NLP2Code: Code Snippet Content Assist via Natural Language Tasks

Brock Angus Campbell and Christoph Treude
School of Computer Science
University of Adelaide
Adelaide, SA, Australia
a1687816@student.adelaide.edu.au, christoph.treude@adelaide.edu.au

***Abstract*—Developers increasingly take to the Internet for code snippets to integrate into their programs. To save developers the time required to switch from their development environments to a web browser in the quest for a suitable code snippet, we introduce NLP2Code, a content assist for code snippets. Unlike related tools, NLP2Code integrates directly into the source code editor and provides developers with a content assist feature to close the vocabulary gap between developers' needs and code snippet meta data. Our preliminary evaluation of NLP2Code shows that the majority of invocations lead to code snippets rated as helpful by users and that the tool is able to support a wide range of tasks.**

Video: https://www.youtube.com/watch?v=h-gaVYtCznI

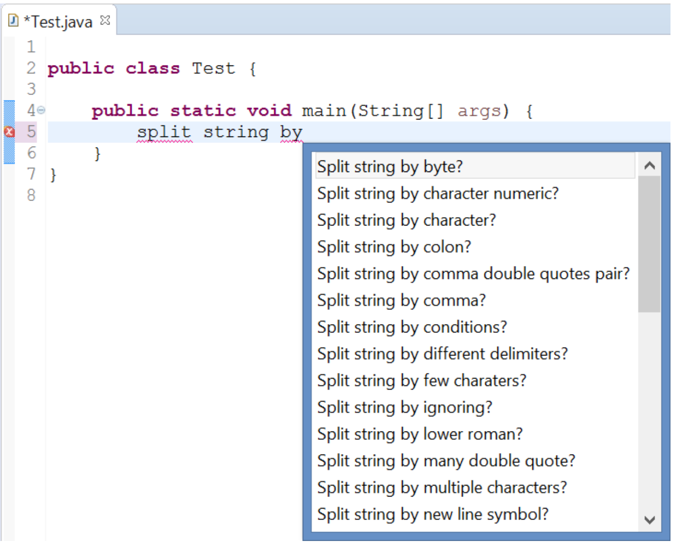
***Keywords*-Code snippets, Stack Overflow, content assist**

I. Introduction and Motivation

The Internet has provided software developers with an effective infrastructure for sharing and accessing programming knowledge, often curated through social media mechanisms, such as voting or commenting [15]. The prime example of this is the Question and Answer website Stack Overflow[1] with more than 12 million questions and more than 20 millions answers as of early November 2016. Many of the questions and answers contain code snippets [14], and much research has been conducted on the quality of these snippets (e.g., [3], [7]).

Question and Answer websites are not primarily designed for direct code reuse [1]. With the current tooling available, a common scenario is for a developer to write code in an Integrated Development Environment (IDE), reach a point at which help is needed, switch to a web browser, type a query into a search engine, evaluate different search results, find a suitable solution, and integrate the newly acquired knowledge into the source code in the IDE—often in the form of a code snippet.

Following the argument that too many context switches between different tools can significantly impact the productivity of a developer [11], we introduce NLP2Code, a plugin for the Eclipse IDE, which allows developers to integrate code snippets from Stack Overflow into their code without leaving the source code editor. Unlike previous efforts to integrate Stack Overflow content into the IDE, such as Seahawk [9], Prompter [10], and T2API [8], NLP2Code does not introduce additional views or windows to the IDE, but works solely in the source code editor.



```
public class Test {

    public static void main(String[] args) {
        split string by
    }
}
```

Split string by byte?
Split string by character numeric?
Split string by character?
Split string by colon?
Split string by comma double quotes pair?
Split string by comma?
Split string by conditions?
Split string by different delimiters?
Split string by few charaters?
Split string by ignoring?
Split string by lower roman?
Split string by many double quote?
Split string by multiple characters?
Split string by new line symbol?

Fig. 1. NLP2Code's content assist

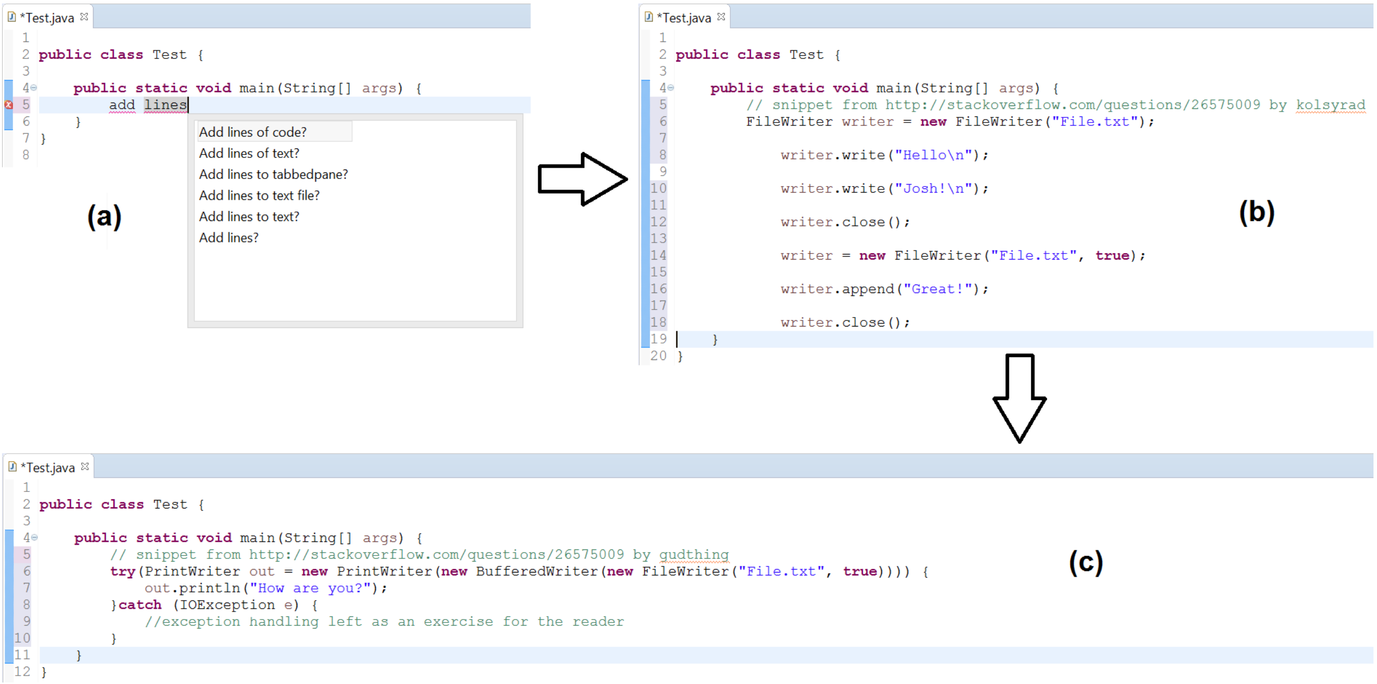
Similar to Eclipse's code completion mechanism [2], we introduce a content assist for code snippets. This content assist is aimed at closing the vocabulary gap [6] between developers' needs and relevant Stack Overflow threads, making suggestions based on natural language tasks that we extract from the titles of Stack Overflow threads, using our previous work on extracting such tasks from software documentation [16], [17].

Our preliminary evaluation of NLP2Code with 6 participants and 101 invocations showed that the majority of invocations led to helpful snippets, often based on the first result. Participants used NLP2Code for a wide range of tasks ranging from looking up implementations of particular algorithms to finding API usage examples and template code.

II. NLP2Code

We developed NLP2Code as a plugin to the Eclipse IDE. However, none of the underlying concepts are Eclipse or Java-specific, and we only focused on this ecosystem since we had access to undergraduate students working with Eclipse for a preliminary evaluation and since Eclipse offers a plugin

[1] http://stackoverflow.com/



```
public class Test {

    public static void main(String[] args) {
        add lines
    }
}
```

```
public class Test {

    public static void main(String[] args) {
        // snippet from http://stackoverflow.com/questions/26575009 by kolsyrad
        FileWriter writer = new FileWriter("File.txt");

            writer.write("Hello\n");

            writer.write("Josh!\n");

            writer.close();

            writer = new FileWriter("File.txt", true);

            writer.append("Great!");

            writer.close();
    }
}
```

```
public class Test {

    public static void main(String[] args) {
        // snippet from http://stackoverflow.com/questions/26575009 by gudthing
        try(PrintWriter out = new PrintWriter(new BufferedWriter(new FileWriter("File.txt", true)))) {
            out.println("How are you?");
        }catch (IOException e) {
            //exception handling left as an exercise for the reader
        }
    }
}
```

Fig. 2. NLP2Code's flow. To transition from (a) to (b), select a content assist suggestion, and to cycle through code snippets, such as (c), press `Ctrl+`‘

infrastructure that has already benefited many other research projects (e.g., [2]).

After installing the NLP2Code plugin, a user can open its content assist anywhere in the source code editor by pressing `Ctrl+1`.[2] Figure 1 shows a screenshot of the content assist after typing *"split string by"*.[3] As shown in this example, even if a user does not exactly know how to specify what they want to split strings by, the content assist helps close this vocabulary gap by providing a large number of suggestions for which NLP2Code can provide code snippets. The content assist suggestions are all natural language tasks that the TaskNav algorithm [16], [17] extracted from all Stack Overflow posts tagged "java" in the latest Stack Overflow data dump.[4] TaskNav works by identifying programming actions in a sentence, and then following grammatical dependencies from the verb to other relevant parts of the sentence. For example, TaskNav extracted the task *"add lines to text file"* from the Stack Overflow question title *"Best strategy to add lines of text to a text file"*.[5] Our assumption that developers need assistance specifically for tasks is supported by the benchmark used to evaluate SWIM [12] where 25 of the 30 benchmark queries follow a similar task pattern. In addition to providing content assist suggestions, the task-based approach helps overcome the commonly used bag-of-words assumption and can distinguish between tasks such as *"convert int to string"* and *"convert string to int"* [5].

Content assist suggestions can be filtered by typing characters and words, and pressing `Enter` selects a suitable suggestion. After the selection, NLP2Code conducts a query using the Google Custom Search Engine[6] on http://stackoverflow.com/. The plugin then collects up to three code snippets (i.e., content surrounded by `<pre><code>` tags) from answers with the highest scores from the four first Stack Overflow threads returned by the search. We chose these numbers by trading off performance (impacted by website queries) and diversity of results (by considering different threads). The first code snippet is automatically inserted into the source code editor along with a comment indicating its source, replacing the original text. The user can then cycle through different code snippets using `Ctrl+`‘.

Figure 2 shows an example of this workflow from our preliminary evaluation. In this case, the participant required help with *"add lines to text file"* and NLP2Code assisted with the suggestions shown in Figure 2(a) after the user typed *"add lines"*. Figure 2(b) shows the first code snippet that NLP2Code returned along with the comment indicating its source. In this case, the participant already rated the first code snippet as helpful. Another option would have been to cycle through further code snippets using `Ctrl+`‘. Figure 2(c) shows the second snippet that NLP2Code would have returned.

In addition to the content assist described above, NLP2Code can be invoked by highlighting any text in the Eclipse IDE and pressing `Ctrl+6` or by surrounding text with question marks (e.g., *"?add lines to text file?"*).

[2] All key bindings are configurable.

[3] *"split string by whitespaces"* was one of the tasks that our study participants completed using the NLP2Code content assist.

[4] https://archive.org/details/stackexchange

[5] http://stackoverflow.com/q/26575009

[6] https://cse.google.com.au/cse/

TABLE I
HELPFUL AND UNHELPFUL NLP2CODE INVOCATIONS

	helpful	unhelpful
invocations	74	27

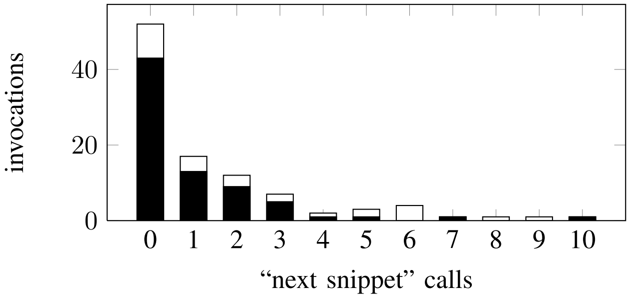


Fig. 3. Number of "next snippet" calls; black denotes helpful snippets and white denotes unhelpful snippets

III. EXAMPLE SCENARIOS

In this section, we discuss two example scenarios for developers using NLP2Code.

A. API usage

One scenario for the use of NLP2Code is that of learning how to use an API. As an example, one of our study participants needed help with *"add custom jpanel to jframe"*. For this task, the participant selected the code snippet in Stack Overflow answer 22621494[7], which was the second code snippet that NLP2Code returned. The code snippet consists of ten lines of code, first setting up instances of JFrame and JPanel before adding the latter to the former. This code snippet was rated as helpful by the participant.

B. Algorithms

Another group of tasks that our participants completed with NLP2Code is the reuse of algorithms. One of our study participants needed help with *"complete bubble sort"*. The first code snippet returned by NLP2Code came from Stack Overflow answer 16089042[8], and it consists of a method implementing bubble sort in 13 lines of code. The participant rated this snippet as helpful.

IV. PRELIMINARY EVALUATION

As a preliminary evaluation of the potential of NLP2Code, we asked ten undergraduate students at the University of Adelaide who are not connected to this research project to use the plugin as part of the development for their coursework. At the time, all participants were undertaking the course "Algorithm & Data Structure Analysis" at the University of Adelaide in which they were required to use Java for their course assignments. We provided all participants with instructions on how to use NLP2Code and encouraged them to use the plugin as part of their Java development. Six of the undergraduate students used NLP2Code and shared their usage data with us (response rate 60%). In addition to the tasks that they used to invoke NLP2Code, we recorded whether their task came from content assist, how many code snippets they cycled through, and whether they considered the final code snippet to be helpful. For the latter, we asked them at the end of each invocation (i.e., at the first Enter keypress after an invocation) *"Was this code snippet helpful?"*

TABLE II
INVOCATIONS WITH CONTENT ASSIST (EXCERPT)

implement dijkstra algorithm
complete bubble sort
sort array for binary search
convert uppercase to lowercase
add custom jpanel to jframe
call sleep on current thread
find number of shortest paths
implement depth first search
compile basic hello world program
generate random integers

TABLE III
INVOCATIONS WITHOUT CONTENT ASSIST (EXCERPT)

initialize two dimensional vector
round double value to two decimal places
download html from a website
remove html tags from a string
evaluate math expression in string form
play a sound
calculate difference between two date instances
generate all permutations of an arraylist
implement floyd warshall
change seed of random generator

Table I shows the number of invocations that resulted in helpful and unhelpful code snippets, respectively. Our 6 participants used NLP2Code for 101 invocations, the majority of which resulted in a helpful code snippet. For about half of the invocations, the content came from content assist. Table II shows examples of invocations using content assist, and Table III shows invocations without using content assist. Note that even the invocations without the content assist feature usually follow a task pattern, i.e., starting with a verb followed by a noun phrase, such as *"initialize two dimensional vector"*. These invocations suggest that our underlying assumption that developers need support for tasks is warranted. Our future work will investigate how we can expand NLP2Code's content assist feature to support more of these tasks.

Figure 3 shows the number of "next snippet" calls that our participants made using Ctrl+` and how many of these invocations led to a helpful result. In many of the cases, the participants did not cycle through results, but chose the first code snippet returned by NLP2Code. The clear majority of these snippets was seen as helpful. Other invocations resulted in up to ten "next snippet" calls, with similar success rates. We conclude that often, the first code snippet returned by NLP2Code can solve the user's task, and in cases where this does not happen, the "next snippet" feature has a high chance of helping the user.

[7] http://stackoverflow.com/a/22621494

[8] http://stackoverflow.com/a/16089042

V. Related Work

NLP2Code is most similar to the recently published T2API by Nguyen et al. [8]. T2API is a statistical machine translation tool that takes a given English description of a programming task as a query and synthesizes an API usage template for this task. In contrast to NLP2Code, T2API has no content assist, its integration into the IDE is not seamless since queries are made in a separate window, and the authors have not evaluated the tool with actual users. In addition, as the name suggests, T2API is specific to API calls whereas NLP2Code can provide code snippets for any task documented in the answers on Stack Overflow. For example, our preliminary evaluation suggests that providing help with algorithms is one of the major benefits of NLP2Code. When we inserted the 20 tasks from Tables II and III into T2API[9], only 4 (i.e., 20%) resulted in reasonable code snippets (*"call sleep on current thread"*, *"generate random integers"*, *"play a sound"*, and *"round double value to two decimal places"*). Another six did not return any code snippet, and the remaining 10 resulted in obviously incorrect code snippets. Similar tools to T2API that also do not provide content assist include the Bing Developer Assistant [19], DeepAPI [5], and swim [12].

Other prominent tools that integrate Stack Overflow content into the IDE are Seahawk [9] and Prompter [10]. Seahawk formulates queries automatically based on the active context of an IDE, presents a ranked list of results, and lets users import code snippets through drag and drop. Prompter takes this idea a step further by notifying developers about the available help. Similar to T2API, these two tools utilize additional windows or views in the IDE and do not integrate into a source code editor in the same way that NLP2Code does.

In the area of code snippet search, Zagalsky et al. [18] presented Example Overflow, an online code search and recommendation tool. Unlike NLP2Code, Example Overflow is not integrated into an IDE, but is its own website. In their work on making sense of online code snippets, Subramanian and Holmes [13] extracted structural information from short plain-text snippets on Stack Overflow and showed that these structural relationships could improve code snippet search.

VI. Conclusion and Future Work

In this paper, we introduced NLP2Code, a content assist for code snippets integrated into the Eclipse IDE. The goal of NLP2Code is to eliminate the context switch that occurs when developers look up code snippets in a web browser while developing in an IDE. NLP2Code closes the vocabulary gap between developers' needs and code snippet meta data through a content assist feature based on natural language tasks extracted from the titles of questions on Stack Overflow. In our preliminary evaluation, NLP2Code returned helpful code snippets in the majority of cases while supporting a wide range of tasks, including API usage examples, template code, and algorithms.

Based on these encouraging results, our future work includes further improvements to NLP2Code's content assist feature and to the mapping of tasks to code snippets, as well as the automated integration of code snippets into the existing code base (e.g., by building on Jigsaw [4]).

[9] http://atwood.encs.concordia.ca:5905/T2APIRESTService/

References

[1] O. Barzilay, C. Treude, and A. Zagalsky. Facilitating crowd sourced software engineering via Stack Overflow. In S. E. Sim and R. Gallardo-Valencia, editors, *Finding Source Code on the Web for Remix and Reuse*. Springer, 2013.

[2] M. Bruch, M. Monperrus, and M. Mezini. Learning from examples to improve code completion systems. In *Joint Meeting of the European Software Engineering Conf. and the Symp. on The Foundations of Software Engineering*, pages 213–222, 2009.

[3] F. Calefato, F. Lanubile, M. C. Marasciulo, and N. Novielli. Mining successful answers in Stack Overflow. In *Working Conf. on Mining Software Repositories*, pages 430–433, 2015.

[4] R. Cottrell, R. J. Walker, and J. Denzinger. Jigsaw: A tool for the small-scale reuse of source code. In *Companion of the Int'l. Conf. on Software Engineering*, pages 933–934, 2008.

[5] X. Gu, H. Zhang, D. Zhang, and S. Kim. Deep API learning. In *Int'l. Symp. on Foundations of Software Engineering*, pages 631–642, 2016.

[6] P. Mika, E. Meij, and H. Zaragoza. Investigating the semantic gap through query log analysis. In *Int'l. Semantic Web Conf.*, pages 441–455, 2009.

[7] S. M. Nasehi, J. Sillito, F. Maurer, and C. Burns. What makes a good code example?: A study of programming Q&A in StackOverflow. In *Int'l. Conf. on Software Maintenance*, pages 25–34, 2012.

[8] T. Nguyen, P. C. Rigby, A. T. Nguyen, M. Karanfil, and T. N. Nguyen. T2API: Synthesizing API code usage templates from English texts with statistical translation. In *Int'l. Symp. on Foundations of Software Engineering*, pages 1013–1017, 2016.

[9] L. Ponzanelli, A. Bacchelli, and M. Lanza. Seahawk: Stack Overflow in the IDE. In *Int'l. Conf. on Software Engineering*, pages 1295–1298, 2013.

[10] L. Ponzanelli, G. Bavota, M. D. Penta, R. Oliveto, and M. Lanza. Prompter: A self-confident recommender system. In *Int'l. Conf. on Software Maintenance and Evolution*, pages 577–580, 2014.

[11] S. Proksch, V. Bauer, and G. C. Murphy. How to build a recommendation system for software engineering. In B. Meyer and M. Nordio, editors, *Software Engineering: International Summer Schools, LASER 2013–2014, Elba, Italy, Revised Tutorial Lectures*. Springer International Publishing, 2015.

[12] M. Raghothaman, Y. Wei, and Y. Hamadi. Swim: Synthesizing what I mean: Code search and idiomatic snippet synthesis. In *Int'l. Conf. on Software Engineering*, pages 357–367, 2016.

[13] S. Subramanian and R. Holmes. Making sense of online code snippets. In *Working Conf. on Mining Software Repositories*, pages 85–88, 2013.

[14] C. Treude, O. Barzilay, and M.-A. Storey. How do programmers ask and answer questions on the web? (NIER track). In *Int'l. Conf. on Software Engineering*, pages 804–807, 2011.

[15] C. Treude, F. Figueira Filho, B. Cleary, and M.-A. Storey. Programming in a socially networked world: the evolution of the social programmer. In *CSCW Workshop on the Future of Collaborative Software Development*, pages 1–3, 2012.

[16] C. Treude, M. P. Robillard, and B. Dagenais. Extracting development tasks to navigate software documentation. *IEEE Trans. on Software Engineering*, 41(6):565–581, 2015.

[17] C. Treude, M. Sicard, M. Klocke, and M. P. Robillard. TaskNav: Task-based navigation of software documentation. In *Int'l. Conf. on Software Engineering - Volume 2*, pages 649–652, 2015.

[18] A. Zagalsky, O. Barzilay, and A. Yehudai. Example Overflow: Using social media for code recommendation. In *Int'l. Workshop on Recommendation Systems for Software Engineering*, pages 38–42, 2012.

[19] H. Zhang, A. Jain, G. Khandelwal, C. Kaushik, S. Ge, and W. Hu. Bing developer assistant: Improving developer productivity by recommending sample code. In *Int'l. Symp. on Foundations of Software Engineering*, pages 956–961, 2016.