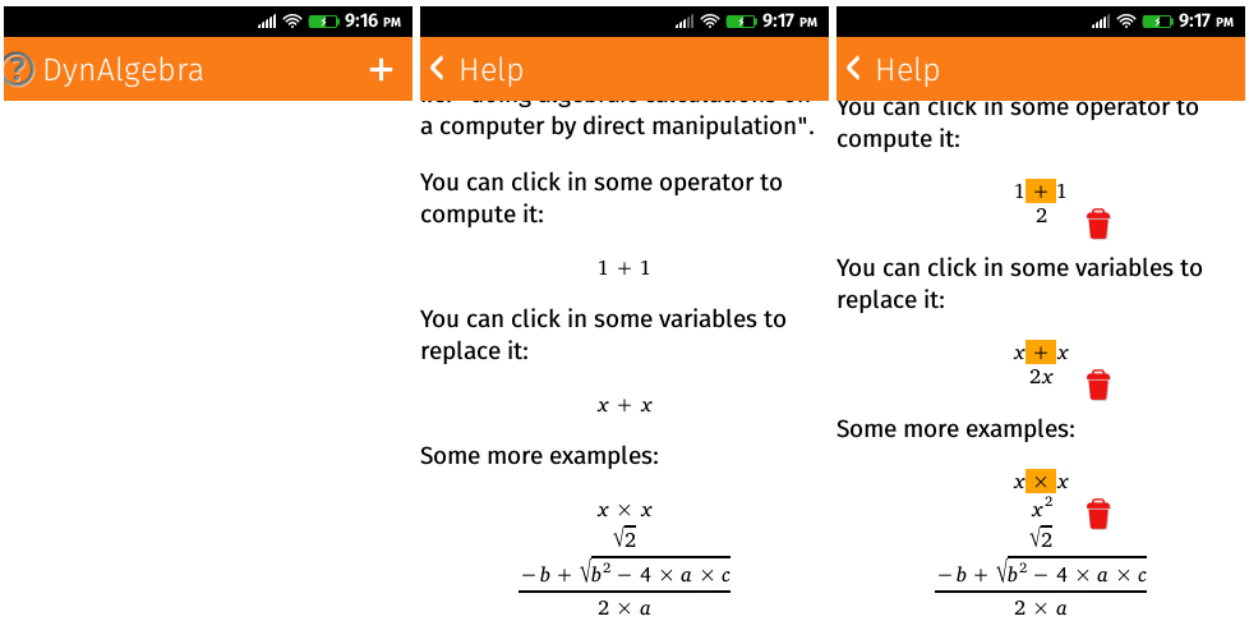
The app has an input box for the LaTeX expression, a few help buttons and an "Add" button. When the user enters the LaTeX expression a preview of the equation is provided (second screenshot) and after pressing the "Add" button the equation is saved and a "delete" icon appears on the right of it.

The screenshots below show a version of the app without the "help" buttons. Instead, a virtual keyboard with suitable keys for math is directly integrated into Firefox OS.



B.3 DynAlgebra

The app has a small help menu accessible from the icon at the top left corner of the screen. It provides examples of features currently available.



From the first screenshot the user can add a new equation using the icon at the top right corner of the screen. After typing the LaTeX expression for the new equation, the user can add it and manipulate it.