UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Mathematica empowered applets for learning school algebra and calculus

Heck, A.; Boon, P.; van Velthoven, W.

Publication date
2008

Published in
Proceedings of the 9th International Mathematica Symposium (IMS 2008)

Link to publication

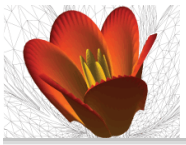
Citation for published version (APA):
Heck, A., Boon, P., & van Velthoven, W. (2008). Mathematica empowered applets for learning school algebra and calculus. In *Proceedings of the 9th International Mathematica Symposium (IMS 2008)* (pp. 1-20). Eindhoven University of Technology (TU/e). http://bmiaserver.bmt.tue.nl/eProceedings/WWW/IMS_2008_e-Proceedings.html

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:11 Oct 2026



Mathematica Empowered Applets for Learning School Algebra and Calculus

André Heck[†]

Peter Boon, Wim van Velthoven[‡]

[†]Universiteit van Amsterdam, AMSTEL Institute, Kruislaan 404, 1098 SM Amsterdam, The Netherlands, A.J.P.Heck@uva.nl

[‡]Universiteit Utrecht, Freudenthal Institute for Science and Mathematics Education, Aïdadreef 12, 3561 GE Utrecht, The Netherlands, P.Boon@fi.uu.nl, W.vanVelthoven@fi.uu.nl

We describe the development and use of *Mathematica* empowered applets for learning and practising algebra and calculus at secondary school level. We discuss how *Mathematica* can be used as back engine to evaluate students' answers and provide appropriate feedback. These applets provide scoring facilities, are fully integrated in a virtual learning environment in which answers are stored, and can be edited by teachers to create their own exercises and tests. Students benefit from the applets because the exercises help them to develop mathematical understanding and skills more thoroughly than is possible with pencil and paper. Using the applets, a teacher can review after a computer aided lesson what the students actually did, what progress they made, which problems arose during learning and to which mathematical subjects attention must be paid in the next lessons. In other words, the applets help the teacher to get insight into the learning process of students and to make grounded choices in the construction of his or her lessons.

Introduction

Many students at secondary school level and beyond have difficulties in mastering algebraic skills. Often the algebraic manipulations are carried out in a superficial way, using rules that are not meaningfully connected with the underlying concepts. To help secondary school students to develop real algebraic understanding (insights and concepts) and to learn and practice algebraic skills in sense-making ways, designers of the Freudenthal Institute have developed several sequences of programs, Java applets, that (1) enable students to connect the formal mathematical activities in a meaningful way with the underlying concepts and (2) provide exercises with feedback and have scoring facilities in a game-like setting. Later these applets were integrated in a virtual learning environment, in which information on the results and the learning could be captured. At the moment the Freudenthal Institute is extending this learning environment with authoring facilities, to create new applets without the need for specialized programming knowledge. This enables teachers to design their own interactive activities and tests on their own ideas.

Many of the interactive exercises are using build-in computer algebra alike facilities for evaluating students' answers and be for generating proper feedback. The evaluation of an answer can be the verification of the correctness of the final answer to a question, but it may also be the verification of the correctness of an intermediate computational step entered by a student. For the more advanced mathematical topics, the need for a professional computer algebra system like *Mathematica* to do this job is felt more and more. As a proof of concept, some of the developed applets are redesigned for using *Mathematica* as back engine to evaluate students' answers. The screen shot in Figure 1 gives an impression of the user interface for the case of exercising integration by parts.

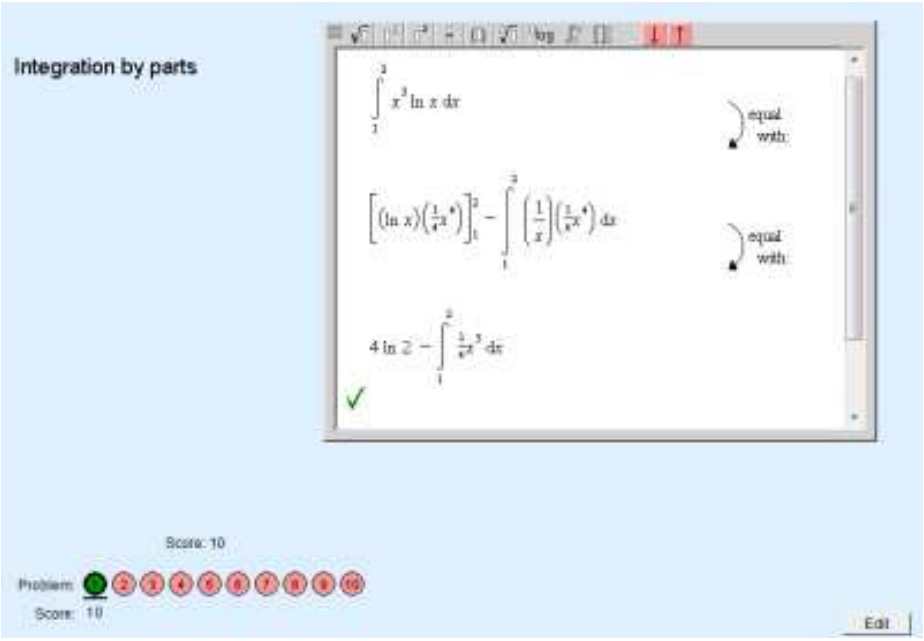


Figure 1. The 'Integrating by Parts' applet with immediate feedback

In this paper we will discuss our experiences from development work and classroom tests. We will pay special attention to the way *Mathematica* is used as back engine. We will also exemplify this use of *Mathematica* in the context of geometry education. The article is organized as follows. First we discuss the theoretical framework for the design and use of the educational applets, their use in the classroom via a virtual learning environment, the feedback provided in the applets, and the authoring of the applets. Hereafter we discuss in detail the use of *Mathematica* as back engine to evaluate students' answers on the basis of the exercise applet for integrating by parts. Finally we give a short example of the use of *Mathematica* in geometric theorem proving when using a dynamic geometry applet. We conclude with our ideas for future development resulting from our experience with the prototypical *Mathematica* empowered Java applets that have been developed so far.

■ Design and Use of Applets for Learning Mathematics

□ Theoretical Framework

In the last decade, a team of researchers, curriculum developers and software designers at the Freudenthal Institute have developed in collaboration with secondary school teachers various applets for learning mathematics. This development effort was made with the expectation that use of such mathematical applets would contribute to visualization of mathematical concepts, exploration of context situations (including purely mathematical contexts), and to bridging the gap between informal knowledge and formal mathematics. The Realistic Mathematics Education (RME, [1]) approach was the key theoretical framework for the design of the computer programs and the accompanying lesson material, and for the flanking educational research. The key element of the RME approach for the design and use of the applets is that the problems that they present are real and meaningful to the students and worth the effort of solving. Considerable work was put into the design and field testing of applets for students to develop algebraic understanding and skills, to introduce and strengthen the notion of function, and to develop geometrical insight and spatial abilities.

Two kinds of applets can be distinguished:

- *model applets*, which help to develop mathematical understanding;
- *exercise applets*, which help to develop mathematical skills.

In model applets visual representations are used as models for mathematical concepts. Within these representations students can work on the basis of their own ideas and experiment freely. An example of a model applet is the 'Geometric Algebra 2D' applet shown in Figure 2. It allows adding and multiplying numbers and variables by combining line segments to new line segments and rectangles. The lengths of the line segments and the areas of the rectangles are displayed as expressions. By splitting and joining rectangles students can develop insight in equivalence of various algebraic expressions.

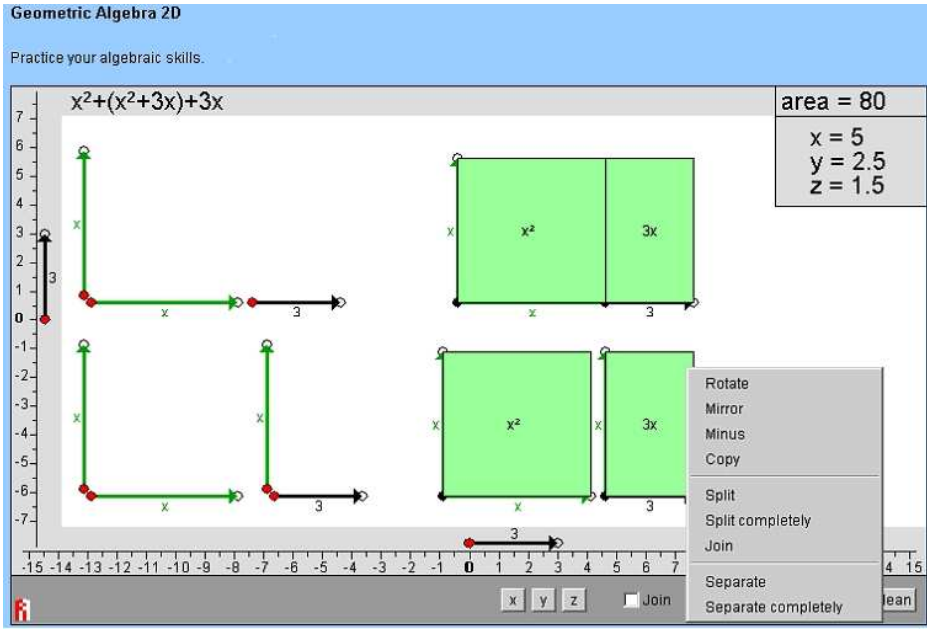


Figure 2. The 'Geometric Algebra 2D' applet.

A model applet such as the 'Geometric Algebra 2D' applet has an open nature: it allows students to work with it on the basis of their own constructions and productions. The ICT tool is only expected to help students to mathematise the situation on the basis of their activities. Activity sheets may guide students in making progress from informal to more formal mathematical activity.

Exercise applets have a more closed structure: a number of tasks of certain type are given to the students, their work is checked and feedback is provided. Figure 3 illustrates the 'Area Algebra' applet, in which the area model is used to verify equivalence of algebraic expressions. The main task is to find the correct expressions for areas and edges of rectangles. The applet is an interactive version of the well-known geometrical representation of the distributive law of algebra. A formula like $3(y+2)$ is treated as the area of a rectangle with length 3 and width $(y+2)$. This results in a stepwise formula with a part equal to $3y$ and a part equal to 6. Students intuitively learn and see that $3(y+2)$ is equivalent to $(3y+6)$.

Feedback in this applet is direct: green symbols in the work area indicate which part of the answer is correct, and red, orange and green circles around the assignment numbers indicate wrong, incomplete and correct answers. This stimulates the students to go on and try another one, and to get as many correct answers as possible. And what is also very important, the stored feedback provides both the student and the teacher detailed information about where and what kind of mistakes have been made. The screen shot shows an example: maybe the student had simplify forgotten to fill out a text box in

the rectangle; maybe (s)he had worked in a too formal manner and written 32 with $3 \cdot 2$ in mind, just as one writes $3y$ when the product is meant.

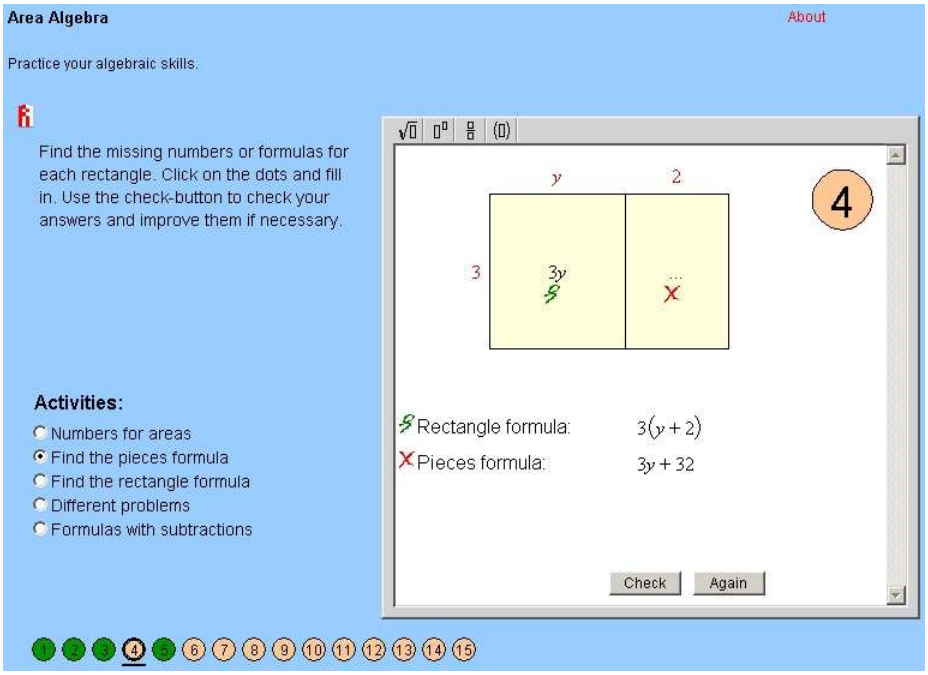


Figure 3. The 'Area' applet with immediate feedback.

The method of completing squares for solving quadratic equations links up well with the above area model. It is supported in the exercise applet 'Quadratic Equations' shown in Figure 4. The applet presents a quadratic equation that is to be solved step by step. With the red buttons on top, the student can select an operation that is to be performed on both sides of the equation. The goal is to simplify the equation and eventually end with $x = \ldots$ solutions. With this applet, a student has to think and decide what operation to perform. Two versions of the applets are available:

- one in which the computer performs the calculation at each step, thus allowing the student to concentrate on the solution strategy;
- one in which the student has to do the calculation himself and the program only verifies whether intermediate expressions are still equivalent.

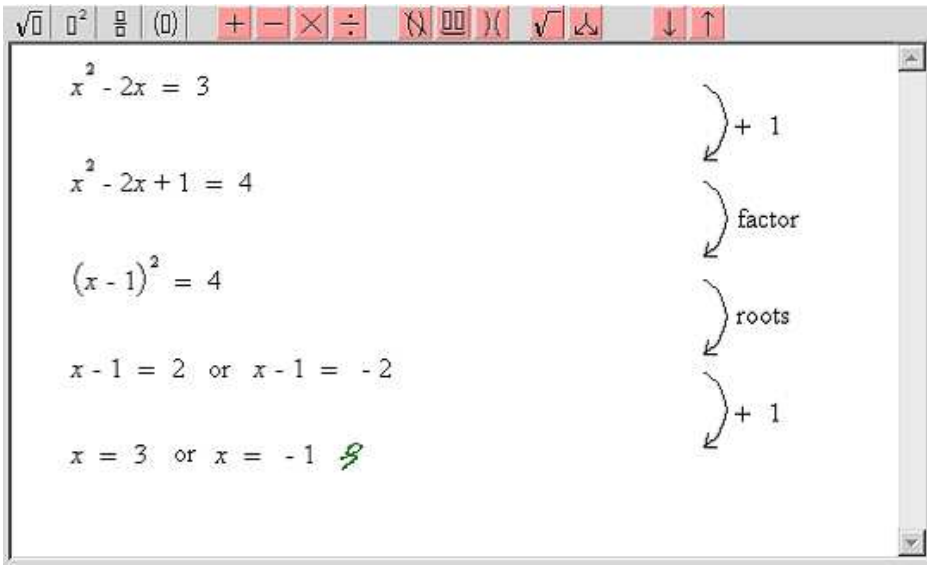


Figure 4. The 'Quadratic Equations' applet, using the completed squares method and the balance strategy for solving equations.

A gaming feature can be added to the second version by letting students earn points for entering correct intermediate results. Using help or letting the computer perform an intermediate step cost points. The idea, proven to be true in field tests, is that changing a set of mathematical exercises into a game with various levels motivates students to work hard.

One of the other exercise applets is the applet called 'Herleiden' (Dutch for 'rewriting') and shown in Figure 5. This applet presents mathematical expressions that are to be simplified step by step, where a student gets in each step feedback in the from of a cross or checkmark whether he or she is still on the right track or not. The purpose of this applet is to practise rewriting of given algebraic expressions and to reinforce students' symbol sense.

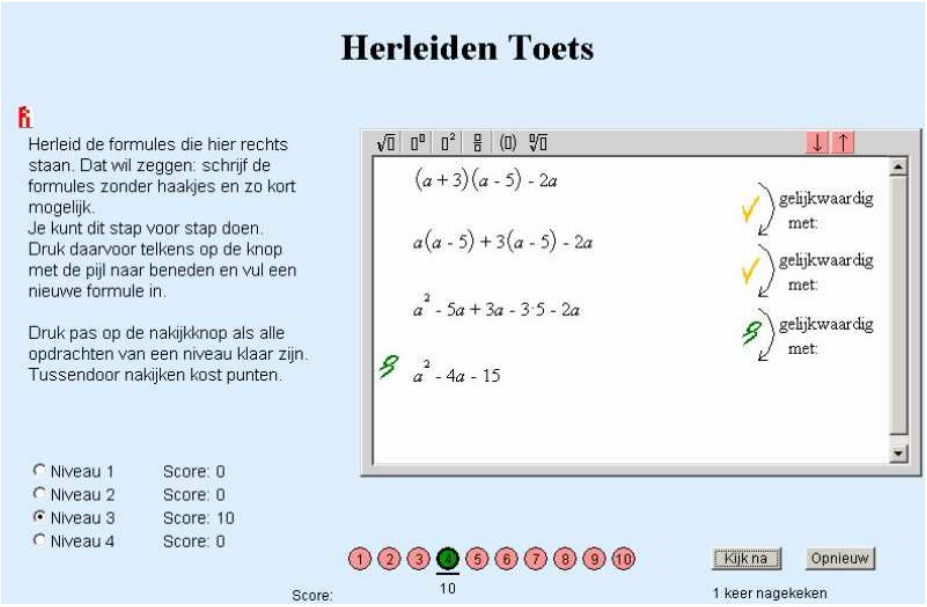


Figure 5. The 'Herleiden' applet with intermediate feedback.

While a student works with an applet (s) he receives instantaneous local feedback, i.e., feedback that is based on the typed answer or another interaction provoked by a question or task. Different forms of local feedback can be distinguished and often more than one of these forms is present in a single reaction. In Figures 1, 3, 4, and 5 green marks and red crosses indicate which part of the answer is correct and incorrect, respectively. In addition, red, orange and green circles around the assignment numbers indicate wrong, incomplete and correct answers. Another form of feedback in the applets are messages informing the student that "the notion used is invalid" or that "superfluous solutions are present in the answer". Via such intelligent, tailored feedback that goes beyond a simple correct/incorrect sign or a mark a student gets the impression that a teacher is looking over his/her shoulder to make useful remarks, to warn for common mistakes, and to give hints if needed.

The Digital Mathematics Environment

The applets can be embedded in a simple electronic learning environment, which is called the *Digital Mathematics Environment* (DME [2]; in Dutch called the 'Digitale Wiskunde Oefenomgeving', DWO). The DME is an answer to the fleeting character of the work with applets, which both students and teachers often experience. In the DME, a student logs in on a central server, works with an applet, and automatically receives feedback from the applet. Students' results are stored on a central server. This allows the student to review his/her work, correct it, and continue it in any location with Internet access, particularly at home. The teacher can check and grade the students' work and (s)he can monitor the learning process of individual students and of entire classes. An example of an overview of students' results is shown in Figure 6.

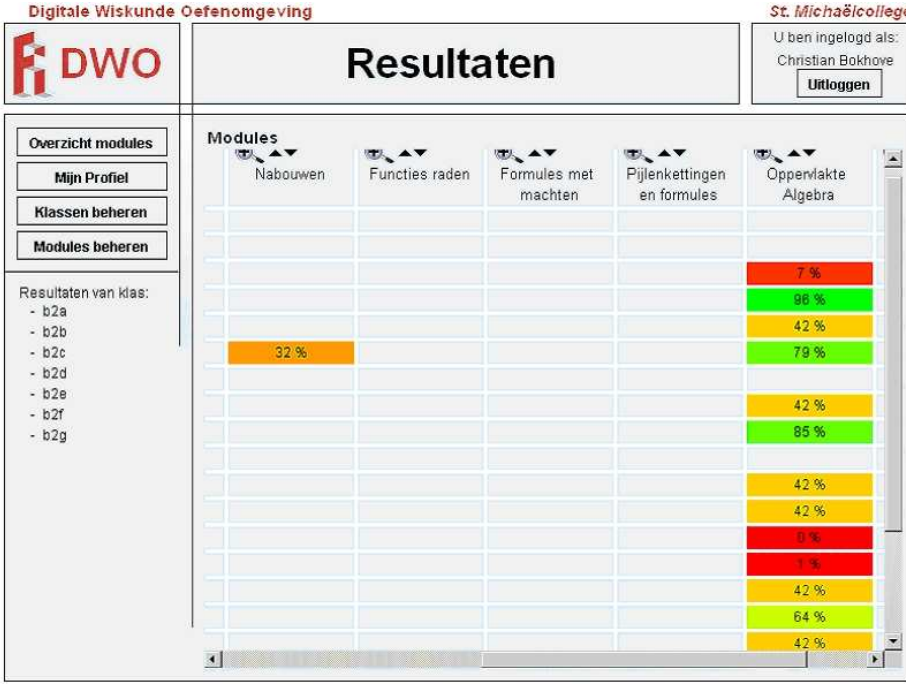


Figure 6. The results of all students in one class using the "Area Algebra" applet.

User-Friendly Authoring Facilities for Teachers

A promising new feature of the DME is the authoring tool for designing new exercise applets. A prototype of this tool was developed in the Galois project [3]. Lately much effort is put in the further development. The editor of the authoring tool enables teachers to create exercises and tests for their students similar to the existing applets, but set up according to their own ideas. The decision on the level of feedback and the marking system is left to the teacher who creates exercises and tests. Since teachers are not computer programmers, the process of creating the interactive exercises is kept as simple as possible. Teachers should be able to concentrate on the mathematical contents and instructional design instead of spending time and effort on technicalities. A teacher or author is able to look at the final result of the applet creation through the eyes of students. Keeping in mind that most teachers have no programming experience, the process of creating randomized questions should be kept simple too. Although designed for teachers, at the moment textbook authors use this authoring tool as well to design the accompanying ICT-materials for students.

Interoperability

The two functionalities of the DME (learning environment and authoring environment) can be separated. As an authoring environment, the DME can, on the one hand, package the designed activities in so-called SCORM-compliant learning objects [4] that can be used in other learning environments, for example Moodle or other SCORM-compliant environments. On the other hand, as a learning environment, the DME can also embed other (SCORM-compliant) learning objects that are not designed and produced with the DME. This possibility, especially the adoption of the SCORM-standard, facilitates the collaboration with other systems and designers.

Description of the Authoring Tool

Figure 7 is a screen shot of the editor of the authoring tool.

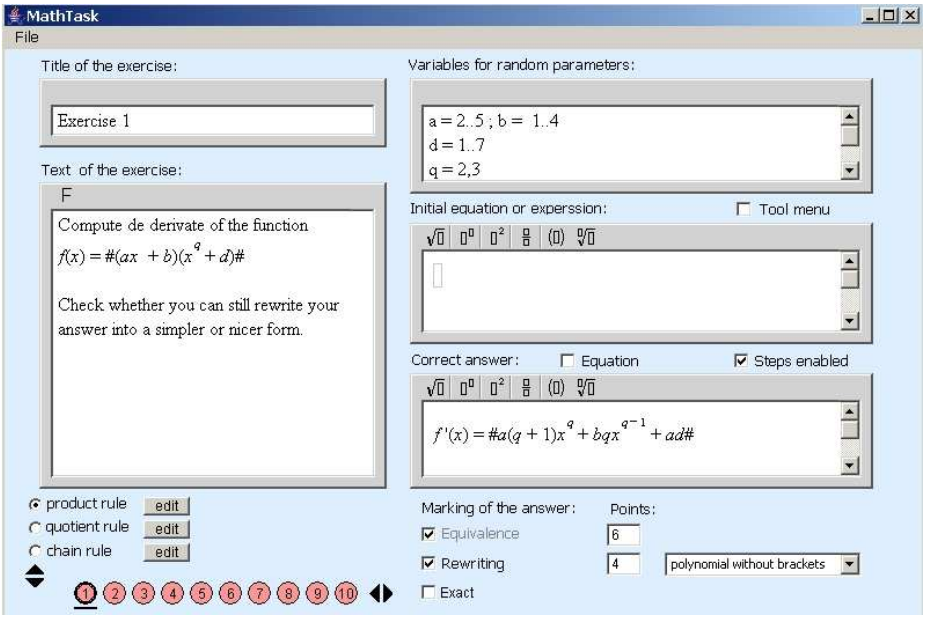


Figure 7. Creation of an exercise with (the prototype of) the applet generator.

We describe the fields in the above editor to illustrate the authoring of questions; more details can be found in [5].

Question variables are of the form parameter = *value*, separated by a semicolon when they are specified in one line. The *value* can be a sequence of integers separated by comma's or a range of values (denoted as a starting value and an end value, separated by the ellipsis (..). When a question is instantiated, a random choice from possible integer values will be made.

Initial formula is the expression or equation posed to the student in the work area. You use this field for example, when you do not want the formula in the question text. In the given example, the function whose derivative must be calculated is specified in the question text.

Question text is the actual question or exercise posed to the student. It is more or less a compulsory field that the teacher must enter for a valid question. It may contain text and formulas. In Figure 7 you see that the formula (invoked by pressing the F button) is a mathematical expression surrounded by hash signs (#). When the question is instantiated, all parameter values are randomly chosen, their values are substituted in the parts that are between hash signs, and basic simplifications (e.g., combining like terms, adding numbers, removal of superfluous zeros and ones in expressions) are automatically carried out.

Correct answer is another compulsory field that the teacher must enter; a checkbox is used to indicate that the answer is an equation instead of an algebraic expression. By using pairs of hash signs, the type of answer that is considered as correct in a randomized question may be specified (as we have

seen before). But there exist other ways of specifying the level of correctness. When the box 'Equivalence' is checked, any student answer that is equivalent to the teacher's answer is marked as correct and a number of points is scored. When the box 'Rewriting' is checked, additional points can be earned by the student when he or she has simplified an intermediate result. In the given example, extra points are given when the answer is a polynomial in expanded form, i.e., without brackets. Other special forms have been implemented in the editor.

Levels of sets of exercises can be specified in the left bottom corner of the user interface of the applet generator. In the example shown in Figure 7, the different levels have been named after the differentiation rule that must be applied. But one can also divide the exercises in a test into sets of easy and difficult exercises. The number of levels and the number of questions at each level can be set by pressing the up/down and left/right arrows, respectively.

Title of the exercise is optional. It can only contain plain text and no formulas. Like in the question field, you can only use alphabetic characters and digits. Use of accents and HTML coding is not provided (yet).

■ *Mathematica* Empowerment of the Applets

The main drawback of the generated Java applets is that there is no real computer algebra system behind it yet. Some limited computer algebra facilities are embedded, like expanding polynomials, simplifying a limited number of expression types and performing substitutions. But most of the equivalence testing is done numerically via a randomized process: random values are repeatedly substituted for the variables in the expressions under consideration and only when the results are notably different from each other, nonequivalence of the expressions is proven. If the tests are passed for enough substitutions it is assumed that equivalence holds. In a few cases, formal equivalence is tested, e.g., in case of arithmetic tasks and when removal of brackets in polynomial expression is applied. Although this numerical equivalence testing may lead to erroneous results — after all, nonequivalent algebraic expressions may evaluate to the same numerical result — in practice it works quite satisfactory for simple algebraic expressions like polynomials. However, when more complicated expressions or mathematical functions come into play, the need for algebraic equivalence testing is felt more and more. A modern computer algebra system can do the work in the background. In the implementation of the prototypical *Mathematica* empowered Java applets we use the **Simplify** command to this end.

□ A Problematic Example: Integration by Parts

Let us have a look again at the example of the *Mathematica* empowered Java applet in Figure 1. The task is to compute the definite integral

$$\int_1^2 x^3 \ln x \, dx . \tag{1}$$

Integrating by parts leads to the following equivalent expression:

$$\left[(\ln x)\left(\frac{x^4}{4}\right)\right]_1^2 - \int_1^2 \left(\frac{1}{x}\right)\left(\frac{x^4}{4}\right) dx\,, \tag{2}$$

which can be worked out into

$$4\ln 2 - \int_1^2 \frac{x^3}{4}\, dx\,. \tag{3}$$

But how can we easily check equivalence of expressions (1), (2) and (3)? Randomized equivalence testing is not possible and numerical integration does not seem appropriate either! Only use of a computer algebra system can really solve this equivalence problem, which is transformed into a zero test problem (also referred to as the problem of normal form simplification) that is in general more easily solved than the problem of bringing expressions to their canonical form.

□ Example of a *Mathematica* empowered Java Applet: Integration by Parts

Verification of the final answer and the intermediate results in the 'Integration by Parts' example shown in Figure 1 is carried out as follows. Whenever the user enters an expression, the parser checks whether the computer algebra system is really necessary for equivalence testing. For example, when the mathematical notation for integration is encountered, the decision is made that *Mathematica* is indeed needed and the whole expression that was entered by the user is translated toward a *Mathematica* expression. This translated expression is send to a *Mathematica* server accessible to the user and running in the background together with the previous expression (translated as well according to the syntax of the *Mathematica* language). The computer algebra system uses the **Simplify** command to verify whether the difference between these two *Mathematica* expressions is equal to zero. If so, the two expressions in the applet are equivalent and a checkmark indicating the student that (s)he may be on the right track can be set. If a non-zero difference was obtained by the *Mathematica* zero test, the two expressions in the Java applet are most probably not equivalent and appropriate feedback can be provided. You may be worried that failure of the *Mathematica* zero test does not guarantee that the expressions are non-equivalent — after all, you may write zero in such a complicated way that even the *Mathematica* command **Simplify** or **FullSimplify** cannot recognize it as zero — but it is our experience that this does not happen for mathematical expressions at the difficulty level of secondary school.

□ Implementation Details

Our approach is similar to *WebMathematica:* the student interacts via a browser with a Java applet (entering expression, dragging objects, etc.) and this interaction leads to one or more *Mathematica* expressions that are then carried out by a *Mathematica* program run on a server that is accessible to the user. The main difference is that we do not use the computer algebra system for evaluation of each and every mathematical expression. Our approach is simple: after the *Mathematica* command is carried out, the *Mathematica* session is 'cleaned' in that the user can no longer refer to the previous command or result.

All communication tools are based on Java Technology: Servlet technology, Java technology and special purpose Java technology developed for example in the OpenMath project [6]. The Java tools govern the interaction with the mathematical service offered by *Mathematica*. They make it possible to contact the computational back engine, submit queries to the service, and also take care of handling the responses obtained from this back engine. The browser-webserver based architecture used for supporting the exercise applets is shown in Figure 8.

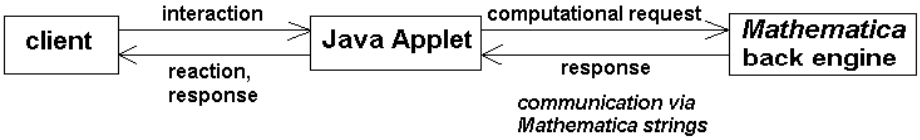


Figure 8. The general scheme of the *Mathematica* empowered architecture

We explain in some detail the main components shown in this picture.

A web browser with Java support represents the *client*. It will present views of the Java applets on an external server to the user, interact with the user, and communicate user interaction (mainly mathematical expressions entered) to the server. The responsibility for interaction is mostly on the server side.

The *Java applet*, which is a either a model applet or an exercise applet, caters to presentation, communication, and context. It supports a wide range of actions ranging from entry and handling mathematical expressions to providing hints and offering graph tools to support the students in doing their tasks.

The *Mathematica* web service serves as an interface to the computer algebra system. We have made use of standard protocols, among which those developed by RIACA at the University of Eindhoven [7]. In short, it is sockets-based technology that allows a *Mathematica* service to run the *Mathematica* package across the web and that relies on a dedicated program to handle the actual connection to/from a *Mathematica* engine. In order to run a *Mathematica* web service, a web server (e.g., Apache Tomcat) must have been set up. A main difference with other approaches such as MathDox [8] is that in our prototypical examples we have not yet used OpenMath technology [6] in

most applets: input to and from the *Mathematica* engine goes via strings in the syntax of the *Mathematica* language.

Another Example of *Mathematica* Empowerment: Geometric Reasoning with a Dynamic Geometry Applet

A dynamic geometry applet lets students build geometric constructions with a digital ruler and compass. The students can test whether certain assertions hold in their configuration. The applet is able to verify numerically these assertions for the displayed construction. The students can move the free points of the construction to other positions and the program will automatically test the assertions again. In popular packages like Cabri Geometry and Cinderella, a probabilistic prover is used to verify assertions [9].

A very simple example is the midpoint theorem:

The midpoint theorem.
Let ABC be a triangle and let D and E be the midpoints of AC and BC, respectively.Then the line DE is parallel to the base AB.

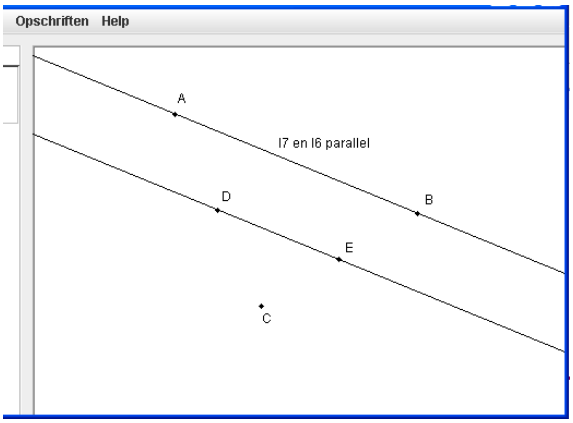


Figure 9. Snapshot of the dynamic geometry applet about the midpoint theorem.

A snapshot of the dynamic geometry applet is shown in Figure 9. When students make the construction, they may quickly conjecture the parallelism of the two lines drawn. The system offers tools to check such facts and here the text label in the middle of the screen informs the students that the system agrees with the assertion of parallelism of the lines. Dragging one of the free points A, B, or C, may convince the students about correctness of the statement.

The next step is that the students are able to verify their assertions in the general case. This is where *Mathematica* comes in. The applet replaces coordinates of the free points with variables and redoes the construction symbolically. Below we discuss the approach that applies for our concrete example.

We may choose a coordinate system such that the coordinates of the five points A, B, C, D, and E are as follows:

$A(0, 0)$

- $B(1, 0)$
- $C(x, y)$
- $D\left(\frac{x}{2}, \frac{y}{2}\right)$
- $E\left(\frac{1+x}{2}, \frac{y}{2}\right)$

The statement that the line AB is parallel to DE means in algebraic form that

$$(x_A - x_B)\times(y_D - y_E) - (y_A - y_B)\times(x_D - x_E) = 0\,. \tag{4}$$

Expression (4) is fed to *Mathematica.*

```
(0 - 1) * (y/2 - y/2) - (0 - 0) * (x/2 - (1 + x)/2) == 0
```

True

Because of our smart choice of the coordinate system, the formulas became very simple and *Mathematica* could perform the above zero testing by simply evaluating the expression entered. In general the mathematical formulas won't be that simple and the use of a computer algebra system is needed. Suppose that we do not make any assumption on the coordinate system, i.e., suppose that the coordinates of the five points are as follows:

- $A(x_A, y_A)$
- $B(x_B, y_B)$
- $C(x_C, y_C)$
- $D\left(\frac{x_A+x_C}{2}, \frac{y_A+y_C}{2}\right)$
- $E\left(\frac{x_B+x_C}{2}, \frac{y_B+y_C}{2}\right)$

In this case we must rely on the **Simplify** command of *Mathematica.*

```
(xA - xB) * ((yA + yC)/2 - (yB + yC)/2) -
    (yA - yB) * ((xA + xC)/2 - (xB + xC)/2) == 0
```

$$-\left(\frac{1}{2}(-x_B - x_C) + \frac{1}{2}(x_A + x_C)\right)(y_A - y_B) +$$
$$(x_A - x_B)\left(\frac{1}{2}(-y_B - y_C) + \frac{1}{2}(y_A + y_C)\right) == 0$$

```
Simplify[%]
```

True

The boolean value proves that the assertion holds. Especially when intersection points or points that are constrained to a particular line or circle come into play, one may easily get a full page formula, which *Mathematica* will happily simplify to `True`.

Alternatively, the constraints for the points D and E can be formulated as polynomial equations.

$$\begin{aligned} 2x_D - (x_A + x_C) &= 0 \\ 2y_D - (y_A + y_C) &= 0 \\ 2x_E - (x_B + x_C) &= 0 \\ 2y_E - (y_A + y_C) &= 0 \end{aligned}$$

Then the GroebnerBasis method can be applied to prove the geometric theorem (see [10] for a short gentle introduction to the theory): Use the **PolynomialReduce** command on a GroebnerBasis of the constraints.

```
polys = {2 xD - (xA + xC), 2 yD - (yA + yC), 2 xE - (xB + xC), 2 yE - (yB + yC)};
vars = {xA, xB, xC, xD, xE, yA, yB, yC, yD, yE};
gb = GroebnerBasis[polys, vars];
Last[PolynomialReduce[
    (xA - xB) (yD - yE) - (yA - yB) (xD - xE), gb, vars]] == 0
```

True

We end this section with a few remarks on the implementation of *Mathematica* support in geometric theorem proving. The approach is similar to the architecture in Figure 8. However this time the OpenMath [6] technology is used. The Java applet will contact a servlet, sending it the configuration and assertion in OpenMath style. This servlet then reformulates the assertion in an algebraic expression also in OpenMath style. With this formula the servlet will call a *Mathematica* service as developed by RIACA [7] after converting the OpenMath expressions to the *Mathematica* syntax and vice versa.

■ Conclusion

We conclude with a summary of our experiences with web-based mathematical tools for learning and practising algebra and with our ideas for future development resulting from our experience with the prototypical *Mathematica* empowered Java applets that have been developed so far.

First we quote some findings on the use of school algebra applets [11]:

"It was found that the use of applets has add-on value: they are fun and motivate students; they allow students to work at their own level of thinking and thus better address individual difference between students; the visual, interactive and dynamic features of applets makes the mathematics more easy to understand; thanks to the calculation power of the applets one can focus on the mathematical concepts and models; students are more creative and get more self esteem; the applets form a model students can fall back on; the practice and feedback features are much more powerful than pencil-and-paper exercises. However, to take fully advantage of these opportunities of applets, applets should be integrated in the daily mathematics class routine. Using the computed should not be a voluntary thing that is extra beyond regular math class."

The last issue played an important role in the GALOIS project that took place in the period 2005-2007 [3]. This project aimed to realize an integrated learning environment for learning and doing mathematics in which

- students can practise mathematics anytime and anywhere;
- mathematical assignments can be generated 'at random', so that the amount of exercise material becomes unlimited; mathematical assignments can be generated 'at random', so that the amount of exercise material becomes unlimited;
- activities and answers of students are stored;
- intelligent feedback on students' work is given automatically.

This project and its one-year extension project SAGE led much to the work presented in this article. We quote the conclusion drawn in [3]:

"Existing mathematical tools become even more powerful and useful when results and performance of students can be tracked online in a virtual learning environment with which teachers and students are familiar. Use of a common standard, in this case SCORM, enables the sharing of content, but also of storing student performance. Students know that they can be and are monitored by their teacher. Together with other features in virtual learning environments, especially for communication, students are stimulated to do their homework, or even more than that. Students do not seem to mind this 'big brother is watching you' concept. On the contrary, they seem to appreciate the fact that a teacher has good insight into common mistakes and can use this knowledge to address certain problems more effectively. They also seem to find that computer practice prepares them better for written exams than making assignments from a book. Computer aided learning is also seen as a more 'fun' way of practicing otherwise 'boring' subjects. Using direct feedback, giving students hints and enabling a 'trial and error' approach makes it possible for students to explore different strategies for solving a given problem.
In our opinion, digital assignments and diagnostic testing add to the motivation and performance of students in mathematics education. By making use of computers, the extra time involved for teachers is acceptable. Contact time can effectively be reorganized because students that need extra attention are signaled earlier."

In field tests at schools during the past projects it was found that all works well in practice, although many wishes for improvement of the software arose at these staged of the development work. Linking the applets to a backbone computer algebra system and having more question types at one's disposal were two of the envisioned extensions. The first steps to meet the first wish have been made with the *Mathematica* empowered Java applets.

■ Additional Material

Sample applets are provided on Internet: to get an idea login as *guest* on the website www.fi.uu.nl/dwo/en/ (accessed April 2008).

References

[1] Koeno Gravemeijer, “Developing realistic mathematics education”. Utrecht: cd-β Press, 1994.

[2] To get an idea of the Digital Mathematics Environment login as *guest* on the website www.fi.uu.nl/dwo/en/ (accessed April 2008).

[3] Christian Bokhove, Gerard Koolstra, Peter Boon, and André Heck, "Towards an integrated learning environment for mathematics". *Electronic Proceedings of the 8th International Conference on Technology in Mathematics Teaching* (ICTMT8) in Hradec Králové (E. Milková, Pavel Prazák, eds.), University of Hradec Králové, 2007. ISBN 397-80-7041-285-5 (cd-rom).

[4] Information about SCORM can be found at the website www.adlnet.gov/scorm/ (accessed April 2008).

[5] André Heck, Peter Boon, Christian Bokhove, and Gerard Koolstra, "Applets for learning school algebra and calculus". *Electronic Proceedings e+Calculus, 1st JEM Workshop*, Lisbon, 2007. www.jem-thematic.net/node/158 (accessed April 2008)

[6] OpenMath Society Website, www.openmath.org (accessed April 2008).

[7] Research Institute for Applications of Computer Algebra (RIACA), www.mathdox.org/new-web/ (accessed April 2008).

[8] Arjeh Cohen, Hans Cupyers, Erneste Reinaldo Barreiro, “MathDox: Mathematical Documents on the Web”. In: M. Kohlhase (ed.), “OMDoc : An Open Markup Format for Mathematical Documents [version 1.2]” (pp. 262-265), Berlin: Springer-Verlag, 2006.

[9] Ulrich Kortenkamp, “Foundations of Dynamic Geometry”. PhD thesis, ETH Zürich, 1999.

[10] André Heck, “Bird's-eye view of Groebnerbases”. *Nuclear Inst. and Methods in Physics Research* A 389 (1997) 16 - 21.

[11] Martin van Reeuwijk, “School Algebra Struggle, what about algebra computer games?”. ICME 10 Proceedings, Topic Study Group 9, Copenhague, Denmark, 4-11 July 2004. www.fi.uu.nl/publicaties/literatuur/6341.pdf (accessed April 2008).

About the Authors

André Heck earned MSc degrees in mathematics and chemistry. He is project manager at the Faculty of Science of the University of Amsterdam. His research area is the application of ICT in mathematics and science education.

Peter Boon is developer of educational software at the Freudenthal Institute for Science and Mathematics Education and the principal architect of the applets for learning mathematics and the digital mathematics environment.

Wim van Velthoven has degrees in physics and education. He is software developer at the Freudenthal Institute for Science and Mathematics Education, mostly developing applets for primary and secondary education.