Using Intelligent Tutors to Teach Students How APIs are Used for Software Engineering in Practice

Vasanth Krishnamoorthy, Bharatwaj Appasamy, Christopher Scaffidi, Member, IEEE

Abstract—Computer science courses typically incorporate integrated training in software engineering, which includes learning how to reuse existing code libraries in new programs. This commonly presents a need to use the libraries' application programming interfaces (APIs) during project-based learning activities. The students learn these APIs with little support from the instructor. Instead, they primarily learn from whatever code examples they find on the web—a scattershot approach that is slow, frustrating and sometimes fruitless. Preliminary work investigated an improved approach that provides students with intelligent API tutors. This interactive educational content walks students through code examples selectively harvested from the web. This approach increased students' knowledge of API method names, but a key pedagogical challenge remained: students still lacked an understanding of common problems and surprises that they might encounter when using APIs in practice. Perhaps even more crucially, the prior statistical study provided no information about how students would perceive and accept the intelligent tutor approach, when given a choice.

Therefore, the current paper presents an enhanced approach that augments intelligent tutors with selected information from online FAQs and online open source code, thereby providing more explanation and context about how to use APIs in practice. A qualitative study assessed student perceptions of the system's usability as well as suggestions for improvement, revealing that 100% of students considered the approach to be useful. A key implication is that providing information from online FAQs, related articles, and open source code could facilitate computer science students' learning how to use APIs.

Index Terms—Application programming interface, computer science education, electronic learning, software engineering, software libraries

The authors are with the School of Electrical Engineering and Computer Science, Oregon State University (krishnav@onid.orst.edu, appasamb@onid.orst.edu, scaffidc@onid.orst.edu)

I. INTRODUCTION

An inherent part of computer science education is learning to use application programming interfaces (APIs), which are the means by which software engineers combine existing code libraries into new programs. Reuse of libraries is a software engineering skill because reuse enables programmers to create more functionality in less time than they could without reuse. Simultaneously, reuse of libraries is a computer science skill because library API embodies an *abstraction*—a concept that can be applied in other contexts [1], [2]. For example, Java and .NET provide APIs for using important data structures (e.g., hash tables) and networking operations (e.g., opening sockets). Every API that students master provides one more abstraction for the intellectual toolboxes that they apply when programming.

Lectures and textbooks provide students with guidance about APIs that are central to a given course's focus, but computer science courses often present students with the need to learn other APIs. This frequently happens when students implement computer programs as "projects" for a course or for a senior capstone. For example, different students may need to use specialized APIs for displaying advanced multimedia, implementing device drivers, controlling mobile phones, or creating telephone voice-response systems [3], [4]. Students work on projects outside class, so instructors currently have no means of providing direct one-on-one assistance as students learn to use specialized APIs. Instead, students mainly learn to use APIs by searching the web for sample code snippets [5]. These snippets are of uneven quality, and they are not organized into any sort of structure that guides learning across multiple online resources.

Preliminary work presented a new approach that enables expert API users to harvest code snippets from Web pages and package them into intelligent API tutors published to an online repository [6], [7]. (Intelligent tutors are interactive educational materials that teach how to perform tasks, and that tailor themselves to each learner's progress [8].) Compared to textbooks, this approach helped students in a study to learn the names of API methods more quickly, but the approach also left them with little understanding of common surprises encountered with using APIs in practice. Further, the prior study provided no insight into how students would perceive, and choose to use, the system if given a choice.

The current paper presents an enhancement to this approach, by *broadening the range of online API-related resources integrated into educational content* and by *evaluating how students perceived the overall approach*. A qualitative study assessed student perceptions of the system's usability as well as suggestions for improvement, revealing that 100% of students considered the approach to be useful.

II. PEDAGOGICAL CHALLENGES OF HELPING PEOPLE LEARN TO USE APIS

A library is a set of functions, methods, and classes implemented as software that is meant to be reused in multiple applications via its Application Programming Interfaces (APIs). For example, the Java API includes over 34,000 methods and classes [5]. Additional libraries are available for download (e.g., from sourceforge.net).

Reuse of libraries—a core competency of software engineering—is a thoroughly integrated aspect of computer science courses, particularly in intermediate and capstone-level courses. Such courses often use a project-based approach where students are organized into teams, each of which constructs a different piece of software [9]-[11]. Students in these classes are far beyond learning to use programming primitives such as loops and conditions; instead, they must figure out how to combine APIs to create new software. This is a challenge; researchers have identified many ways in which misunderstandings about APIs can interfere with programming [2], [12] and reduce productivity by as much as a factor of six [13]-[15].

Helping students to perform these projects is extremely difficult because different student projects can require using very different APIs [4]. No student could master *every* API—nor is there any need to do so, since many APIs are outside the specialized interests of each student. Yet similarly, no *instructor* leading such a course could hope to master and teach the full range of APIs required for all of the projects (except perhaps over the course of many years). Consequently, each student team currently must learn specialized libraries with minimal instructional support. *This issue—that the number of APIs greatly surpasses the knowledge of each instructor—was called "the most serious problem facing instructors who try to teach Java" by a task force of the Association for Computing Machinery (ACM)* [1].

Studies show that programmers who struggle to learn an API, either in school or on the job, usually respond by searching the web for code snippets showing how to use the API, then copying and pasting the code into their own programs [5], [16]. Increasingly powerful tools can automate the process of finding, copying, and pasting sample snippets [5], [17]-[21]. These tools do an effective job of retrieving snippets for a given topic and finding the most common way in which a particular API is used for a task. Yet this search-copy-paste process strips the snippets of any "comments, context, and explanatory prose" that had been available to explain *why* certain API calls are needed [16]. The result is that people using this approach often lack a good understanding of how to use the API [22]. Consequently, they trust code that has errors [16], modify code in ways that add errors [23], and create still more errors by inappropriately combining APIs [24], [25]. Thus, effective approaches are needed to help students to obtain code samples and to learn from them.

III. PRELIMINARY WORK: BASIC INTELLIGENT API TUTORS

Preliminary work developed an approach aimed at helping people learn from intelligent tutors how to use APIs [6], [7]. The preliminary prototype helped expert API users to harvest code snippets that illustrate correct ways to combine methods and classes from libraries, with hints and other annotations provided by the expert library users to guide learning. An empirical study showed that the preliminary prototype could be used to create intelligent tutors to support many API-learning tasks, and that the resulting intelligent tutors helped students to learn API method names in less time, compared to textbooks.

A. Background on Intelligent Tutors

An intelligent tutor is a form of interactive instructional material that trains a student in how to perform procedural tasks, and that tailors itself to the student's progress [8]. An intelligent tutor typically contains a set of screens or windows, each of which provides either a problem-solving example or a quiz containing a problem. The examples, provided by a human expert when the intelligent tutor is created, are selected to display key aspects of what the student needs to understand. Quizzes then pinpoint areas that the student does not yet understand and which still need reinforcement through more examples and explanation. Typically, the first few quizzes and examples are relatively simple or provide explicit instructions about how to complete difficult parts, while later quizzes and examples become increasingly difficult. Intelligent tutors have proven effective in teaching about humanities [26], science [27], math [28], and introductory programming concepts, such as loops and lists [28]-[31].

B. Intelligent API Tutors

Preliminary work yielded a prototype tool suite called ApiTutors that helps expert programmers harvest code snippets from Web pages to create intelligent tutors [7]. Such an “expert” could be an instructor, teaching assistant, or professional programmer experienced with using a particular library. There are five phases in the process of creating an intelligent API tutor (see [6] for details of each phase). First, the expert enters keywords such as “connect to database”. Second, ApiTutors queries search engines for a list of relevant web pages. Third, ApiTutors retrieves these pages, then extracts and intelligently selects a set of code examples from those pages. Fourth, the expert uses ApiTutors to organize the examples into screens that comprise a new intelligent tutor. Within the intelligent tutor, some examples are typically used for teaching, while others are used for “fill-in-the-blank” programming quizzes to assess student learning. Fifth, ApiTutors publishes the intelligent API tutor to a repository website for students to view and use during project-based learning. The prototype supports Java, with C++ and .NET support underway. (The Java-specific repository is called “jTutors”.)

C. Empirical Study of Usefulness for Creating Training Materials

An empirical study was conducted to investigate whether the preliminary prototype could be used to generate intelligent tutors for common learning tasks [6]. The tool was tested on 16 typical queries that had been posted by programmers to the Mica search engine (which retrieves code examples from the web but does not generate intelligent tutors) [5]. The prototype was considered effective on a query if the prototype identified at least three snippets meeting specific inclusion criteria in terms of relevant content and usable length. The study showed the prototype could successfully generate intelligent tutors for 88% of queries, indicating that the approach could be applied to many common API-related queries.

D. Empirical Study of Usefulness in Helping Students Learn API Names

A second empirical study investigated to what extent intelligent tutors improved students' ability to remember the names of API methods [6], [7]. Ten computer science students were each assigned two randomly-selected APIs from a set of five covered in undergraduate textbooks (JDBC, StringBuffer, HashMap, FileWriter, and DocumentBuilder). These APIs were selected because they appear in many computer science courses that have an integrated software engineering component (such as database, data structures, and networking courses, all of which may have integrated programming assignments). Each student learned one API with a textbook and the other API with an intelligent tutor (or vice versa, with the order randomly determined). For each API, the student had to complete a three-question fill-in-the-blank quiz on paper (with each blank corresponding to a method name), similar to what might be expected as part of a midterm or final exam. Averaged over all APIs, students scored

93% on quizzes after ApiTutors-based training and 45% after textbook-based training. Moreover, students learned more quickly with the ApiTutors training, since they studied for an average of ten minutes before choosing to take the test, but they consistently used the full 15 minutes allowed during the textbook training mode. Consequently, compared to textbook-based training, ApiTutors-based training enabled students to learn more API method names in less time.

This study also solicited informal feedback from participants. Several expressed reservations that, although the intelligent tutor enabled them to complete the quiz, the intelligent tutor still left them unsure about how to use the APIs in practice. This concern is discussed further below, as are attempts to address this unmet need in the current work.

IV. Helping students learn how APIs are used in practice

Given that many computer science undergraduates ultimately pursue careers in software engineering, a key challenge is preparing them to create software programs as professional engineers. This includes helping them to learn how to use abstractions embodied by APIs that they can later apply—not just to learn the names of API methods, as in the preliminary intelligent API tutors approach, but also to understand how they would use those APIs in practice.

Therefore, the preliminary approach has now been extended to encompass a broader range of information about how to use APIs, with particular emphasis on providing students with information about common challenges with using APIs as well as information about how APIs are actually used by professional software engineers in the context of larger applications. This augmented information is packaged with the intelligent tutor to provide an enhanced API tutor that can be published to the repository for use by students during project-based learning.

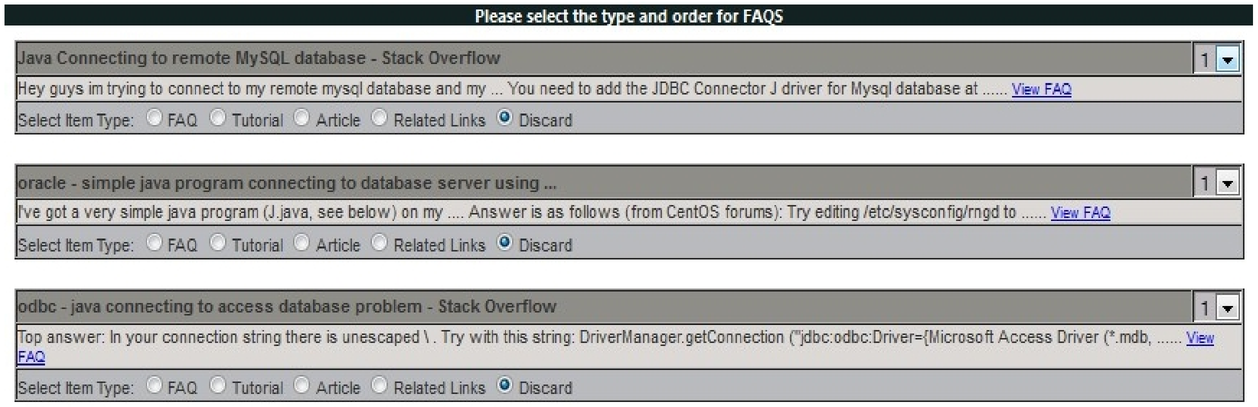
A. Pedagogical Challenge: Students Need to Learn How to Use APIs in Practice

The informal student comments after the preliminary studies (Section III.D) were consistent with prior research indicating that using an API effectively requires more than just remembering the names of API methods. One study explored challenges encountered by students who were learning to write programs with Visual Basic.NET, finding that students often struggled with deciding which APIs to use, anticipating what effects APIs will have when provided with different parameters, and dealing with surprises that result from combining APIs in inappropriate ways [24]. Studies have also highlighted the important role that conceptual knowledge plays in helping people to use APIs effectively [32]; for example, participants in one study were unable to use Bluetooth APIs effectively because they did not understand the overall process of establishing a Bluetooth connection [32]. Participants also struggled to actually obtain this information—in many cases, they were not even able to find relevant websites, let alone locate and learn the necessary information within those websites.

B. New Educational Content: Enhanced API Tutors

Preliminary work demonstrated that student learning of API methods could be increased if a *teacher gathered together useful online resources*; a similar approach was therefore taken to support student learning of how APIs are used in practice. The ApiTutors tool was extended so that after an expert programmer creates an intelligent tutor from code examples, the same expert can then gather together other resources containing *information about common challenges encountered with APIs* as well as *information about how APIs are used in the context of real applications.*

In this enhanced approach, information about common challenges is gathered from online FAQs and other related articles. Based on related work about where successful programmers go to find this information [33], five popular websites were identified as examples to explore the utility of gathering information from FAQs and related articles (stackoverflow.com, dzone.com, java2s.com, jguru.com, and javablogs.com). The ApiTutors tool was extended so that the expert programmer's query (provided as in Section III.B) is sent to each of these web sites (or to Google, which can perform a site-specific search). The first five results are presented to the expert, who places each link into one of several pre-defined categories ("FAQ", "Tutorial", "Other Article", and "Other Related Links"), or else discarded if of no use (Fig. 1). The teacher can also sort the links within each category. Along with each link, the expert can provide annotations and/or content taken from the corresponding FAQ or other related articles. (Copying a small amount of text in this manner is expected to be legal under fair use provisions for noncommercial instruction.) The FAQs and related articles typically include questions about common problems that professional software engineers encounter when trying to use the APIs, as well as solutions to those problems. The FAQs and related articles also explain how APIs can be used together, how they should not be used together, how to install the libraries that implement those APIs, and how to overcome a wide range of other common challenges related to APIs.



Please select the type and order for FAQS

Java Connecting to remote MySQL database - Stack Overflow 1
Hey guys im trying to connect to my remote mysql database and my ... You need to add the JDBC Connector J driver for Mysql database at ...... View FAQ
Select Item Type: FAQ Tutorial Article Related Links [✓ Discard]

oracle - simple java program connecting to database server using ... 1
I've got a very simple java program (J.java, see below) on my .... Answer is as follows (from CentOS forums): Try editing /etc/sysconfig/rngd to ...... View FAQ
Select Item Type: FAQ Tutorial Article Related Links [✓ Discard]

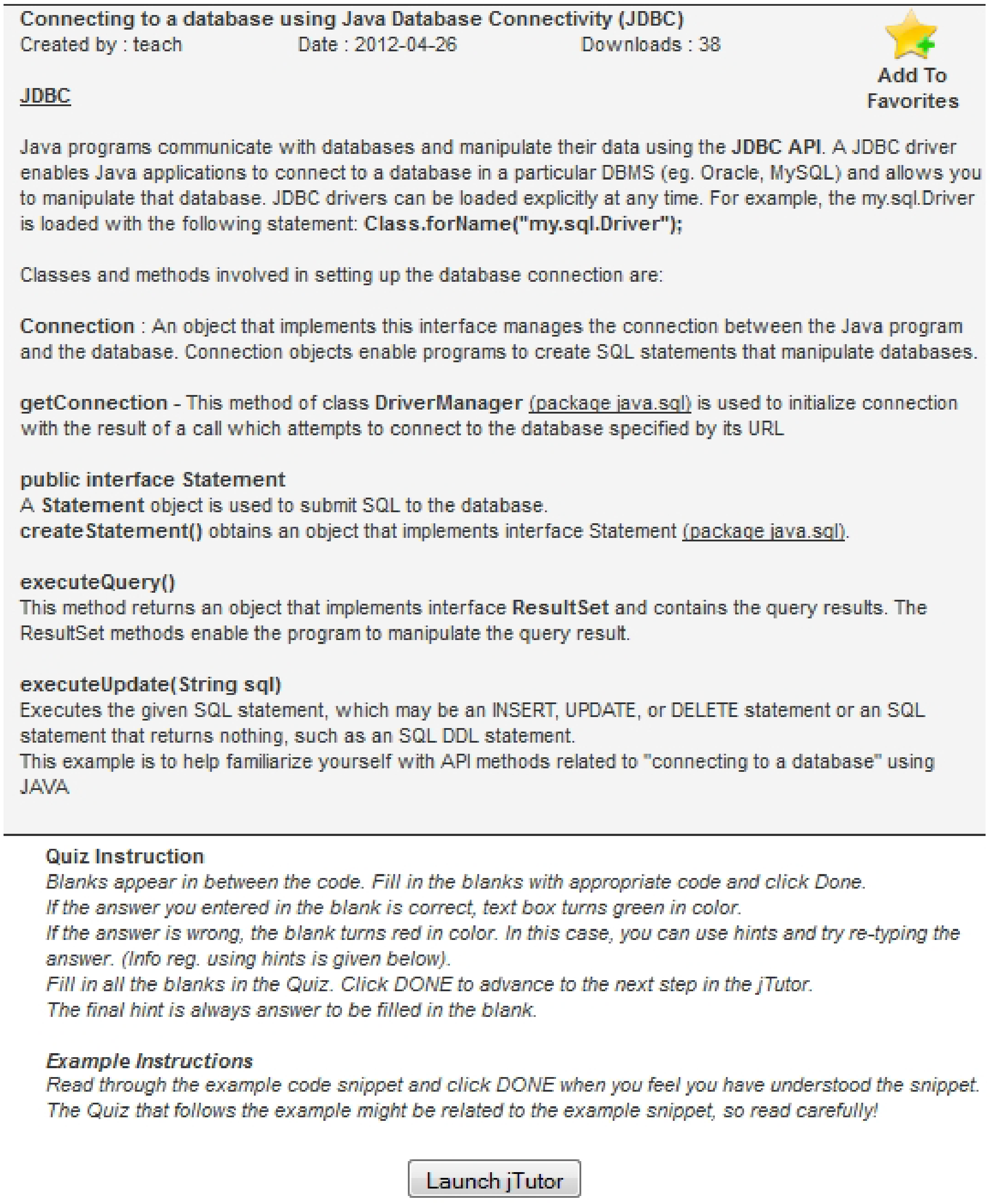
odbc - java connecting to access database problem - Stack Overflow 1
Top answer: In your connection string there is unescaped \ . Try with this string: DriverManager.getConnection ("jdbc:odbc:Driver={Microsoft Access Driver (*.mdb, ... View FAQ
Select Item Type: FAQ Tutorial Article Related Links [✓ Discard]

Fig. 1. Teacher can review, categorize, and sort the FAQ / related article links.

Information about how APIs are used in the context of real applications is gathered from open source industry code provided. Such code is available for free under a variety of licenses from many online repositories. ApiTutors now includes a feature for retrieving code examples from one repository (sourceforge.net) that demonstrate how a given API method was invoked in real applications. Expert programmers can also upload other code to ApiTutors that they have acquired in some other fashion; for example, they may have downloaded the code via a browser or written the code themselves. (The experts are responsible for ensuring that uploading the code is legal, which is expected to generally be the case with any open source code.) In general, the open source industry code available through these sources is much longer and more contextualized than the snippets included in the basic intelligent API tutors (Section III). This code shows how the APIs of interest integrate with other APIs, as well as how exceptions and errors raised by the APIs have been handled in real applications. Unlike with FAQs and related articles, the experts do not need to sort or otherwise organize the open source industry code; instead, a search tool is provided for students (below).

The list of FAQs and related articles, as well as the open source industry code, are packaged along with the basic intelligent tutor and stored on a repository that students can browse. After a student completes an intelligent API tutor (walking through the examples and quizzes as shown in

Fig. 2, Fig. 3, and Fig. 4), the API's corresponding FAQs and related articles are available for browsing, and the API's open source industry code is available for searching (Fig. 5 and Fig. 6). Students can enter one or more API methods or class names to retrieve industry code that mentions those methods or classes. Search results (Fig. 5) are sorted according to a joint weighting heuristic so the first search results are those that (a) most frequently mention the user-specified methods or classes, (b) mention all of the specified methods or classes (particularly those that most rarely appear in the open source industry code), and (c) are relatively short. Clicking on a link in the search results opens up the corresponding source code (in a prettified format) on the screen.



Connecting to a database using Java Database Connectivity (JDBC)
Created by : teach Date : 2012-04-26 Downloads : 38

JDBC

Java programs communicate with databases and manipulate their data using the **JDBC API**. A JDBC driver enables Java applications to connect to a database in a particular DBMS (eg. Oracle, MySQL) and allows you to manipulate that database. JDBC drivers can be loaded explicitly at any time. For example, the my.sql.Driver is loaded with the following statement: **Class.forName("my.sql.Driver");**

Classes and methods involved in setting up the database connection are:

Connection : An object that implements this interface manages the connection between the Java program and the database. Connection objects enable programs to create SQL statements that manipulate databases.

getConnection - This method of class **DriverManager** (package java.sql) is used to initialize connection with the result of a call which attempts to connect to the database specified by its URL

public interface Statement
A **Statement** object is used to submit SQL to the database.
createStatement() obtains an object that implements interface Statement (package java.sql).

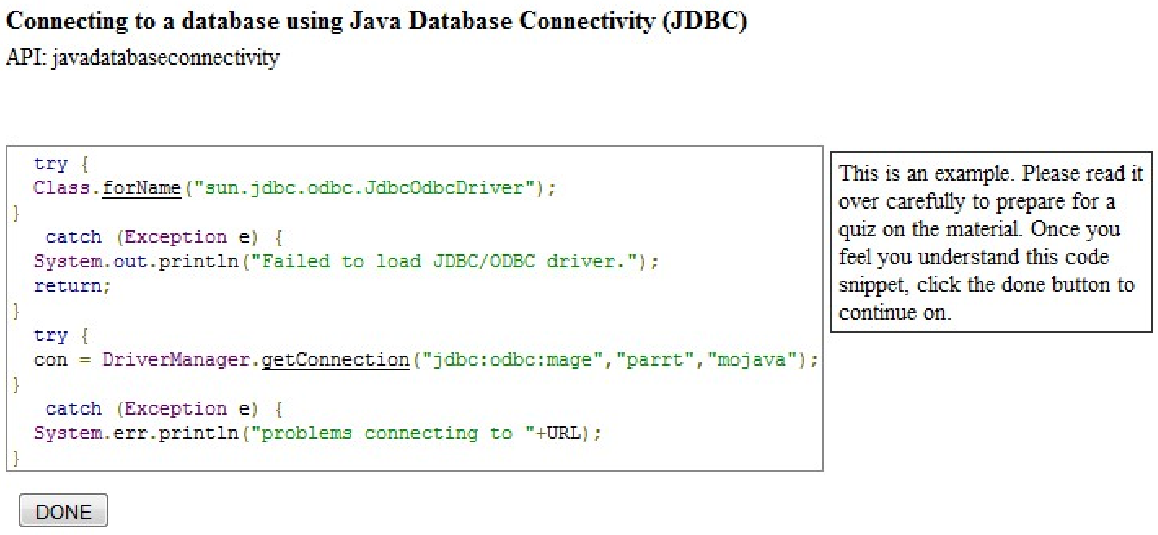
executeQuery()
This method returns an object that implements interface **ResultSet** and contains the query results. The ResultSet methods enable the program to manipulate the query result.

executeUpdate(String sql)
Executes the given SQL statement, which may be an INSERT, UPDATE, or DELETE statement or an SQL statement that returns nothing, such as an SQL DDL statement.
This example is to help familiarize yourself with API methods related to "connecting to a database" using JAVA

Quiz Instruction
Blanks appear in between the code. Fill in the blanks with appropriate code and click Done.
If the answer you entered in the blank is correct, text box turns green in color.
If the answer is wrong, the blank turns red in color. In this case, you can use hints and try re-typing the answer. (Info reg. using hints is given below).
Fill in all the blanks in the Quiz. Click DONE to advance to the next step in the jTutor.
The final hint is always answer to be filled in the blank.

Example Instructions
Read through the example code snippet and click DONE when you feel you have understood the snippet.
The Quiz that follows the example might be related to the example snippet, so read carefully!

Fig. 2. The learner can view a summary of the API, as entered by the expert.



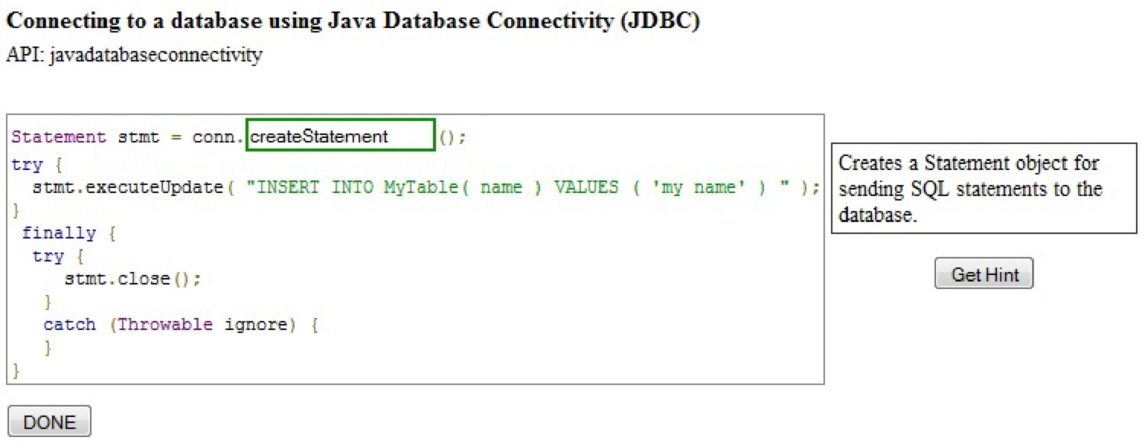
Connecting to a database using Java Database Connectivity (JDBC)

API: javadatabaseconnectivity

```
  try {
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
}
   catch (Exception e) {
  System.out.println("Failed to load JDBC/ODBC driver.");
  return;
}
  try {
  con = DriverManager.getConnection("jdbc:odbc:mage","parrt","mojava");
}
   catch (Exception e) {
  System.err.println("problems connecting to "+URL);
}
```

This is an example. Please read it over carefully to prepare for a quiz on the material. Once you feel you understand this code snippet, click the done button to continue on.

Fig. 3. A quiz typically starts with a few examples showing how to use the API.



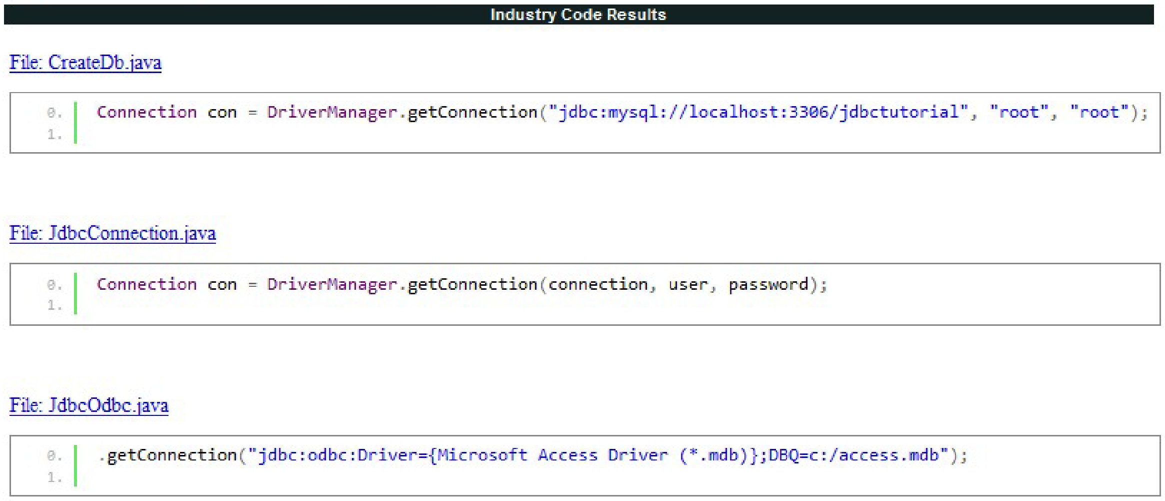
Connecting to a database using Java Database Connectivity (JDBC)

API: javadatabaseconnectivity

```
Statement stmt = conn.createStatement();
try {
  stmt.executeUpdate( "INSERT INTO MyTable( name ) VALUES ( 'my name' ) " );
}
 finally {
  try {
     stmt.close();
    }
   catch (Throwable ignore) {
    }
}
```

Creates a Statement object for sending SQL statements to the database.

Fig. 4. The intelligent tutor shows quizzes to assess learning, typically followed by harder examples and harder quizzes.


Industry Code Results

File: CreateDb.java

```
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbctutorial", "root", "root");
```

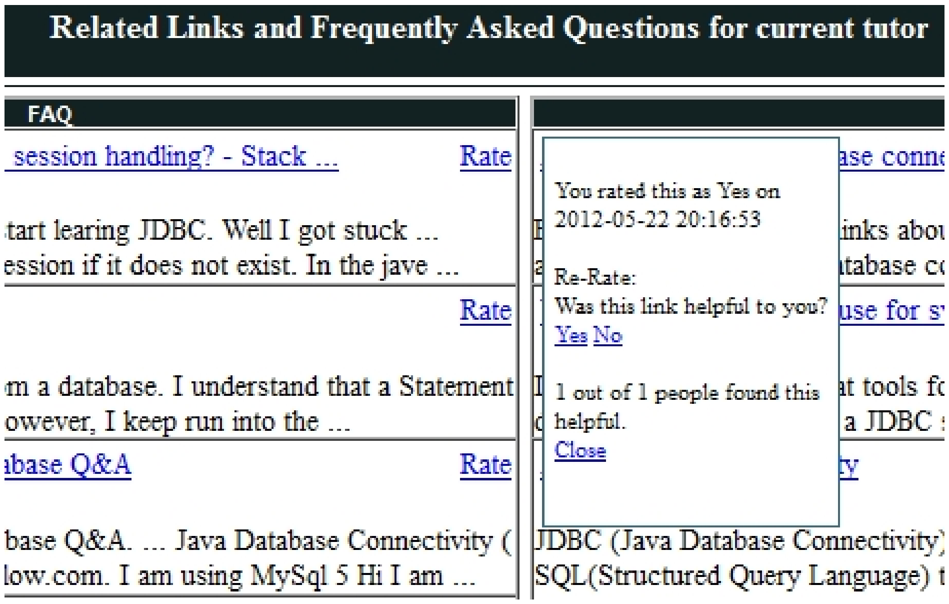
File: JdbcConnection.java

```
Connection con = DriverManager.getConnection(connection, user, password);
```

File: JdbcOdbc.java

```
.getConnection("jdbc:odbc:Driver={Microsoft Access Driver (*.mdb)};DBQ=c:/access.mdb");
```

Fig. 5. After the examples and quizzes, the learner can search and browse through industry code examples that use the API (search results shown here).


Related Links and Frequently Asked Questions for current tutor

FAQ

session handling? - Stack ... Rate

tart learing JDBC. Well I got stuck ...
ession if it does not exist. In the jave ...

Rate

m a database. I understand that a Statement
owever, I keep run into the ...

ıbase Q&A Rate

base Q&A. ... Java Database Connectivity (
low.com. I am using MySql 5 Hi I am ...

ase conne

inks abou
ıtabase co

use for s

at tools fo
a JDBC

ty

JDBC (Java Database Connectivity)
SQL(Structured Query Language) t

You rated this as Yes on 2012-05-22 20:16:53

Re-Rate:
Was this link helpful to you?
Yes No

1 out of 1 people found this helpful.
Close

Fig. 6. Finally, the learner can browse and search through FAQs and related articles for the API.

C. Using Enhanced API Tutors to Complement Formal Course Curricula

Students could rely on the repository of enhanced API tutors during their project-based learning activities. For example, teaching assistants could be tasked with creating enhanced API tutors for the classes that they help to instruct; students could then refer to the repository while working on their projects and send emails to a pool of teaching assistants to request additional materials when needed. Such a repository could be shared among multiple courses that have a somewhat overlapping focus (to provide a pool of teaching assistants); for example, some students might need to use database APIs for projects in a course on databases, while other students might need the same APIs for projects in their senior capstones. The repository would then gradually grow over multiple terms, incrementally providing more value to students as the years go by. There may also be the potential for multiple universities to share the repositories.

Eventually, repositories may contain so many valuable enhanced API tutors that professors could directly integrate the tutors into their formal course curricula. For example, a course on databases might include a unit where students (individually or in teams) work through enhanced API tutors that teach how to use APIs to create tables and establish foreign keys among the tables. The integration of FAQs and other related articles would explain the corresponding conceptual knowledge in the context of challenges commonly encountered with the APIs. Thus, enhanced API tutors eventually could supplement or even supplant textbook-based and lecture-based learning, provided that students find the enhanced API tutors approach to be just as effective as these alternate approaches.

V. Assessment: Qualitative evaluation of enhanced API tutors

Statistical measures of impact (as in the earlier study) are not necessarily enough to indicate that an educational intervention is ready for widespread testing. Consider an analogy to medical science: a single study showing statistical impacts of a new treatment would not be enough evidence to justify a clinical trial without also providing a qualitative investigation about how patients would accept and adhere to the treatment. Likewise, the preliminary statistical study [6], [7] describing statistical impacts of the new educational approach is not enough evidence to justify widespread testing. What is missing is an investigation of how students would respond to the approach.

Therefore, a qualitative laboratory study was conducted to assess student perceptions of the system's usability (including the new features that integrate information from FAQs and open source industry code), as well as to generate suggestions for improvement. This focus called for a qualitative study method rather than a quantitative method. This type of research is usually preferred for investigation of open-ended human phenomena, such as feelings and choices [34]. In such a study, participants can be given a variety of options, providing the researcher insight into what choices the participants prefer to make and the reasons for those choices. Such insights, in turn, can shed light on whether statistically significant results (as in the prior study) would likely persist even when people have a choice about whether to engage in a prescribed activity (such as using intelligent tutors). Due to the time-consuming nature of coding qualitative data, a qualitative study typically has a smaller sample than a statistical study (which normally involves dozens of participants).

For the current study, students performed several learning activities with enhanced API tutors and then were interviewed. These data were analyzed to identify recurring themes in students' responses, revealing that students considered these enhanced API tutors to be very useful.

A. Study Methodology

After a brief tutorial showing how to use intelligent API tutors, each participant selected one of five APIs that sounded interesting and worth learning. These five APIs were those used in the preliminary user study (Section III.D), for which they were selected because they appear in many computer science courses that have an integrated software engineering component. The participant completed a basic intelligent API tutor (constructed beforehand for the selected API). This basic intelligent API tutor code included examples from the web, quizzes about API method names, and hand-coded hints (as in the preliminary prototype).

Each participant was then given access to educational content of the enhanced API tutors. They were not required to read every piece of information (i.e., FAQ/related article or industry code segment). Instead, they could browse and search the content, then choose to read only the portions that seemed most useful. (In practice, all participants chose to use some but not all of the content.)

To prompt participants to consider using this content, they were asked to complete questions related to the APIs. Many of these questions could be answered based on information in the enhanced API tutors, while other questions could be answered using basic intelligent API tutors. The questions covered nearly the full range of Bloom's taxonomy of learning [35], from simply remembering APIs to evaluating tradeoffs among diffent APIs. (No questions were asked about creating new APIs, which would have corresponded to the highest level of Bloom's taxonomy.) For example, these questions asked about when to combine methods and the kinds of errors that could be encountered with the API. Other questions provided Java source code with API method names blanked out and asked participants to fill in the method names.

Finally, an interviewer asked participants to describe their experience, to identify what they liked, to suggest improvements, to discuss ApiTutors' usability, and to rate on a 5-point scale the usefulness of each part of ApiTutors (i.e., the basic intelligent tutor, the FAQs/related articles, and the open source industry code).

Each student participated in the study separately, alone in a room with the interviewer and a second observer who took handwritten notes. Screen capture was recorded during tasks above. Each participant received $20 for this one-hour study.

B. Qualitative Analysis Methodology

Qualitative study findings are often generated by an inductive process where detailed information obtained from the study is classified into general themes [36], [37]. A thematic analysis typically involves (1) reading the notes of the study and identifying recurring themes, (2) coding each participant's answers according to which themes are mentioned while referring to other data (such as screen capture video) as needed to aid in interpreting participants' statements, (3) independent recoding of the same answers by another researcher, and (4) comparing the two researchers' code assignments to assess validity. The process is repeated until the Intercoder Agreement Coefficient, defined as (Agreements) / (Agreements + Disagreements), exceeds a threshold of acceptability. The current study targeted a threshold of 80%, as recommended by Krippendorff [36] (though other researchers argue that 70% is often acceptable [37].)

C. Results

A total of 17 computer science students were recruited (mostly juniors). Analyzing their answers led to recurring themes and code assignments that yielded an acceptable level of agreement (87%). Themes mentioned by at least two participants are shown below (Table). Note that each person could mention more than one theme.

These answers revealed that 100% of participants liked at least one part of ApiTutors, with the most frequently-mentioned part being open source industry code (liked by ten of the 17 participants). The FAQ / related articles and basic intelligent API tutor were mentioned less often (by six and seven, respectively). Students' explanations of why they liked these parts tended to emphasize that the enhanced features showed usage in practice: they said the system "helps by showing method usages", "exposed what issues people commonly face," was "helpful with syntax and usage," and "provides examples of different usage".

These perceptions of usefulness were consistent with the ratings that participants gave later in the interview of the approach's three parts on a 5-point scale (from "not useful" to "very useful"). Participants rated the industry code feature as 4.53 on average, the tutorial experience as 4.29, and the FAQ / related articles as 4.24. These three statistics were not significantly different from one another (under two-tailed Mann-Whitney nonparametric U tests at $p<0.05$), but they highlight that all three parts of the approach were well-received, particularly the open source industry code.

More than half of participants (eight) suggested the basic intelligent API tutor should incorporate more detail about the API, perhaps by integrating material from FAQs into the intelligent tutor. For example, one participant suggested, “It would be good to have incremental descriptions about the API methods in the tutor.” Such integration would provide progressive detail about methods as the user encounters them. Two participants suggested providing a “dictionary” to look up information about methods.

Table I. Themes mentioned by multiple participants.

	Mentioned by	
Students liked…	*# of participants*	*% of participants*
Open source industry code	10	59%
Basic intelligent API tutor	7	41%
FAQs / related articles	6	35%
Students suggested…		
More details in the tutorial	8	47%
Dictionary of API methods	2	12%
Students commented on usability…		
User friendly	13	76%
Simple/intuitive	10	59%
Not distracting	7	41%
Inconsistent	3	18%
Students also expressed…		
Learning from examples was helpful	11	65%
Using ApiTutors overall felt good	7	41%
Preferring ApiTutors overall vs textbooks	5	29%
Preferring industry code feature vs Google	5	29%
Using ApiTutors overall felt easy	4	24%
Sifting through same information online is hard	3	18%

Overall, most participants mentioned that ApiTutors was user friendly (13), most mentioned that ApiTutors was helpful (11), and nearly half mentioned that ApiTutors “felt good” to use (7). One said, “Easy to use and simple. This is faster compared to hour-long lectures”. Approximately half of participants, unprompted, said ApiTutors was preferable to textbooks and/or Google (five and five, respectively). One said, “Googling for code search and code snippets is a long process and very often the results are not what I am looking for,” and another explained, “While using the internet, I feel there is a lot of information to sift through, but this system provides relevant information all in one place.” In contrast, not a single student stated that textbooks, lectures, Google, or other resources were preferable to ApiTutors in any respect.

VI. IMPLICATIONS FOR EDUCATORS

The qualitative study has demonstrated the potential usefulness of *gathering selected online resources* to help computer science students learn to use APIs. These resources include code snippets (as in preliminary work) as well as FAQs / related articles and open source industry code. Overall, 100% of participants indicated their liking for one or more parts of the resulting training materials, though they differed in which part they preferred the most. The empirical assessment revealed that students considered the approach usable, and many found it superior to existing approaches. In addition, none of them gave any reason why, given the choice of which training materials to use, they would prefer to use existing static training materials or existing search methods instead of the new approach.

A key implication for university faculty in computer science is that providing students with selected resources in these categories could facilitate their learning of APIs. Since not all APIs could be covered, it would be necessary to focus on providing resources for APIs that students are likely to encounter during projects. It is difficult to anticipate these APIs ahead of time; one approach could be for faculty to survey students at the end of the term on the APIs they encountered, and then use this information to gather relevant resources for the subsequent offering of the course. Future empirical research could then investigate the extent to which student projects then improve in quality due to the availability and use of intelligent API tutors.

Educators interested in helping students to learn APIs could also work together to reuse selected resources across universities, thereby covering a broader range of APIs than any individual course, instructor or university could cover alone. For example, instructors could provide their students with access to the online ApiTutors repository, which would be shared among multiple universities; alternatively, ApiTutors could be integrated with one of several existing repositories of educational materials (e.g., Khan Academy or Coursera). Students who encounter difficulties during project-based learning could refer to the repository for assistance and request content for specific missing APIs that they need. Incentive schemes would need to be established so that more experienced students or perhaps teaching assistants would provide enhanced tutors for these student-requested APIs. Such a scheme could lead to a large online repository organized to support easy navigation by different groups of students, such as those in graphics courses, operating systems courses, database courses, or other computer science courses with an integrated emphasis on software engineering.

References

[1] E. Roberts, K. Bruce, R. Cutler, J. Cross, S. Grissom, K. Klee, S. Rodger, F. Trees, I. Utting, F. Yellin, “ACM Java Task Force Report – Project Rationale,” Association for Computing Machinery, New York, NY, 2006.
[2] M. Henning, “API design matters,” *Queue*. vol. 5, pp. 24-36, 2007.

[3] M. Buckley, H. Kershner, K. Schindler, C. Alphonce, and J. Braswell, “Benefits of using socially-relevant projects in computer science and engineering education,” in *SIGCSE Technical Symp. Computer Science Education*, 2004, pp. 482-486.

[4] N. Clark, “Evaluating student teams developing unique industry projects,” in *7th Australasian Conf. Computing Education*, 2004, pp. 21-30.

[5] J. Stylos and B. Myers, “Mica: A web-search tool for finding API components and examples,” *Symp. Visual Languages and Human-Centric Computing*, 2006, pp. 195-202.

[6] A. Dahotre, “jTutors: A Web-based Tutoring System For Java APIs,” Master’s Project Report, School of Electrical Engineering and Computer Science, Oregon State University, 2011.

[7] A. Dahotre, V. Krishnamoorthy, M. Corley, and C. Scaffidi, “Using intelligent tutors to enhance student learning of application programming interfaces”, *ACM J. Computing Sciences in Colleges,* vol. 27, pp. 195-201, 2011.

[8] K. Koedinger, J. Anderson, W. Hadley, and M. Mark, “Intelligent tutoring goes to school in the big city,” *Intl. J. Artificial Intelligence in Education*, vol. 8, pp. 30-43, 1997.

[9] T. Clear, M. Goldweber, F. Young, P. Leidig, and K. Scott, “Resources for instructors of capstone courses in computing,” *ACM SIGCSE Bulletin*, vol. 33, pp. 93-113, 2011.

[10] S. Fincher and M. Petre, “Project-based learning practices in computer science education,” in *28th Annual Frontiers Education Conf.,* 1998, pp. 1185-1191.

[11] R. Todd, S. Magleby, C. Sorensen, B. Swan, and D. Anthony, “A survey of capstone engineering courses in North America,” *J. Engineering Education*, vol. 84, pp. 165-174, 1995.

[12] J. Stylos and B. Myers, “Mapping the space of API design decisions,” in *IEEE Symp. Visual Languages and Human-Centric Computing,* 2007, pp. 50-60.

[13] B. Ellis, J. Stylos, and B. Myers, “The factory pattern in API design: A usability evaluation,” in *Intl. Conf. Software Engineering*, 2007, pp. 302-312.

[14] J. Stylos and S. Clarke, “Usability implications of requiring parameters in objects’ constructors,” in *Intl. Conf. on Software Engineering*, 2007, pp. 529-539.

[15] J. Stylos, B. Graf, D. Busse, C. Ziegler, R. Ehret, and J. Karstens, “A case study of API redesign for improved usability”, in *IEEE Symp. Visual Languages and Human-Centric Computing,* 2008, pp. 189-192.

[16] J. Brandt, P. Guo, J. Lewenstein, M. Dontcheva, and S. Klemmer, “Two studies of opportunistic programming: Interleaving web foraging, learning, and writing code,” in *Intl. Conf. on Human Factors in Computing Systems*, 2009, pp. 1589-1598.

[17] J. Brandt, M. Dontcheva, M. Weskamp, and S. Klemmer, “Example-centric programming: Integrating web search into the development environment,” in *Intl. Conf. on Human Factors in Computing Systems*, 2010, pp. 513-522.

[18] R. Holmes and G. Murphy, “Using structural context to recommend source code examples,” in *Intl. Conf. on Software Engineering*, 2005, pp. 117-125.

[19] E. Linstead, S. Bajracharya, T. Ngo, P. Rigor, C. Lopes, and P. Baldi, “Sourcerer: Mining and searching internet-scale software repositories,” *Data Mining and Knowledge Discovery*. vol. 18, pp. 300-336, 2009.

[20] D. Mandelin, L. Xu, R. Bodik, and D. Kimelman, “Jungloid mining: Helping to navigate the API jungle,” in *Conf. on Programming Language Design and Implementation*, 2005, pp. 48-61.

[21] J. Stylos, A. Faulring, Z. Yang, and B. Myers, “Improving API documentation using API usage information,” in *IEEE Symp. Visual Languages and Human-Centric Computing*, 2009, pp. 119-126.

[22] S. McLellan, A. Roesler, J. Tempest, and C. Spinuzzi, “Building more usable APIs,” *IEEE Software*, vol. 15, pp. 78-86, 2002.
[23] U. Dekel and J. Herbsleb, “Reading the documentation of invoked API functions in program comprehension”, in *Intl. Conf. on Program Comprehension,* 2009, pp. 168-177.
[24] A. Ko, B. Myers, and H. Aung, “Six learning barriers in end-user programming systems,” in *IEEE Symp. Visual Languages and Human-Centric Computing,* 2004, pp. 199-206.
[25] M. Robillard and R. DeLine, “A field study of API learning obstacles,” in *Conf. Empirical Software Engineering*, 2010, pp. 1-30.
[26] S. Chakraborty, T. Bhattacharya, P. Bhowmick, A. Basu, and S. Sarkar, “Shikshak: An intelligent tutoring system authoring tool for rural education,” in *Intl. Conf. on Information and Communication Technologies and Development,* 2007, pp. 1-10.
[27] B. McLaren, K. DeLeeuw, and R. Mayer, “Polite web-based intelligent tutors: Can they improve learning in classrooms?” *Computers & Education*, vol. 56, pp. 574-584, 2011.
[28] J. Anderson, C. Boyle, A. Corbett, and M. Lewis, “Cognitive modeling and intelligent tutoring,” in *Artificial Intelligence and Learning Environments*, 1990, pp. 4-21.
[29] C. Butz, S. Hua, and R. Maguire, “A web-based intelligent tutoring system for computer programming,” in *Intl. Conf. on Web Intelligence*, 2005, pp. 159-165.
[30] J. Holland, A. Mitrovic, and B. Martin, “J-LATTE: A constraint-based tutor for Java,” in *17th Intl. Conf. Computers in Education*,” 2009, pp. 142-146.
[31] B. Reiser, J. Anderson, and R. Farrell, “Dynamic student modelling in an intelligent tutor for Lisp programming,” in *Intl. Joint Conf. Artificial Intelligence*, 1985, pp. 8-13.
[32] A. Ko and Y. Riche, “The role of conceptual knowledge in API usability,” in *IEEE Symp. Visual Languages and Human-Centric Computing*, 2011, pp. 173-176.
[33] C. Treude, O. Barzilay, and M. Storey, “How do programmers ask and answer questions on the web?” in *33rd Intl. Conf. on Software Engineering,* 2011, pp. 804-807.
[34] B. Johnson, and L. Christensen, *Educational research: Quantitative, qualitative, and mixed approaches*, Thousand Oaks, CA: Sage Publications, 2008.
[35] B. Bloom and M. Engelhart, *Taxonomy of educational objectives. Handbook I: Cognitive domain*, David McKay Publishers, 1956.
[36] K. Krippendorff, *Content Analysis: An Introduction to Its Methodology*, Sage Publications, 2004.
[37] D. Riffe, S. Lacy, and F. Fico, *Analyzing Media Messages: Using Quantitative Content Analysis in Research*, Lawrence Erlbaum Assoc Inc, 2005.

Vasanth Krishnamoorthy has a Master’s degree in computer science from Oregon State University and is now a professional software engineer at FindTheBest.com. He is a SUN/Oracle-certified Java programmer and has a Bachelor’s degree from the Sri Venkateswara College of Engineering.

Bharatwaj Appasamy is pursuing his Master’s degree in computer science at Oregon State University. He has been a professional software and/or system engineer at Trimble (maker of GPS tracking systems), Infosys, and Maruti Computers. He has a Bachelor’s degree from the Anna University of Chennai.

Christopher Scaffidi (M ‘05) is an Assistant Professor in computer science at Oregon State University. He has a Ph.D. in software engineering from Carnegie Mellon University and has been developing software as a researcher or a professional software engineer for 17 years. His research focuses on helping software engineers to produce high-quality software by more effectively leveraging one another’s code.