

Drag-and-drop of Formulæ from a Browser

Paul Libbrecht and Dominik Jednoralski
Competence Center for E-Learning, DFKI GmbH, Saarbrücken, Germany
(jedom|paul)|(activemath.org|dfki.de)

Abstract

Formulæ in a semantic source such as OPENMATH [2] can be presented in web-browsers with the benefit of features that can help in reading and understanding them. Important aspects include the rendering quality as well as tooltips to indicate the semantics of presented symbols.

Within the ACTIVEMATH learning environment we have added another feature: the ability to drag-and-drop terms from presented formulæ into formula-input places such as the exercise formula editor, the function plotter, or to the external world.

This drag-and-drop is initiated using a contextual-menu which comes up by clicking on a presented term. We describe the user-interface limitations that have lead us to require these extra interactions instead of relying on the simple copy-and-paste paradigm. We explain how the current mechanism works and sketch how ACTIVEMATH could collaborate with external receiving applications.

Keywords: drag-and-drop, copy-and-paste, formulæ, OPENMATH, web-browsers

[...] Having found the relevant information, you would have to copy this into your clipboard (or the back of an envelope) and proceed to then paste in (or retype) the information into the relevant document. The type of information obtained was more often a factor of least resistance than the most relevant, timely or accurate.

*T. Berners-Lee, J. Hendler, E. Miller
Integrating Applications on the Semantic Web [8]*

1 Introduction

The ACTIVEMATH learning environment [14] presents mathematical documents from a source encoded in the OMDoc [11] language. These are made of *mathematical content items* each containing metadata, text, embedded elements, and mathematical formulæ. The latter are based on the OPENMATH standard.

Mathematical documents are presented to the user as part of a learning experience, and her active participation is expected among others within interactive

exercises. In order to help the learner in her input it appears natural to try to involve transfer paradigms such as copy-and-paste or drag-and-drop: in the presentations she sees, much content is actually close to what she should input. Typical situations where we expect the transfer paradigm to be useful include:

- In the process of reading examples of mathematical constructs, such as a real function or a group presentation, the learner will often want to manipulate this object further in order to explore this property. The transfer of a representation of this object into tools such as an explorative computer-algebra-system or the function plotter can then be used. Because these tools may be only capable of handling the concrete object representations and not, for example, an equation giving it a name, the sub-term of the representation has to be transferred.
- In the process of an interactive exercise where the learner has to compute, say, the derivative of a real function, the transfer of the function from the question presentation to the places where to input appears naturally, followed by a manipulation to achieve the differentiation.

Most of the content presented to the learner in `ACTIVEMATH` should thus be transferrable including text fragments and terms of formulæ. The transfer should be received by the input of text or formulæ within interactive exercises and by the other exploratory activities such as the computer algebra system or the function plotter. Formulæ should be transferrable in part or in whole so that terms in large formulæ can be transferred even though the complete formulæ could not. The transfer should support consistently the extensibility offered by `OPENMATH` content dictionaries.

If possible, transferring content from `ACTIVEMATH` to the outside world should be possible. Aside from textual and image transfer, it could be a plus to be able to transfer formulæ to desktop applications.

1.1 Outline

The paper is organized as follows: we first present results of our explorations for technologies supporting our wishes of transfer from and to components living in a web-browser. The presentation of formulæ in `ACTIVEMATH` is then described and the interactions it offers. The reception and translation operations are then described with hints to reception by external applications. We conclude with future work.

2 Transfer Capabilities in Browser

Let us recall that the two classical forms of transfer are:

- copy-and-paste, which involves selection, copy-action, storage of all transfer-data in the clipboard, and paste-actions at insertion point which reads the clipboard data and inserts the chosen content format.
- drag-and-drop which involves dragging-out a visual element, moving the mouse until the destination, accepting the drop, followed by an actual transfer in the content format agreed upon between the two applications.

2.1 Transfer Actions of Browsers' Content

Contemporary web-browsers, which are targeted clients of ActiveMath, offer very little support for advanced forms of transfer.

Only the following transfer-interactions have been observed to work with contemporary browsers:

- copy and paste and drag-and-drop of text extracts selected by the user to either input-forms in the browser or to external applications. Depending on the browser and target applications, the transfer may copy the whole HTML of the underlying fragment (Internet Explorer (see §6.2) offers this), or a platform-dependent rich-text interpretation.
- copy-and-paste and drag-and-drop of images work between web-browsers and most image-aware applications on Windows and MacOS X.
- drag of links as URIs is easily done from most browsers to the exception of versions up to 1.6 of the Mozilla browser-suite (see §6.1) on Linux. Receiving applications can be any application that can process text.
- Dropping links to web-browsers' textual input elements is only possible in the Mozilla (see §6.1) family of browsers. When receiving a URI-drop anywhere except in plug-in components (e.g. in applets), both Internet Explorer (see §6.2) and Safari-based browsers (see §6.3), trigger no less than the opening of this URI as a URL to be presented in the browser window.
- MATHML-encoded formulæ [3] embedded in web-pages can be selected and copied, moreover MATHML allows the usage embedding of semantic content aside of the presentation-MATHML which is copied along. The selection is only to whole formulæ in MathPlayer (see §6.2) or is purely textual in Mozilla (see §6.1). The resulting copy is either a complete formula or a piece of text.

From this observation, we conclude that text selection, hence copy-and-paste transfers, can work for plain or styled text only or complete formulæ. We also conclude that drag-and-drop of links is working well.

forming the difference quotient:

$$\frac{f(x)-f(x_0)}{x-x_0} = \frac{a \cdot x^n - a \cdot x_0^n}{x-x_0} = \frac{a \cdot (x_0+h)^n - a \cdot x_0^n}{h}$$

From the binomial theorem we obtain

$$(x_0+h)^n = \sum_{j=0}^n \binom{n}{j} \cdot x_0^j \cdot h^{n-j} = \left(\sum_{j=0}^{n-1} C_j^n \cdot \right.$$

this, we get: binomial coefficient

$$\frac{f(x)-f(x_0)}{x-x_0} = \frac{a}{h} \cdot \left(\sum_{j=0}^{n-1} C_j^n \cdot x_0^j \cdot h^{n-j} \right) = a \cdot$$

Figure 1: Value-added presentation of formulæ

2.2 An API for Transfer at the Scripting Level

JavaScript is the scripting-language of web-browsers. However, an API to support user-initiated-transfer is absent from the Document Object Model [5] specifications be it copy-and-paste or drag-and-drop. APIs for both paradigms exist in incompatible versions for MicroSoft Internet Explorer (documented in [10]), Mozilla (documented at [4]), and Safari (documented at [9]).

For drag-and-drop, a standardization is in process in the W3C webAPI working group devoted to the development of standards that enable rich web clients.¹

For copy-and-paste the methods are reserved to trusted applications since the API gives direct read and write access to the clipboard which would let any webpage *sniff* a user's clipboard. ACTIVEMATH could have gone down the route of being trusted using distribution of signed executables and request of the users for trust as is done with signed applets. We have experienced, however, that such request can make users feel uncomfortable and have preferred to avoid it.

We have thus abandoned the exploitation of the familiar copy and paste mechanisms with the user's default clipboard and have given preference to the drag-and-drop paradigm which does not incur the risk of a hidden clipboard-reading since it allows the user to explicitly indicate the source and target of the exchange.

3 Presentation of Mathematical Formulæ and their Transfer

Having described the wishes of our user-initiated-transfer and the technological landscape, we now turn to the implementation in ACTIVEMATH of the transfer of mathematical formulæ using a contextual menu and the drag-and-drop paradigm.

¹More about the webAPI Working Group is at <http://www.w3.org/2006/webapi/>.

3.1 Presentation and Sub-term highlight

Formulae presented in `ACTIVEMATH` are generated from `OPENMATH` content using the presentation architecture, an authorable rendering engine for documents with mathematical formulae [13].

The rendering output, for formulae, is done either in HTML (with CSS), MATHML-presentation, or `TEX` (finally rendered to PDF or SVG). The semantic of the formulae encoded in `OPENMATH` is presented with added-value features:



- tooltips indicating the (localized) name of each symbol when the mouse is on the operators of this symbol
- links to the search for symbol's definition on click on the presented symbol
- in the case of HTML, a highlight of the presentation of the smallest sub-term behind the mouse. Figure 1 shows such a presentation.

This highlighting is performed by decorating, before XSL-application, each of the source's `OMOBJ` elements (the root of each formula) with the identifier of the containing item followed by it's order-number and the XPath coordinate of each `OPENMATH` application elements (`OMA`). These coordinates are passed along the transformation as `id` attributes of elements enclosing the rendering of each term. When presented in the browser, a script can use the elements hierarchy as well as the `id` attribute-values to change the background color of the rendering of the smallest sub-term term under the mouse.

3.2 The Formula Context Menu

When the user clicks on the formula, a contextual menu appears offering to:

- search for the indicated term in the `ACTIVEMATH` search tool [12].
- search for definitions of the presented symbol that was clicked.
- display this term in MATHML or `OPENMATH` either in original form (which potentially contains many unknown symbols) or translated to have terms in appropriate `OPENMATH` CD-group if possible. This display is delivered by the *clipping-controller*
- This menu offers to prepare the term-highlight for later drag, which draws a black box around the sub-term by applying a layer on top of the term. The content of this layer is entirely draggeable because it is an anchor. This link's URL points to the *clipping-controller*, which delivers the term in various formats.

  **Example of Axiom** ★★★★★

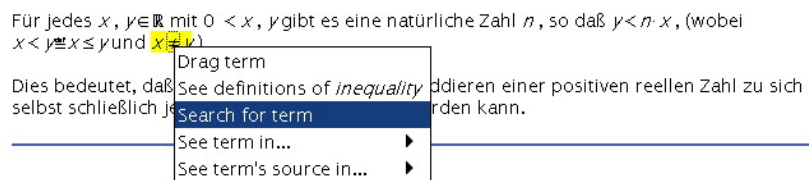


Figure 2: The contextual menu on a term.

3.3 Receiving and Transferring Formulæ

We have indicated that a hyperlink is dragged out to represent a term. The receiving applications such as the Wiris input-editor [1] integrated in ACTIVE-MATH, receive drops of URIs and launch the corresponding HTTP GET request. This request indicates the supported MIME-types: which are variants of the OPENMATH MIME type² by a parameter indicating the supported set of symbols as in:

```
application/xml+openmath;cdgroup=http://www.openmath.org/cdfiles/cdgroups/mathml.cdg
```

The *clip-controller* when requested for extracts and the OPENMATH from the content store and tries to apply the translation to obtain a term with symbols only in the wished CD-group, if that fails it tries the next supported content-type or returns an error. The translation is operated by the declaration of the relevant CD-groups and of *rephrase-rules* which are pairs of OPENMATH expressions mapping one to the other and are input by authors declaring new symbols. An example of such translation is for the symbol `unary-plus`: this symbol has been avoided as part of the standard OPENMATH content-dictionaries but is important for presentation purposes. The rephrase rule thus states:

```
<rephrase> <!-- maps +a to a -->
  <OMOBJ>
    <OMA>
      <OMS cd="basics_symbols" name="unary_plus"/>
      <OMV name="a"/>
    </OMA>
  </OMOBJ>
  <OMOBJ>
    <OMV name="a"/>
  </OMOBJ>
</rephrase>
```

²The OPENMATH MIME type is, actually, not registred in the central authority at IANA but the usage `application/xml+openmath` follows naturally from the RFC-3023 [15].

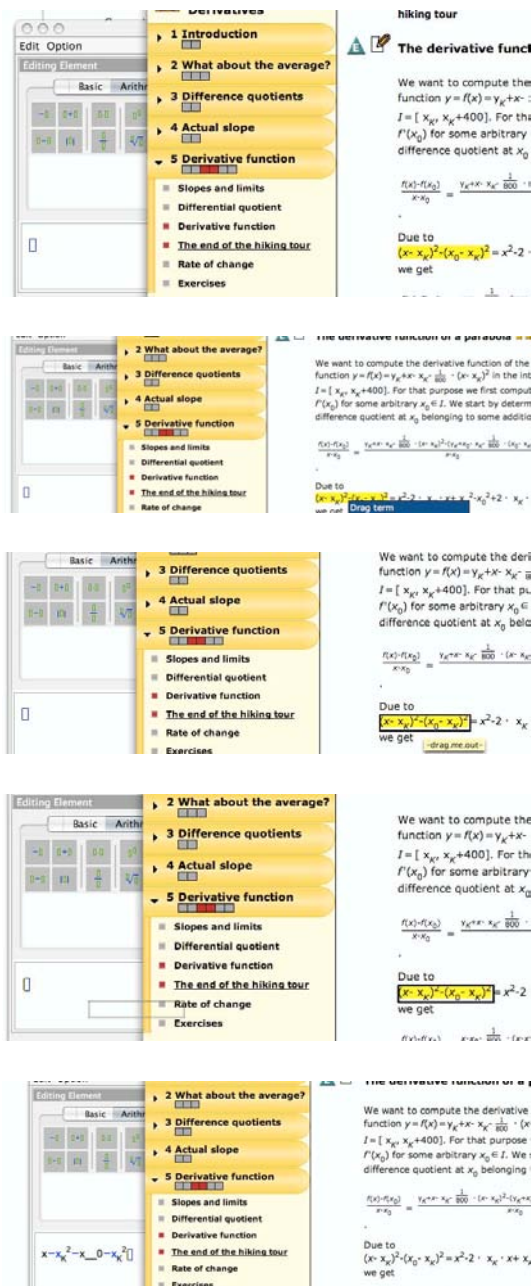


Figure 3: The steps of the drag-and-drop of a term: hover the mouse to see the term, right-click, invoke drag-term in the context menu, drag-it, drop it input editor.

The translation procedure may be concluded by an XSL transformation and a *compilation* which provides the visual formats indicated above.

The Wiris Input Editor is used in the exercises [7] and search tools as formula editor with palettes of choosable symbols. This editor is extensible and a translation from the symbol-presentation-pairs of the notation system [13] is being realized. This will guarantee that any symbol dropped on the input-editor from its enclosing ACTIVEMATH can be presented.

Drops can also go to input places which cannot be easily extended such as the function plotter used in ACTIVEMATH.³ Drag-and-dropping to the function plotter requires both:

- sub-terms to be transferred and not only complete formulæ since functions presented in texts or exercises are very often presented as equations and not as simple functions
- the transfer to perform translations so as to allow authors to define new symbols and their notations independently of the supported symbols of the plotter and provide rephrases if possible

Drag-and-drop to the function plotter is now at the prototype stage and has been implemented by an XSLT-stylesheet on the received OPENMATH term which produces the linear input of the plotter component. When the drop is successfully converted to a function in one variable, a graph is automatically drawn.

External desktop applications can be consumers of this transfer form in two fashion:

- if wishing the best integration within ACTIVEMATH, applications receiving a drop of *clip* URLs could GET them as the input-editor and the plotter do. The ability for HTTP requests to specify the supported content-types as well as the user-agent identifications leaves space for such a communication to be highly tuned.
- using the functions of the formula context menu, the user can also select an appropriate format to view followed by the invocation of the copy-and-paste actions.

For the presentation-oriented applications such as graphical programmes or word-processors, the presentation formats such as MATHML, HTML, TEX or PDF should provide good support.

The MATHML format's **semantics** elements allow the embedding of data which can be received by contemporary desktop computer-algebra-systems. We intend to put the stylesheets at [6] to use for this. It is important to note that, in this case, the translation target is not known in advance, as a result, all possible translation should be included as **semantics** elements. This may imply a large number of translations.

³The function plotter in ACTIVEMATH is derived from the Java Components for Mathematics of David Eck and others at <http://math.hws.edu/javamath/>.

4 Conclusion

This paper has described the implementation in ACTIVE-MATH for user-initiated transfers. Possible implementation avenues have been explored. As a result we decided to solely rely on drag-and-drop of links, simulating selection via JavaScript. But the overload of possible actions, which the user can initiate on a term has led us to the introduction of a popup menu. This menu contains the possible interactions on the highlighted term and has allowed us to add actions on terms, for example, the *search for term*.

To our experience, the drag-and-drop paradigm is less usable than the copy-and-paste paradigm for most users, because the former requires both the source and target windows to be visible on screen.

For this reason we are currently proposing an approach called *passive copy-and-paste* to the W3C WebAPI working group: untrusted scripts, which are embedded in web-pages, are called to deliver the content to be copied when the user invokes standard gestures to copy (e.g. using the menus of the browser or pressing CTRL-C).

An emerging movement has been launched by Ray Ozzie called *Live Clipboard* whereby the clipboard is shared on the Web, see [16]. It is probable that the ACTIVE-MATH copy-and-paste efforts could converge with these efforts.

Our implementation allows each term to be seen in several formats. This supports exchanges to external applications. The translation processes from OPEN-MATH are also applied when transferring to internal tools such as the function plotter. We have provided evidence that such translations are necessary.

The usability of the currently implemented drag-and-drop mechanism will be evaluated in September 2006 with about 100 students in calculus courses at high-school level. We shall report on these experiments.

5 Acknowledgements

This publication is partly a result of work in the context of the LEACTIVE-MATH project, funded under the 6th Framework Program of the European Community – (Contract IST-2003-507826). The author is solely responsible for its content. More information about the LEACTIVE-MATH project can be read from <http://www.leactivemath.org/>.

References

- [1] Ramon Aixarch, Dani Marqués, Olga Capprotti, and Tim Smith. The Wiris input editor. In *Submitted to Proceedings of MathUI-06*, 2006.
- [2] Stephen Buswell, Olga Caprotti, David Carlisle, Mike Dewar, Marc Gaëtano, and Michael Kohlbase. The openmath standard, version 2.0.

- Technical report, The OpenMath Society, June 2004. Available at <http://www.openmath.org/>.
- [3] D. Carlisle, P. Ion, R. Miner, and N. Poppelier. Mathematical markup language, version 2.0, 2001. <http://www.w3.org/TR/MathML2/>.
 - [4] Neil Deakin. Drag and Drop. Mozilla Developer Center, Accessed in June 2006, See http://developer.mozilla.org/en/docs/Drag_and_Drop.
 - [5] A. Le Hors et al. Document object model level 2 core specification. See <http://www.w3.org/TR/DOM-Level-2-Core>.
 - [6] Ontario Research Centre for Computer Algebra Software Team. Translator between openmath and mathml. See http://www.w3.org/Math/Software/mathml_software_cat_converters.html#Itranslator_between_openmath_and_mathml.
 - [7] G. Gogvadze, A. González Palomo, and E. Melis. Interactivity of Exercises in ActiveMath. In *In Proceedings of the 13th International Conference on Computers in Education (ICCE 2005)*, pages 107–113, Singapore, 2005.
 - [8] J. Hendler, T. Berners-Lee, and E. Miller. Integrating applications on the semantic web. *Journal of the Institute of Electrical Engineers of Japan*, 122(10):676–680, October 2002. See <http://www.w3.org/2002/07/swint>.
 - [9] Apple Computer Inc. Safari javascript programming topic: Using the pasteboard from javascript. Accessed in June 2006, Available at <http://devworld.apple.com/documentation/AppleApplications/Conceptual/SafariJSProgTopics/>.
 - [10] Microsoft Inc. About DHTML Data Transfer. See <http://msdn.microsoft.com/workshop/author/datatransfer/overview.asp>.
 - [11] M. Kohlhase. OMDoc: Towards an OPENMATH representation of mathematical documents. Seki Report SR-00-02, Fachbereich Informatik, Universität des Saarlandes, 2000. See also <http://www.mathweb.org/omdoc>.
 - [12] Paul Libbrecht and Erica Melis. Methods for Access and Retrieval of Mathematical Content in ActiveMath. In *To Appear in Proceedings of ICMS-2006, LNCS*. Springer Verlag GmbH, 2006. Available from <http://www.activemath.org/publications/Libbrecht-Melis-Access-and-Retrieval-ActiveMath-ICMS-2006.pdf>.
 - [13] S. Manzoor, P. Libbrecht, C. Ullrich, and E. Melis. Authoring presentation for OpenMath. In Michael Kohlhase, editor, *Mathematical Knowledge Management: 4th International Conference, MKM 2005, Bremen, Germany, July 15-17, 2005, Revised Selected Papers*, volume 3863 of LNCS, pages 33–48, Heidelberg, 2006. Springer.

- [14] E. Melis, E. Andrès, J. Büdenbender, A. Frischauf, G. Gogvadze, P. Libbrecht, M. Pollet, and C. Ullrich. ActiveMath: A Generic and Adaptive Web-Based Learning Environment. *International Journal of Artificial Intelligence in Education*, 12(4):385–407, 2001.
- [15] M. Murata, S. St.Laurent, and D. Kohn. XML media types, January 2001. <ftp://ftp.rfc-editor.org/in-notes/rfc3023.txt>.
- [16] Ray Ozzie. Simple bridge building. Keynote at the Emerging Technology Conference, 2006, San Diego, California. See also <http://liveclipboard.org/>.

6 Annex: List of Browsers and Platforms

Because of the strong web-browser orientation of this report, we include a list of the browsers explored and considered for the LEACTIVEMATH.

6.1 The Mozilla Family

Centered around the Mozilla FireFox web-browser, this open-source project delivers web-browsers for most contemporary platforms (Windows 98-XP, many Linuxes and BSDs, MacOSX >10.2). Current versions include Mozilla 1.7 and FireFox 1.5. More information can be read from <http://www.mozilla.org/>.

This browser is the choice target for ACTIVEMATH-project and is supported. Mozilla displays MATHML natively (since version 1.0 with a broken support in versions Mozilla 1.3 until Firefox 1.0 on MacOSX). The installation of a set of fonts is required supplementary to the installation of the browser.

6.2 Microsoft Internet Explorer

Internet Explorer is the default web-browser deployed on the Windows operating system hence is the most used browser (more than 90%) Versions range from 5 to 6 with version 7 being a beta. More information can be read from <http://msdn.microsoft.com/ie/>. The ACTIVEMATH-project supports version 6. In order for MicroSoft Internet Explorer to display MATHML, a plugin called MathPlayer from Design Science Inc. can be installed.

6.3 Konqueror-based browsers

The Konqueror project is a the browser of the KDE desktop environment. Safari is based on it and is the default web-browser deployed on the Mac OS X operating system hence is the most used browser on this platform. OmniWeb is another project using this component. More information can be read from <http://konqueror.kde.org/>, <http://www.apple.com/macosx/features/safari/>, and <http://www.omnigroup.com/applications/omniweb/>. The ACTIVEMATH-project supports Safari only in a limited way. No Konqueror-based project supports MATHML yet to our knowledge.