PhidgetLab

Crossing the Border from Virtual to Real-World Objects

Michael Haupt[1] Michael Perscheid[1] Robert Hirschfeld[1] Lysann Kessler[2]
Thomas Klingbeil[2] Stephanie Platz[2] Frank Schlegel[2] Philipp Tessenow[2]
Hasso-Plattner-Institut
University of Potsdam, Germany
[1]{firstname.lastname}@hpi.uni-potsdam.de
[2]{firstname.lastname}@student.hpi.uni-potsdam.de

ABSTRACT

Teaching pupils the ideas behind objects in programming languages can be difficult since these concepts are mostly abstract and not comprehensible at first sight. Etoys as a visual programming environment counters such issues by introducing visible objects and simple tiles for programming them. However, all of these objects can only be experienced virtually on the screen. This paper presents PhidgetLab, a programming environment for electronic components (Phidgets) realised on top of the Etoys environment. PhidgetLab helps crossing the border from virtual to real-world objects. Pupils interact with tangible objects that are seamlessly connected to the digital world. PhidgetLab was evaluated in a case study with 22 pupils, following the principles of the Design Thinking methodology and comprised the realisation of five prototypes within a short period of time.

Categories and Subject Descriptors

K.3.1 [**Computers and Education**]: Computer Uses in Education—*Collaborative learning*; K.3.2 [**Computers and Education**]: Computer and Information Science Education—*Computer science education, information systems education*; D.1.5 [**Programming Techniques**]: Object-oriented Programming

General Terms

Design, Human Factors, Experimentation

Keywords

Etoys, Phidgets, PhidgetLab, Squeak, Design Thinking

1. INTRODUCTION

Visual programming languages are usually deemed a better tool than textual ones for introducing people, especially children, to programming. Object-oriented programming is generally regarded as a paradigm easily accessible by beginners, since it allows for modelling real-world entities as software objects.

The Etoys [1] programming environment provides clean object-oriented programming abstractions to inexperienced "programmers"; the typical target audience are children and pupils. One of the core strengths of Etoys is that *software objects* are directly mapped to *visual entities* that users can freely shape. Programming is made easy by employing *tile scripting*: no textual syntax is used; instead, control elements can be arranged using drag-and-drop techniques.

The objects that Etoys users deal with are purely virtual. However, tangibility, involving haptic abilities in the learning process, greatly improves on learning success [7]. This favourable approach—dealing with *tangible* real-world objects—is not supported by Etoys out of the box.

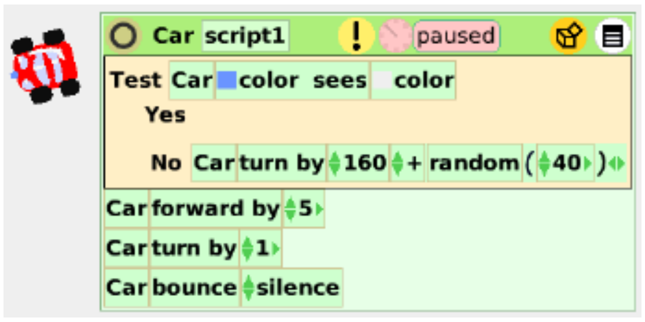
This paper introduces PhidgetLab[1], a programming infrastructure for hardware components built on top of the Etoys environment. Phidgets are electronic components connected to computers via USB. Different types of components (sensors and actors) are available for experimentation. The collection is well suited for being used in product prototypes. However, the programming means available for Phidgets so far are rather low-level and not beginner friendly.

In PhidgetLab, the benefits of both the Etoys programming approach and the ease of use of Phidgets components are combined. Object-oriented programming is, in a simple and easily accessible way, made available at the hardware level, allowing for dealing with tangible yet computer-controlled objects up to designing product prototypes—even if no or little programming experience is present.

PhidgetLab was successfully applied in a setting where pupils were confronted with prototyping tasks that they had to address using the Design Thinking [9] methodology. The user study we conducted in this setting revealed that PhidgetLab is very beneficial when users with little programming experience are expected to build hardware prototypes with programmatically controlled behaviour.

In the next section, we briefly introduce Phidgets, Squeak, Etoys, and Design Thinking. In Sec. 3, we describe the architecture of PhidgetLab. The setting of the user study we performed is described in Sec. 4, and the evaluation results are summarised in Sec. 5. Sec. 6 briefly discusses related work, and Sec. 7 summarises the paper and outlines future work.

[1]`www.hpi.uni-potsdam.de/swa/projects/phidgetlab/`



Car script1 paused

```
Test Car color sees color
  Yes
  No Car turn by 160 + random ( 40 )
Car forward by 5
Car turn by 1
Car bounce silence
```

Figure 1: An Etoys car with a script letting it drive and bounce off obstacles.

2. BACKGROUND

Squeak[2] [8] is an implementation of the Smalltalk programming language [6]. Smalltalk is purely object-oriented and dynamically typed. It allows for rapid prototyping in a very natural way, as software development is actually the manipulation of live objects, instead of editing source files, having them compiled, and debugging.

The Etoys[3] environment [1] is implemented on top of Squeak. It targets programming beginners—children and pupils, for instance. The concept of Etoys is to first let people design objects, e. g., by painting a car, and then adding scripts to them that control their behaviour. Scripts are however not represented in source code, but rather as *tile scripts* composed of drag-and-droppable tile components representing various kinds of behaviour and data (see Fig. 1 for an example). The advantage of this approach is that (albeit virtual) objects are manipulated directly, and that the effects of scripts are immediately visible.

Phidgets[4] are standardised electronic hardware components that can be connected to a PC via USB. Available sensor components include, e. g., sliders, light and temperature sensors, and 3-axis accelerometers (see Fig. 2). As actors, e. g., servo and step motors exist. More sophisticated Phidgets include 2-line LCDs and RFID readers/writers. Phidgets are programmable in several languages. Low-level access to them is achieved via a C library that is available under the terms and conditions of the GNU LGPL[5].

Design Thinking [9] is a mind-set and methodology for creating innovative solutions for particular needs or problems. The Design Thinking process collects and combines best practices from several fields, such as elaborating user needs, developing creative ideas, and prototyping, testing, and refining solution candidates efficiently and effectively. As a creative rather than analytical approach to problem solving, it encourages diversity in ideas, defers judgement, and relies on interdisciplinary teams. The HPI School of Design Thinking[6] teaches a process including the following concrete steps: *understand* the problem by talking to stakeholders, *observe* the current situation, assume a *point of view* that helps identify the core problem, *ideate* (collect ideas whilst abstracting from the problem setting) to form a high-level reusable solution, provide a *prototype* solution, and *test* it with the customers. The process is highly interactive and iterative; it is usual to traverse these steps along various, possibly cyclic, paths.

[2]www.squeak.org
[3]www.squeakland.org
[4]www.phidgets.com
[5]www.gnu.org/copyleft/lesser.html
[6]www.hpi.uni-potsdam.de/d-school

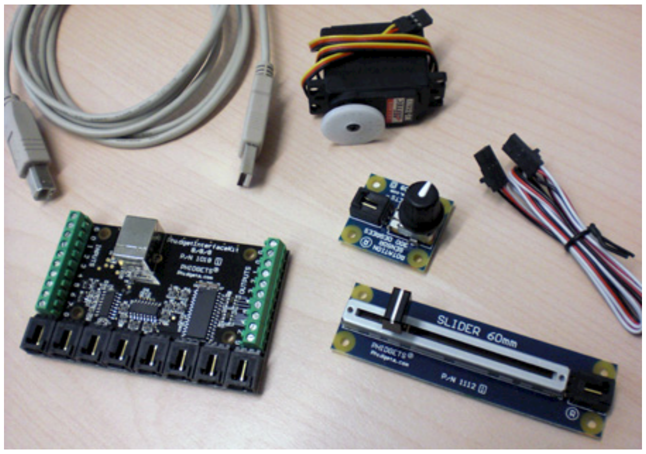


Figure 2: Phidget components: servo motor, interface kit for connecting components, control dial and slider, cables.

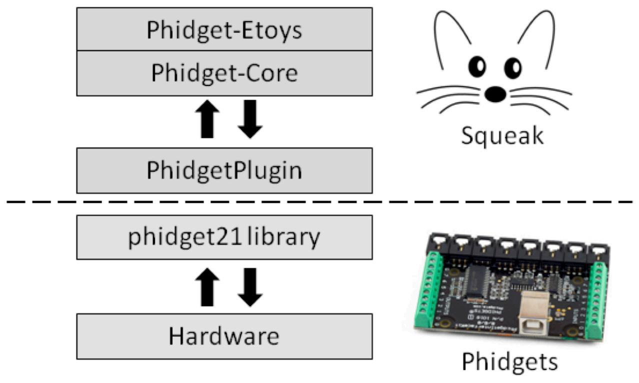


Figure 3: General Architecture of PhidgetLab

3. PHIDGETLAB

The PhidgetLab implementation consists of two parts. On the one hand, it features a Squeak API that enables developers to access Phidgets and use them in their applications. The foundation of the API is a plugin for the Squeak runtime environment that connects to the Phidget C API. On the other hand, Etoys was extended to enable its users to access and control Phidgets as Etoys objects. The general architecture of all components is shown in Fig. 3. We will elaborate on both parts below.

The core package provides the Phidget API for Squeak. Its classes and methods can be used to communicate with Phidgets directly from the Squeak environment. For instance, the value of the first sensor attached to a connected Phidget interface kit (`ifkit`) can be obtained by executing `(ifkit sensors at: 1) value`. To communicate with Phidgets, the manufacturer's programming API is used. A plugin for the Squeak virtual machine, compiled as a shared library, interfaces the core package with that API.

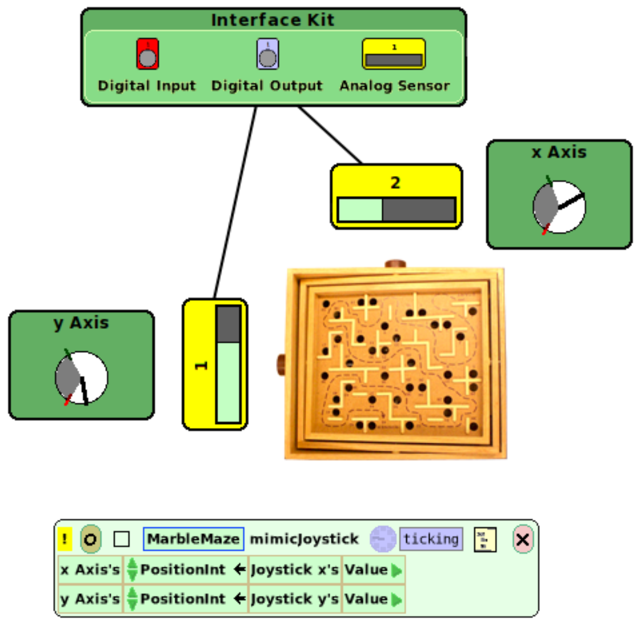


Figure 4: Sample Etoys script and visualisation for Phidgets.

The core API is relevant for people with a certain fluency in programming Squeak. They can make use of it when programming new applications using Phidgets. It organically adapts the official Phidget manufacturer's API to Squeak. The API was also expanded to provide some handy converted values in additon to "raw" sensor values.

Each Phidget component such as the interface kit or the servo motor has its own class, each of them derived from one common `Phidget` class that provides basic operations like creating Phidget objects, discarding them, and setting event handlers. This architecture allows to extend PhidgetLab and to support new types of Phidgets.

The "Phidget-Etoys" package integrates functionality provided by the core package with Etoys. Thus, our Etoys extension hides the complexity of the Phidget C library as well as our Smalltalk API. People without any prior programming knowledge can use PhidgetLab as every Phidget has an equivalent in Etoys.

Figure 4 shows an example that connects a joystick Phidget with two servo boards to control the x- and y-axis of a marble maze game[7]. At the top, the interface kit represents the hardware component that combines all actor and sensor Phidgets. Below it, two Etoys objects comprise the joystick axes followed by two other objects for the two servo motors. At the bottom, the Etoys script connects both actors and sensors together with only two lines of code. With this approach, users can simply perceive the Phidget's state, set values, or make the device part of an Etoys script to let it interact with other Etoys objects.

4. STUDY SETTING

PhidgetLab was evaluated in a one-week Design Thinking

[7]A demonstration video of the steered marble maze game can be found at `www.hpi.uni-potsdam.de/swa/projects/phidgetlab/`

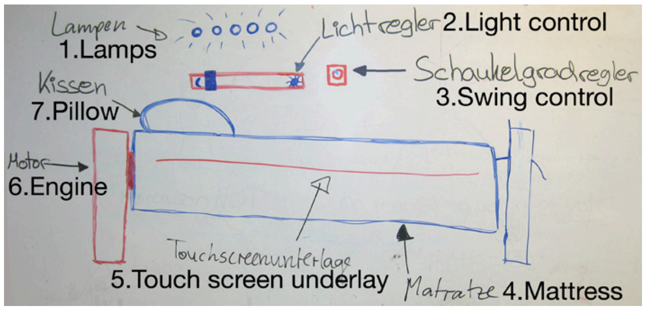


Figure 5: Conceptual design of the "Sleep Well Bed".

workshop with 22 pupils at the average age of 15 years. By following the Design Thinking process of understanding and observing, all participants identified problems of everyday life and built prototypes to solve them. Besides common Design Thinking tools, all prototypes were to be realised primarily with Etoys and PhidgetLab.

The group consisted of 8 girls and 14 boys between 13 and 17 years. Most of the workshop participants attended the 10th grade of a grammar school and had at most one year experience in topics related to computer science. Computer science education at German schools at this early stage typically encompasses basics such as using office tools. Thus, all participants had a certain experience with office applications, games or the Internet. Moreover, since about one year, 13 pupils were interested in programming in their spare time, with a focus on HTML and PHP. Only five used programming languages such as Python, Java or Pascal. Although 16 out of 22 pupils had many years of experience with electronics and controlling external hardware, this knowledge was mostly limited to slot car racing tracks, model railways or Lego Mindstorms. None of them knew Etoys or PhidgetLab. In summary, programming experience was very low in the pupils.

The pupils were divided into five groups of three to five. Each group was supervised by a student ensuring guidance with our tools and concepts. The workshop schedule comprised five days—in the first two days, several presentations and tutorials introduced Design Thinking, Etoys and PhidgetLab. Moreover, since the pupils did not know each other before, team building games were played. In the following two days, the five groups worked on their prototypes to solve their self-chosen problems. On the last day, the results were presented.

To illustrate an exemplary two days' work result, we describe the "Sleep Well Bed" project in more detail. A group of two girls and three boys considered sleeping troubles as their everyday problem and developed a sophisticated bed with automatically dimming lights and a swinging engine that stops when the user is fast asleep. Fig. 5 presents the original sketch—the result of the first day.

After the second day, the group was able to present an operational prototype of their "Sleep Well Bed" (cf. Fig. 6). It was constructed from Lego bricks, hidden Phidgets and several Etoys scripts controlling the former. The lamps (1) simulated by LEDs are dimmed manually with a slider (2) and automatically with a timer script. The swing engine (6), driven by a small servo motor, is controlled by a user preference, which is adjustable with the rotation sensor (3),

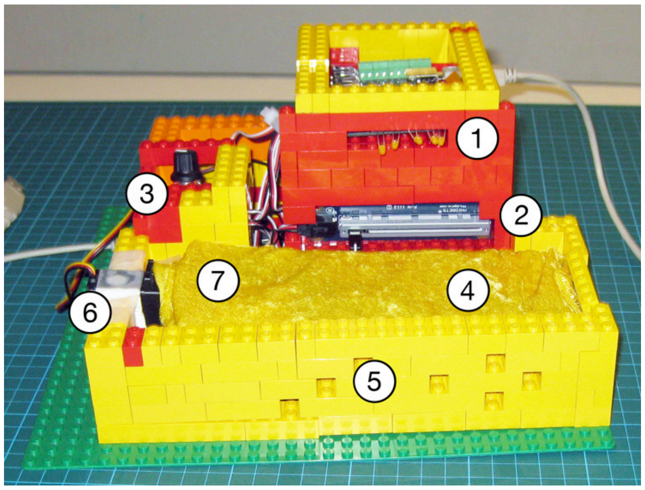


Figure 6: Final "Sleep Well Bed" prototype built with Lego bricks, Phidgets and Etoys (index numbers correspond to the sketch in Fig. 5).

and the user's activity during sleep, which is checked by a touch screen underlay (5) implemented using a force sensor. The bed itself (4,7) was built with simple cloth. All in all, the group demonstrated the feasibility of their idea and developed a prototype within two days.

5. EVALUATION

The evaluation was done in two parts—we first asked the pupils about their opinions on Etoys, PhidgetLab, and the workshop concept; we then asked their supervisors to estimate the pupils' motivation and knowledge level as well as the roles of all participants in their projects.

Tab. 1 (see next page) summarises the first part with the median values of all 22 opinions and a two-tailed sign test that compares the different answers between Etoys and PhidgetLab. Etoys was evaluated favourably, benefiting from its combination with Phidgets. Although most pupils liked PhidgetLab very much, both tools' levels of difficulty were only ranked satisfying. Furthermore, the sign test revealed a significant difference between the Etoys and PhidgetLab answers for the first two questions ("Did you like it?", and "Did you have fun working with it?"). Many pupils gave PhidgetLab preference over Etoys. However, the results for the last two questions ("Are you able to continue on your own?", and "How difficult was it for you?") are mostly equal for both tools. Hence, we consider further simplification of PhidgetLab important future work.

During the evaluation, we noticed that eight participants had previous knowledge with Lego Mindstorms so that they could answer the same questions in relation to Etoys and PhidgetLab. The latter is attested a slightly increased level of difficulty but six out of eight pupils score PhidgetLab higher than Lego Mindstorms (based on a two-tailed sign test). Moreover, we could recognise a tendency that PhidgetLab is "more fun". Apart from that, the comparison reveals that pupils who know alternative approaches rank PhidgetLab better than all participants together.

Since all projects were successful, it is not possible to ascribe any benefits to prior knowledge on Lego Mindstorms or electronics. Instead, we can conclude that PhidgetLab's barrier to entry is low enough to allow inexperienced pupils a quick sense of achievement.

The second part of our evaluation considers the pupils from the perspective of their supervisors. All primary project requirements were realised without any large problems; the groups even extended their ideas with additional requirements. We conclude that no one was overstrained by the workshop. During work, help was requested mostly amongst group members, but rarely from supervisors. Each group applied the Design Thinking process as intended with much communication and different roles ranging from programmers to artists. The motivation and ability to concentrate were continuously good. Supervisors estimated the application and understanding of Etoys and PhidgetLab approximately identical to the pupils' evaluation.

In summary, all pupils graded the complete workshop between 1 and 2.

One of PhidgetLab's goals is to arouse pupils' interest in computer science. Since, in Germany, computer science is mostly an optional subject at school, we asked our target audience whether they were, after the workshop, more inclined choose computer science as part of their A level or not. 7 out of 22 pupils affirmed they were willing to learn more about computers and programming, 12 were undecided, and the remaining 3 pupils explained that computer science was still not their favourite subject. From the answers we received, it is not possible to deduce how PhidgetLab in particular affected their decisions.

From the study, we conclude that PhidgetLab provides an easily accessible entry to computer science, without necessarily touching upon any of the deeper topics involved. Pupils get in touch with object-oriented abstraction through manipulating tangible objects (Phidgets) by message passing, and also through treating objects as encapsulations of state and behaviour in their virtual representation as Etoys widgets. While all the object-oriented principles are at work behind the scenes, pupils are not directly exposed to the involved complexity—PhidgetLab clearly supports the didactic approach that the spiral pattern suggests [4].

6. RELATED WORK

We restrict this discussion to systems related to Squeak and Etoys. A prominent example in this scope is Scratch[8] [10], which is implemented in Squeak, albeit without uncovering this fact. It provides an easily accessible multimedia scripting environment, and also supports interaction with the outside world by means of the "Pico Board" (formerly named "Scratch Board"), which can be connected to the PC via USB. The Pico Board features different sensors, but no output. Interaction with real-world objects is thus limited.

Physical Etoys[9] is a recently released Etoys extension that, like PhidgetLab, connects hardware components to Etoys scripts. Supported hardware includes the Arduino[10] [2] platform and Lego Mindstorms NXT. Physical Etoys complements PhidgetLab in that it does not support Phidgets.

NXTalk[11] [3] is a Smalltalk programming environment for

[8]`scratch.mit.edu`

[9]`http://tecnodacta.com.ar/gira/projects/physical-etoys/`

[10]`www.arduino.cc`

[11]`www.hpi.uni-potsdam.de/swa/projects/nxtalk`

	Etoys	PhidgetLab	Significance of Equality	#pupils favor Etoys	#pupils favor PhidgetLab
Did you like it?	2	2	0.8 %	0	8
Did you have fun working with it?	2	2	3.1 %	0	6
Are you able to continue on your own?	3	3	100 %	N/A	N/A
How difficult was it for you?	3	3	50 %	N/A	N/A

Table 1: Median values (1 = *very good* ... 5 = *unsatisfying*) and a two-tailed sign test of the pupils' evaluation between Etoys and PhidgetLab with the hypothesis that there is no difference in the central tendency.

the Lego Mindstorms NXT[12] platform. NXTalk applications are implemented in Squeak and then transferred to the dedicated Smalltalk run-time environment installed on the NXT. NXTalk does not yet feature Etoys accessibility but is developed in conjunction with PhidgetLab to eventually provide a uniform environment for advanced programming tasks.

Alice[13] [5] is a 3D authoring environment targeting programming beginners. Like Etoys and Scratch, it provides tile scripting to ease programming. Alice is implemented in Java, but was re-implemented in Squeak. Hardware connectivity is not provided.

7. SUMMARY AND FUTURE WORK

We have presented PhidgetLab, the result of connecting Etoys and Phidgets, which has shown to be a promising piece of software in a case study taken out with pupils inexperienced in programming. We continue to apply PhidgetLab in similar settings, and are also working on extended support for existing Phidgets as well as controls for Phidgets that are so far not supported. The shape of Etoys controls is an important topic of ongoing research, as it is our intent to make desired functionality available intuitively.

As mentioned in Sec. 6, PhidgetLab and NXTalk are two members of the same family. Part of our ongoing research is concerned with providing Etoys connectivity for interacting with physical objects on different scales of complexity, ranging from robots (NXT) over pre-assembled electronic components (Phidgets) to processor-controlled circuits (Arduino). Results from the work on Etoys/Arduino are likely to be integrated in this project. Eventually, the project will result in a complete suite covering Etoys-programmable real-world objects at different degrees of complexity.

Acknowledgements

We are grateful for the time and effort that the pupils participating in our user study have invested. We also thank the members of the "HPI Schülerclub" for organising the workshop in which the study took place. Finally, we thank Bastian Steinert for insights on Design Thinking and Stefan Hanenberg for his advice with the user study evaluation.

Part of this work was supported by the HPI—Stanford Design Thinking Research Program (HPDTRP).

8. REFERENCES

[1] B. J. Allen-Conn and K. Rose. *Powerful Ideas in the Classroom.* Viewpoints Research Institute, Inc., 2003.

[2] M. Banzi. *Getting Started with Arduino.* O'Reilly, 2008.

[3] M. Beck, M. Haupt, and R. Hirschfeld. NXTalk. Dynamic Object-Oriented Programming in a Constrained Environment. In *Proceedings of the International Workshop on Smalltalk Technologies.* ACM, 2010. To appear.

[4] J. Bergin, J. Eckstein, M. Manns, and E. Wallingford. Patterns for Gaining Different Perspectives. In *Proceedings of PLoP*, 2001.

[5] S. Cooper, W. Dann, and R. Pausch. Teaching objects-first in introductory computer science. In *SIGCSE '03: Proceedings of the 34th SIGCSE technical symposium on Computer science education*, pages 191–195. ACM, 2003.

[6] A. Goldberg and D. Robson. *Smalltalk-80: The Language and its Implementation.* Addison-Wesley, 1983.

[7] B. Hergenhahn and M. Olson. *An Introduction to Theories of Learning.* Prentice Hall, 1997.

[8] D. Ingalls, T. Kaehler, J. Maloney, S. Wallace, and A. Kay. Back to the future: the story of squeak, a practical smalltalk written in itself. In *Proceedings of the 12th ACM SIGPLAN Conference on Object-oriented Programming, Systems, Languages, and Applications*, pages 318–326. ACM, 1997.

[9] T. Kelley and J. Littman. *The Art of Innovation.* Profile Books, 2001.

[10] D. J. Malan and H. H. Leitner. Scratch for budding computer scientists. In *SIGCSE '07: Proceedings of the 38th SIGCSE technical symposium on Computer science education*, pages 223–227. ACM, 2007.

[12] `www.mindstorms.com`
[13] `http://www.alice.org/`