Dear Diary: On Documenting Novices' Development Process

Dear Diary: On Documenting Novices' Development Process

Juan Pablo Sáenz
Dip. di Automatica e Informatica
Politecnico di Torino
Turin, Italy
juan.saenz@polito.it

Luigi De Russis
Dip. di Automatica e Informatica
Politecnico di Torino
Turin, Italy
luigi.derussis@polito.it

***Abstract*—In the development projects implemented by novices, the usefulness of the documentation in the form of comments on the final working code is minimal to guide future implementations. Such documentation does not account for novices' development process, including their choices, the errors they faced, the solutions they found, the sources they consulted, the lessons learned, and the advice to remember or give to someone else. Indeed, novices do not usually rely on their documentation to keep track of the successes and errors they find during the development process. Nevertheless, if enabled to capture various moments of the process seamlessly, novices can produce documentation that has the potential to become a valuable asset for them and other developers. This paper presents *Dear Diary*, a tool to support non-expert programmers in straightforwardly creating documentation artifacts directly from the IDE.**

***Index Terms*—novices, documentation artifacts, integrated development environment**

I. Introduction

When approaching a new programming language or working with a given development framework for the first time, novice programmers rely on online tutorials, forums, or source code examples to find solutions and overcome development and execution errors [1], [2]. Once they finish their implementation or reach a working version of their projects, they comment on the resulting code to make it understandable to others (e.g., the teachers who will grade it in introductory programming courses). In this scenario, the usefulness of the produced documentation to guide future implementations is minimal: it does not account for the development process they completed. In particular, it does not provide any traceability on the novices' choices, the errors they faced and how they overcame them, the sources they consulted, the lessons learned, and pieces of advice to remember or give to someone else.

Therefore, the documentation that novices produce (if produced) is not commonly helpful to themselves or other developers to overcome cognitive barriers or guide the development of new projects. Critical aspects such as the background knowledge, the rationale for the solution, or step-by-step instructions for arriving at similar or related solutions are typically left out from the documentation [3]. While various works aim to ease the understanding and integration of online code examples [4]–[6], fewer research efforts have been devoted to supporting novices in seamlessly documenting their development process on the go and with their own words. Additionally, the current documentation practices do not commonly include technical information crucial to reproducing the implemented solution, such as the development and execution environment, the packages used, the dependencies, and the operating system.

Against this backdrop, we consider that novices can produce documentation that has the potential to become a valuable asset meaningful for them and other developers if enabled to capture various points of the development process seamlessly and add self-explanatory insights to it [7]. In particular, we rely on the fact that a self-explanation strategy can increase their awareness of their implemented solution [8], [9]. When learners provide explanations — even to themselves — they learn more effectively and generalize more readily to novel situations [10], [11]. In this paper, we present *Dear Diary*, a Visual Studio Code extension to document the novices' development process by explaining in their own words (as one would do with a diary) how they achieved the working versions and overcame the development and deployment errors.

II. Dear Diary

Hereafter we present our proposed tool through a usage scenario framed in the context of a novice developer creating a web project and we indicate how the various steps are completed through Figure 1.

(a) Claudia is implementing her first React application. She opens Visual Studio Code, her preferred IDE, and she follows a React "getting started" guide that she found online.
(b) In React, the first steps correspond to installing and setting up the execution environment and its dependencies. These installation and setup steps are commonly completed by executing various command-line instructions with diverse parameters.
(c) After executing them to the letter as indicated in the getting starter guide, she manages to create her first *Hello World* React application. At that point, she notices that she has achieved a stable checkpoint, notwithstanding its simplicity. Therefore, directly from Visual Studio Code,

978-1-6654-4214-5/22/$31.00 ©2022 IEEE

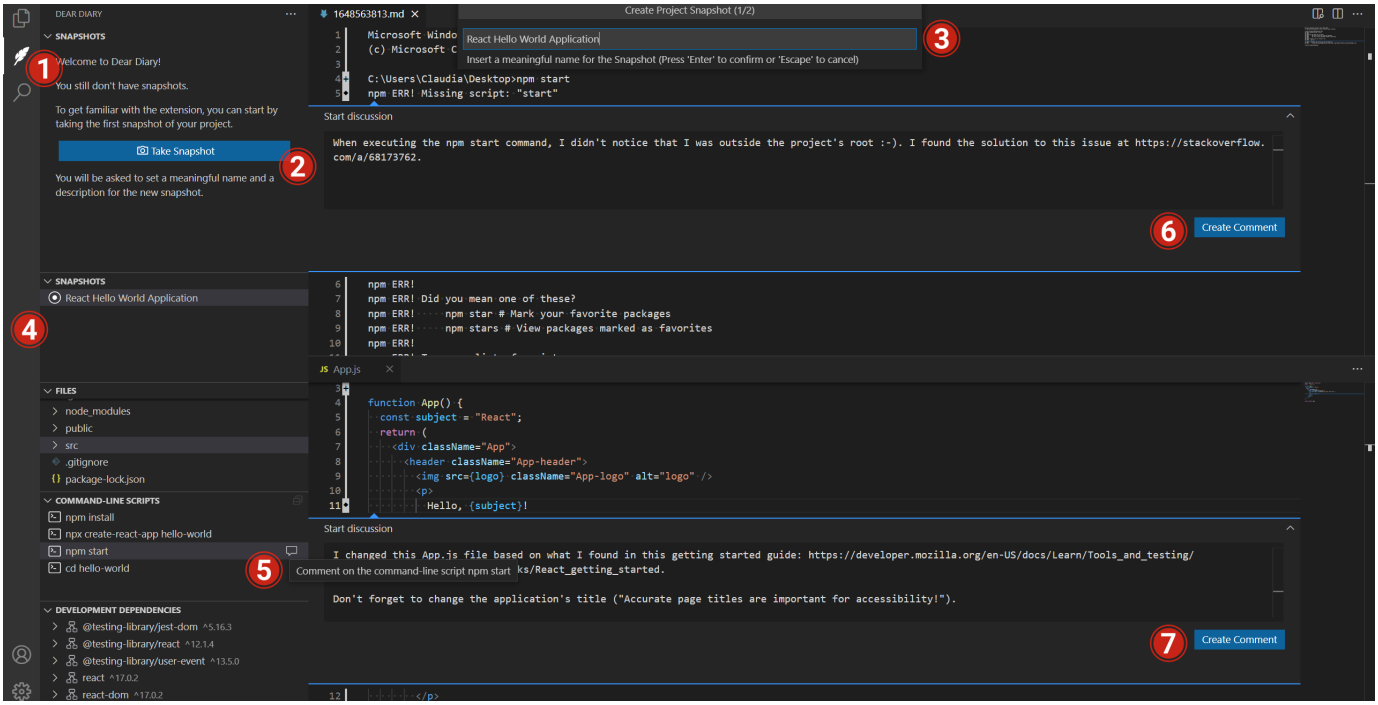

Fig. 1. Dear Diary Visual Studio Code extension snapshot. For illustrative purposes, the figure shows several functionalities occurring simultaneously. Nevertheless, the tool displays fewer graphical components to avoid distracting or overwhelming the programmers.

she opens the Dear Diary extension (**1**) and takes the first snapshot of the current state of the application (**2**). Two popup forms are displayed, asking for the snapshot's title and description (**3**).

(d) Snapshots are partial versions of the development project and are composed of the files, the command-line scripts with their outputs, and the development dependencies. Therefore, when creating the snapshot (**4**), Dear Diary automatically gathers: (i) a copy of all the React application project folders and files as they were when the snapshot is done ('Files' view in the left sidebar); (ii) a log of all the command-line instructions executed since the last snapshot, along with their corresponding outputs ('Command-line scripts' view in the left sidebar); and (iii) information about the development and execution environment, i.e., the Visual Studio Code version; the Node and `npm` installed version; the development and execution dependencies ('Development dependencies' view in the left sidebar).

(e) Claudia tries to execute her application and she gets an error with which she is not familiar. After searching online she finds the cause of this error: the application had to be executed from the project's root directory. Claudia has overcome her first error, and she wants to remember its cause, solution, and the source that she consulted in case it happens again. Then, Claudia selects the corresponding snapshot command-line script (`npm start`, **5**). In the erroneous execution output, she adds an explanation written in her own words and the URL of the Stack Overflow post with the solution to her problem (**6**).

(f) Finally, during the whole implementation process of her React application, she wants to document how the code she is developing is linked to a given architectural decision. For this purpose, she adds several explanations describing the reasoning behind the implementation, the link to the consulted sources, and observations to remember (**7**). Furthermore, these explanations can also be associated with the whole snapshot, a command-line script, or a file (as indicated in **5**, where the comment icon is shown, and its corresponding tooltip says "Comment on the command-line script `npm start`").

(g) From then on, several snapshots can be created by Claudia, each with its associated files, command-line scripts, and development and execution dependencies. Those snapshots are displayed in chronological order (**4**).

III. Conclusion and Future Work

This paper presented Dear Diary, a tool to enable novices to document their development process and add self-explanatory insights regarding the code, the execution outputs, and project versions. These insights account for their choices, the errors they faced, the sources they consulted, and pieces of advice to remember or give to someone else. Future research will concern the qualitative assessment of Dear Diary in the context of a web development university course to evaluate its usability and determine to what extent it effectively supports novices in documenting their projects.

REFERENCES

[1] M. Ichinco and C. Kelleher, "Exploring novice programmer example use," in *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2015, pp. 63–71.

[2] B. Hartmann, D. MacDougall, J. Brandt, and S. R. Klemmer, "What would other programmers do: Suggesting solutions to error messages," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 1019–1028.

[3] H. Li, Z. Xing, X. Peng, and W. Zhao, "What help do developers seek, when and how?" in *2013 20th Working Conference on Reverse Engineering (WCRE)*, 2013, pp. 142–151.

[4] S. Oney and J. Brandt, "Codelets: Linking interactive documentation and example code in the editor," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 2697–2706.

[5] A. Head, E. L. Glassman, B. Hartmann, and M. A. Hearst, "Interactive extraction of examples from existing code," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–12.

[6] M. X. Liu, J. Hsieh, N. Hahn, A. Zhou, E. Deng, S. Burley, C. Taylor, A. Kittur, and B. A. Myers, "Unakite: Scaffolding developers' decision-making using the web," in *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 67–80.

[7] K. Bisra, Q. Liu, J. C. Nesbit, F. Salimi, and P. H. Winne, "Inducing self-explanation: a meta-analysis," *Educational Psychology Review*, vol. 30, no. 3, pp. 703–725, Sep 2018.

[8] C. Vieira, A. J. Magana, M. L. Falk, and R. E. Garcia, "Writing in-code comments to self-explain in computational science and engineering education," *ACM Trans. Comput. Educ.*, vol. 17, no. 4, aug 2017.

[9] C. Vieira, A. J. Magana, A. Roy, and M. L. Falk, "Student explanations in the context of computational science and engineering education," *Cognition and Instruction*, vol. 37, no. 2, pp. 201–231, 2019.

[10] J. J. Williams and T. Lombrozo, "The role of explanation in discovery and generalization: Evidence from category learning," *Cognitive Science*, vol. 34, no. 5, pp. 776–806, 2010.

[11] K. Kwon and D. H. Jonassen, "The influence of reflective self-explanations on problem-solving performance," *Journal of Educational Computing Research*, vol. 44, no. 3, pp. 247–263, 2011.