# Realization of a Platform-Independent Web-Application for Engineering Education

http://dx.doi.org/10.3991/ijac.v8i1.4419

H. Siemund, M. Kracht and H. Goebel
Helmut Schmidt University, Hamburg, Germany

***Abstract*—In this paper an approach to realizing a web-based and interactive learning environment for engineering education is presented which is runnable on conventional desktop computers as well as on modern mobile devices such as tablets or smartphones. The presented course program is intended to supplement traditional lectures and text books at the Helmut Schmidt University, Hamburg.**

***Index Terms*—E-Learning, Engineering Education, Google Web Toolkit, Mobile Learning, Web-Application.**

## I. INTRODUCTION

In order to support its lectures in electrical engineering, the Institute of Electronics at the Helmut Schmidt University, Hamburg, Germany, developed a web-based and interactive course program called S.m.i.L.E [1] (http://smile.unibw-hamburg.de). The acronym S.m.i.L.E stands for: "Studieren mit dem interaktiven Lehrbuch fuer Elektronik" = "Studying with the interactive Textbook for Electronics". This course program is a collection of HTML text pages, each being concerned with a particular subject of microelectronics. Within the HTML pages, embedded Java applets [2] are used as interactive animations in order to illustrate complex physical effects and dynamic processes. Due to its benefits, the learning environment S.m.i.L.E was published together with two text books [3, 4].

During the planning phase of S.m.i.L.E in the late 1990s, the authors decided to implement their interactive animations as Java applets, due to the platform independence of the Java programming language and the possibility to run the applets in a web browser. (In this context, the term "platform independence" means, that the compiled Java bytecode can be executed on any hardware platform / operating system (OS), on which a Java Runtime Environment (JRE) is installed). This concept proved successful for more than a decade, since there were JREs and Java-compatible web browsers available for virtually all common desktop computer platforms.

But in the last years, the situation has increasingly deteriorated with the rise of modern mobile devices running Android or iOS, to mention only the two most popular operating systems for smartphones [5]. At present, neither for Android nor for iOS a JRE/browser is obtainable which supports Java applets. A further aggravating factor is the high number of severe security issues concerning the Java Runtime Environment in recent times [6]. These have led to a growing scepticism towards Java applets and also to more restrictive Java security settings [7], and thus to a more complicated handling of the S.m.i.L.E program.

In order to solve the problem and make the S.m.i.L.E learning environment runnable on desktop computers as well as on mobile devices, different approaches are suitable: First, separate platform-specific native applications could be developed, but this procedure results in an inacceptable overhead in programming effort and maintenance. Second, cross-platform compilers like PhoneGap [8] or SenchaTouch [9] could be used to generate platform-specific versions from only one single code base. This approach in fact reduces the programming effort. However, most of these cross-platform tools rely on the programming languages JavaScript, C++, Lua and Ruby [5] but not on Java, so that the existing large amount of Java source code for the S.m.i.L.E project can hardly be reused. In this report, a third approach, the usage of the Google Web Toolkit (GWT), which provides platform independence and reusability of Java code, is shown. In this paper, the new mobile course program S.m.i.L.E Mobile, its implementation, as well as some applications are presented after an introduction to the Google Web Toolkit.

## II. THE GOOGLE WEB TOOLKIT

### A. Overview

The Google Web Toolkit [10] is a free and open source development toolkit for building high-performance browser-based web applications, following the "write once, run anywhere" approach. The core component of the toolkit is the GWT-compiler, which actually is a Java-to-JavaScript converter. Additionally, a set of core Java application programming interfaces (APIs) and user interface (UI) components as well as powerful debugging- and optimizer tools are provided [11, 12]. GWT allows the programmer to write complex web applications, both client- and (optional) server-side, entirely in Java. The client-side Java sources are compiled into highly optimized JavaScript code which runs in all major browsers, including mobile browsers for Android and iOS. Compatibility problems arising from different JavaScript implementations between different browsers are circumvented by generating multiple permutations of the JavaScript code, one unique code version for each supported browser (see Fig. 1).

At runtime, after loading the main HTML page, the so-called bootstrap process of the application automatically performs a browser detection and thereafter only loads the JavaScript code version corresponding to the users browser. In addition to browser detection, it is also possible to generate customized code permutations for different languages (internationalization).

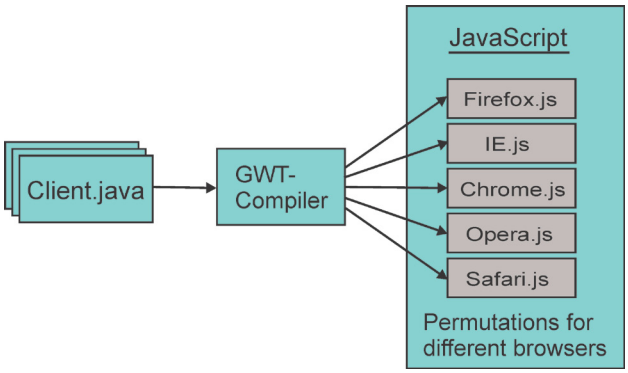


Figure 1. The client-side Java sources of the application are compiled into multiple JavaScript code permutations, one specific version for each supported browser

In order to support e.g. 5 browsers and 4 languages, in total 5 x 4 = 20 code permutations are generated by the GWT-compiler.

The optional server-side Java code is compiled into a server applet (servlet), however, other implementations of the server are also possible. The way of communication between the client- and server-side of the application depends on the server infrastructure, more details about GWT-provided communication protocols can be found in [10].

### *B. Benefits and drawbacks*

Besides the features described above, the following benefits of GWT are worth to mention:

- No browser plugin is required to execute GWT-applications, since all modern browsers support JavaScript.
- There is no need for the developer to consider browser compatibility, because GWT produces browser-specific optimized JavaScript code.
- No programming in the weakly-typed and thus error-prone language JavaScript is necessary at all. Instead, the strongly-typed and robust language Java can be used for the development of the entire GWT-application.
- GWT-Plugins are available for integrated development environments (IDE) like Eclipse [13] or NetBeans [14], so that the programmer can benefit from comfortable project management, code completion, refactoring- and build-tools. Since GWT-applications can be debugged in the Java language (before translation to JavaScript), the programmer can also take advantage of the IDE's effective debugging capabilities.
- The GWT-project is open source and is actively being developed and maintained.

The above-mentioned advantages of GWT, however, are coupled with a few potential drawbacks:

- GWT is based entirely on Java. There is no choice for other languages like C, C++, Python or Ruby, as is the case e.g. with the Code-to-JavaScript compiler Emscripten [15].
- Not all Java methods and APIs are implemented in GWT, and there are also a few differences between GWT and core Java language syntax and semantics (see [10] for details). Additionally, the multithreading capability of Java cannot be used in GWT, since JavaScript interpreters are single-threaded.
- Web standards, browsers and JavaScript are continuously evolving. In order to take advantage of new changes or to eliminate compatibility problems, a GWT-application should regularly be recompiled against an updated GWT-version. This leads to a strong dependence on the GWT-project, which currently is freely available under the Apache 2.0 open source license [16], but there is no guarantee that the GWT licensing model remains unchanged in the future.

## III. THE S.M.I.L.E MOBILE LEARNING ENVIRONMENT

For reasons explained in chapter I, the in total 57 Java applets of the S.m.i.L.E project are presently being converted into (pure client-side) GWT-applications. The most cumbersome task is to port the code for the UI components (widgets), because class names, methods and event handling mechanisms of the GWT widgets differ from those of the Abstract Window Toolkit (AWT) widgets [17] which have been utilized for the S.m.i.L.E applets. The remaining code, however, can easily be reused.

Although the conversion process is still at an early stage, a preliminary version of the S.m.i.L.E Mobile learning environment (currently in german language only) can already be accessed under the following link: http://smilemobile.unibw-hamburg.de. Upon entering, the visitor will see the main page for navigation within the course program. Each selectable document consists of an interactive GWT-animation and an accompanying explanation text (which appears after clicking the "Info" button below the animation). Fig. 2 shows as an example one of the animations running on an Android smartphone, together with one of the text books [3] referring to the S.m.i.L.E course program.

The shown animation illustrates the behavior of charge carriers in a semiconductor in dependence of the lattice temperature. In contrast to the text book, which can only use text and static graphics to explain the physical behavior, the animation allows the user to change variables as for example the temperature of the semiconductor and thus directly observe the resulting physical effects. For future editions of the text book, QR-codes containing links (see Fig. 2) are planned, which will facilitate and accelerate the start of the appropriate animations.

The new S.m.i.L.E Mobile learning environment has successfully been tested on the following device/browser combinations without any compatibility problems:

- Smartphone and tablet running Android OS with Chrome, Firefox and Dolphin browser,
- iPad running iOS with Chrome and Safari browser,
- Personal Computer (PC) running Windows 7/8 with Firefox, Internet Explorer and Chrome.

As two future projects, WSPICE, a web-based circuit-simulation environment with graphical schematics editor [18] as well as a drift-diffusion device simulator will also be ported into GWT-applications in order to further improve the teaching portfolio of the Institute of Electronics at the Helmut Schmidt University, Hamburg.

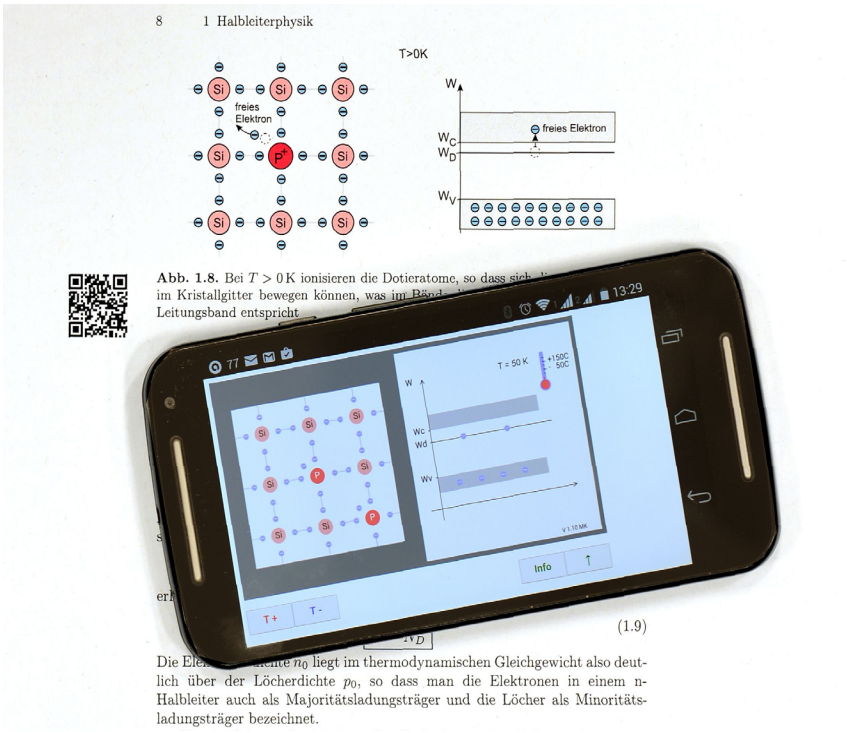


Figure 2. S.m.i.L.E GWT-animation running on an Android smartphone and the corresponding section of the text book. The animation illustrates the complex behavior of charge carriers in a semiconductor, depending on the user-adjustable temperature T.

## REFERENCES

[1] H. Siemund and H. Goebel, "A Web Assisted Electronics Course Using the S.m.i.L.E Program," *International Journal of Engineering Education, IJEE,* vol. 18, no. 6, pp. 736-744, 2002.

[2] Oracle Corp., "Java Homepage," [Online]. Available: http://www.java.com/en/. [Accessed 23 Jan. 2015].

[3] H. Goebel, Einführung in die Halbleiter-Schaltungstechnik, Berlin, Heidelberg: Springer, 2014. http://dx.doi.org/10.1007/978-3-642-53869-8

[4] H. Goebel and H. Siemund, Uebungsaufgaben zur Halbleiter-Schaltungstechnik, Berlin, Heidelberg: Springer, 2014. http://dx.doi.org/10.1007/978-3-642-53903-9

[5] P. Klima and S. Selinger, "Towards Platform Independence of Mobile Applications," in *Computer Aided Systems Theory - EUROCAST 2013*, Berlin Heidelberg, Springer, 2013, pp. 442-449. http://dx.doi.org/10.1007/978-3-642-53862-9_56

[6] Cisco Systems, Inc., "Cisco 2014 Annual Security Report," [Online]. Available: http://www.cisco.com/web/offer/gist_ty2_asset/Cisco_2014_ASR.pdf. [Accessed 23 Jan. 2015].

[7] D. Goodin, "Ars Technica," [Online]. Available: http://arstechnica.com/security/2013/01/firefox-to-block-content-based-on-java-reader-and-silverlight/. [Accessed 23 Jan. 2015].

[8] Adobe Ltd., "Adobe PhoneGap Homepage," [Online]. Available: http://phonegap.com/. [Accessed 23 Jan. 2015].

[9] Sencha Labs, "Sencha Touch Product Page," [Online]. Available: http://www.sencha.com/products/touch/. [Accessed 23 Jan. 2015].

[10] Google, Inc., "GWT Homepage," [Online]. Available: http://www.gwtproject.org/. [Accessed 23 Jan. 2015].

[11] E. Burnette, Google Web Toolkit - Taking the pain out of Ajax, Dallas: The Pragmatic Bookshelf, 2006.

[12] V. Gupta, Accelerated GWT, New York: Apress, 2008.

[13] Eclipse Foundation, Inc., "Eclipse Homepage," [Online]. Available: https://eclipse.org/. [Accessed 23 Jan. 2015].

[14] Oracle Corp., "NetBeans Homepage," [Online]. Available: https://netbeans.org/. [Accessed 23 Jan. 2015].

[15] Emscripten community, "Emscripten Homepage," [Online]. Available: http://kripken.github.io/emscripten-site/. [Accessed 23 Jan. 2015].

[16] Apache Software Foundation, "Apache License (Version 2.0)," [Online]. Available: http://www.apache.org/licenses/LICENSE-2.0. [Accessed 22 Jan. 2015].

[17] AgileEngine, LLC, "Bringing Swing applications into AJAX world with GWT and AjaxSwing," [Online]. Available: http://www.creamtec.com/products/ajaxswing/solutions/swing_to_GWT_and_AJAX.html. [Accessed 23 Jan. 2015].

[18] H. Goebel and H. Siemund, "WSPICE: A web-based SPICE circuit-simulation environment with schematic editor," in *International Conference on Interactive Computer Aided Learning*, Villach, Austria, 2004.

## AUTHORS

**Henning Siemund** is research assistant and is with the Institute of Electronics at the Helmut Schmidt University, Hamburg, Germany (e-mail: henning.siemund@hsu-hh.de).

**Michael Kracht** is certified technician and programmer at the Institute of Electronics at the Helmut Schmidt University, Hamburg, Germany (e-mail: michael.kracht@hsu-hh.de).

**Holger Goebel** is Professor in Electronics and is with the Institute of Electronics at the Helmut Schmidt University, Hamburg, Germany (e-mail: holger.goebel@hsu-hh.de).

Submitted 28 January 2015. Published as resubmitted by the authors 10 March 2015.