Application of the Pre-scaffolded Parsons Problem Approach to Mobile Programming Learning with the Topic of Flutter Layout

Putra Prima Arhandi   
*Department of Information Technology*  
*State Polytechnic of Malang*Malang, Indonesia  
putraprima@polinema.ac.id

Dharma Yudistira Eka Putra  
*Department of Information Technology*  
*State Polytechnic of Malang*Malang, Indonesia  
dharmayudistira2000@gmail.com  
Banni Satria Andoko  
*Department of Information Technology*  
*State Polytechnic of Malang*Malang, Indonesia  
ando@polinema.ac.id

line 1: 5th Given Name Surname  
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCIDAnnisa Taufika Firdausi  
*Department of Information Technology*  
*State Polytechnic of Malang*Malang, Indonesia  
annisa.taufika@polinema.ac.id

*Abstract*— Flutter is a product from Google to develop multi-platform applications. However, the declarative coding style of the layout makes Flutter code different from the typical native platform. With the differences in writing the layout code in Flutter, there needs to be an appropriate method so that students can understand the concept of Flutter Layout. This study introduces a new approach in learning Flutter Layout by using Pre-scaffolded Parsons Problem approach. With this approach, students will be provided with an arrangement of code blocks that have been scrambled and students can arrange them into the correct order. In practice, this approach is implemented in a web-based enrichment application called EasyFlutter. To see the impact of implementing this application, the researcher conducted an experiment by providing a pre-test and post-test to 25 participants. In the pre-test, the average result was 77,332 and the post-test was 83,464. Descriptively, there was an average increase of 6,132, then the average difference was carried out using the Paired T-test and got a significant result of 0,00. Based on these results, it can be concluded that there is a significant average difference between the pre-test and post-test.

Keywords—Flutter Layout, Pre-scaffolded Parsons Problem, Paired T-test

# Introduction

The effect of the widespread improvement in information technology has caused several countries to set basic informatics education to be compulsory in school environments [1]. This is also evidenced by the launch of the "Computer Science for All" initiative in 2016 by the US government to introduce programming skills at all levels of education [2]. Therefore, one of the skills that students need in the 21st century is programming skills, where programming education is seen as effective in acquiring these skills [3].

Because mobile devices are portable, learning mobile programming is equally vital to learning computer programming in the discipline of computer science [3]. Many project managers will decide to focus their goods primarily on mobile platforms [4]. Because of this, potential companies now demand that universities turn produce graduates with experience in mobile application development [5].

There are many frameworks that can be used to develop mobile applications, one of which is Flutter. Flutter is a product from Google to develop multi-platform applications [6]. Widgets are the fundamental building blocks used to create Flutter apps [7]. As a result, when developing Flutter applications, it is crucial to comprehend the widget concept.

Flutter's declarative style makes Flutter code different from the native platform. Placing widgets in a declarative style to make a Flutter's layout makes a widget just a lightweight "blueprints" [8]. With the differences in writing the Flutter's layout, it is necessary to have an appropriate method so that students can understand the concept of Flutter Layout.

Lines of code, text, and instruction-based modules are frequently used to teach programming. The teaching module can only be completed by guiding students through a series of steps; nevertheless, it is uncertain how effectively students will comprehend the principles of the teaching module [9]. Additionally, one of the biggest cognitive loads for students is studying through creating lines of code [10]. Because of our limited processing capacity, learning requires a reduction in cognitive loads [11]. For that, we require a different strategy that can help in programming skills, especially in Flutter Layout.

Completion-task is one of the recommended approaches to reduce the cognitive load of students rather than having to do the whole task [12]. Parsons Problem is a programming learning method with a code-completion approach, in which students must rearrange previously scrambled code blocks to reach a solution [13]. Since students do not have to create every single line of code from start, Parsons Problem has a lesser cognitive load and a smaller problem area [12]. At university level, block-based programming exercises are rarely used [10]. In fact, Parsons Problem has shown to be more effective at teaching beginner programmers than creating code directly, correcting problems, and reading instruction-based training modules [13].

There are two types of Parsons Problem, namely student-scaffolded and pre-scaffolded. Student-scaffolded is a type of Parsons Problem that enables students to create their own working framework of code blocks [10]. According to research by Ihantola and Karavirta, student-scaffolded is more challenging than pre-scaffolded [14]. Pre-scaffolded is a type of Parsons Problem that allows students to concentrate on organizing the code blocks in the right order by providing code blocks that have been structured correctly [10].

With reduced cognitive load and limited student problems, the researcher aims to create an enrichment application that can implement Pre-scaffolded Parsons Problem. Therefore, the assumption built in this research is by using this application, the participants will have a good impact for Flutter learning, especially Flutter Layout. To support the assumption, research questions need to be made. So, the research question is: Is there any improvement found from the post-test compared to the pre-test within the group?

This research will use pre-test and post-test to represent the understanding of participants considering that this is still an early stage of the research.

# Literatur Study

## Flutter Layout

Flutter's declarative style makes Flutter code different from the native platform. Placing widgets in a declarative style to make a Flutter's layout makes a widget just a lightweight "blueprints" [8]. Another difference also lies in the layout code. Flutter creates the layout without separating it from the logic code, unlike several mobile application development frameworks that segregate the layout code [15]. This difference is visualized in the **Figure 1**.

Diagram

Description automatically generated

**Figure 1. Visualized Flutter Layout Difference**

## Parsons Problem

Parsons Problem is a programming learning method with a code-completion approach, in which students must rearrange previously scrambled code blocks to reach a solution [13]. Du claim that the Parsons Problem offers various benefits, including the ability to identify student's difficulties, provide feedback, increase student interaction, and reduce cognitive load [10]. The scaffold made depends on the material to be taught. For example in Java, scaffold can be represented as curly brackets or as in the research of Ihantola and Karavirta which uses indentation as a scaffold for learning in the Python [14] **Figure 2**.

Graphical user interface, application

Description automatically generated

**Figure 2. Parsons Problem on Python**

## Pre-scaffolded Parsons Problem

Pre-scaffolded Parsons Problem is a type of Parsons Problem that focuses students on arranging code blocks into the correct order. This can be done by creating a plan on how to solve the issue or by creating pieces of code that have been written correctly but are arranged randomly [10]. Additionally, compared to creating and modifying program code, the Pre-scaffolded Parsons Problem is a more effective teaching method [16].

# Easy Flutter

EasyFlutter is an enrichment application that is intended for students to understand Flutter programming concepts, especially on the topic of Flutter Layout. The "Code Reconstruction" feature in this application will provide some practice questions with the topic of Flutter Layout. This feature implements the Pre-scaffolded Parsons Problem approach as a new approach in learning Flutter Layout. The Pre-scaffolded Parsons Problem approach can reduce the cognitive load of students because students do not need to write lines of code and can limit the problems faced by students such as null pointer exceptions, uninitialized late variables, and several other error messages that are often encountered when writing code directly.

The lecturer is responsible to manage the student's data, data class, and see the logs. In experiment, the lecturer can monitor the student's progress while answering the practice questions **Figure 3**.

Graphical user interface, application, table

Description automatically generated

**Figure 3. Lecturer's View**

The student is responsible to answer ten practice questions that provided by EasyFlutter. To answering the question, student will use the Pre-scaffolded Parsons Problem approach by rearrange the code blocks into the right order **Figure 4**.

Graphical user interface, application

Description automatically generated

**Figure 4. Student's View**

# Experimental setting and discussion

An experiment must be carried out to verify the hypothesis. A controlled environment will be supported by several activities. The activities of each student will be constrained to the designated time and steps.

## Design Experiment

The research was conducted in 2 meetings and 25 participants with majoring Information Technology from State Polytechnic of Malang were involved.

Diagram

Description automatically generated

**Figure 5. First Meeting Experiment**

Based on the **Figure 5**, there were 3 main activities at the first meeting, namely the introduction of research, giving Flutter Layout learning material, and the introduction of EasyFlutter. In the first activity, the researcher will introduction himself and provide an explanation of the aims and objectives of the research. Furthermore, the researcher provided learning material with the topic of Flutter Layout within 45 minutes, this was done with the aim of ensuring that participants had the same understanding on the topic of Flutter Layout. At the introduction of EasyFlutter, participants will access the EasyFlutter and try directly on the tutorial feature for 10 minutes. This feature aims to guide participants on how to use the application, this is done so that participants can get used to using the application.

Diagram

Description automatically generated

**Figure 6. Second Meeting Experiment**

Based on the **Figure 6**, there are 3 main activities in the second meeting, namely pre-test, use of EasyFlutter, and post-test. The pre-test contains 15 questions with the topic of Flutter Layout within 20 minutes, this is done with the aim of knowing the participants’ initial understanding before using the application. Furthermore, participants will use the EasyFlutter and do 10 practice questions on the EasyFlutter for 30 minutes, then participants will take a post-test with a time of 20 minutes. In its application, experiments were carried out using the Zoom Meeting application. The pre-test and post-test were carried out using the Google Form application to make it easier for participants to work on question by online.

## Result

The experiment's pre-test and post-test results will be used to represent the understanding of participants. All the 25 participants participated in the whole series of experiments. The result of pre-test and post-test visualized in the **Figure 7**. Box plot for pre-test has a lower result than the post-test. For the minimum score, there is an increase from pre-test 66,70 to post-test 73,30. For the lower quartile there is also an increase from pre-test 73,30 to post-test 78,325. For the higher quartile there is an increase from the pre-test 80 to post-test 86,70. And for the maximum score there is an increase from pre-test 86,70 and post-test 93.30.

Chart, box and whisker chart

Description automatically generated

**Figure 7. Box Plot for Pre-Test**

## Analysis

An enhanced explanation of the result was subsequently provided using a Paired T-test statical analysis. The data will then conduct normality testing using the Kolmogorov Smirnov method to ensure that it meets the requirements of the Paired T-test.

Table I. Normality Test Result

|  |  |  |  |
| --- | --- | --- | --- |
| **One-Sample Kolmogorov-Smirnov Test** | | | |
|  | | | PRE-TEST | POST-TEST |
| N | | | 25 | 25 |
| Normal Parameters | | Mean | 77,3320 | 83,4640 |
| Std. Deviation | 6,09438 | 7,23941 |
| Most Extreme Differences | | Absolute | 0,229 | 0,233 |
| Positive | 0,186 | 0,169 |
| Negative | -0,229 | -0,233 |
| Kolmogorov-Smirnov Z | | | 1,146 | 1,163 |
| Asym. Sig. (2-tailed) | | | 0,144 | 0,134 |

Based on the results of the normality test in the Table I, the value of "Asymp. Sig. (2-tailed)" both pre-test and post-test are greater than 0,05. So, it can be concluded that the two data are proven to be normally distributed. Thus, the pre-test and post-test data have met the requirements for the Paired T-test.

Table II. Paired T-test Result

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Paired T-test Result** | | | | |
| Pair 1  PRETEST - POSTTEST | Paired Differences | t | df | Sig.  (2-tailed) |
| Mean |
|  |
| -6.132 | -6.545 | 24 | 0.00 |  |

Based on the results of the Paired T-test in the Table II, the value of "Sig. (2-tailed)" is smaller than 0,05. Thus, it can be concluded that there is a significant average difference between pre-test and post-test.

# Conclusion

Based on the data obtained, as many as 25 participants have participated in the entire series of experiment. From this data, there were 21 participants who experienced an increase in the post-test, there were 3 participants who had the same pre-test and post-test score, and there was 1 participant who experienced a decrease in the post-test.

Based on the test results, the average pre-test score of participants was 77,332 and the average post-test score was 83,464. These results prove that descriptively there was an increase in the average post-test score by 6,132 after using EasyFlutter. Not only that, the increase in the average score has also been proven to be significant with the Paired T-test. Thus, the use of EasyFlutter has a positive impact on participants' understanding of the Flutter Layout.

The result of this experiment cannot be used to illustrate the circumstances of a real classroom or to support the assertion that EasyFlutter may be applied there. It still needs a lot of supporting data and facts, which will necessitate active participation from many students, a dedicated lecturer, and extended use of the application in class. However, the data gathered from this experiment might provide a sound basis for doing subsequent study.

##### References

[1] M. Kesselbacher, ‘Supporting the Acquisition of Programming Skills with Program Construction Patterns’, in *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, Montreal, QC, Canada, May 2019, pp. 188–189. doi: 10.1109/ICSE-Companion.2019.00077.

[2] E. Fedorenko, A. Ivanova, R. Dhamala, and M. U. Bers, ‘The Language of Programming: A Cognitive Perspective’, *Trends in Cognitive Sciences*, vol. 23, no. 7, pp. 525–528, Jul. 2019, doi: 10.1016/j.tics.2019.04.010.

[3] M. Dağyar, G. Kasalak, and E. Sezgin, ‘What do primary school students think about mobile programming education? “Developing my own mobile game”’, *WJET*, vol. 12, no. 4, pp. 258–277, Oct. 2020, doi: 10.18844/wjet.v12i4.5179.

[4] M. K. Khachouch, A. Korchi, Y. Lakhrissi, and A. Moumen, ‘Framework Choice Criteria for Mobile Application Development’, in *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, Istanbul, Turkey, Jun. 2020, pp. 1–5. doi: 10.1109/ICECCE49384.2020.9179434.

[5] P. Muyan-Ozcelik, ‘A Hands-On Cross-Platform Mobile Programming Approach to Teaching OOP Concepts and Design Patterns’, in *2017 IEEE/ACM 1st International Workshop on Software Engineering Curricula for Millennials (SECM)*, Buenos Aires, Argentina, May 2017, pp. 33–39. doi: 10.1109/SECM.2017.12.

[6] A. Praveen *et al.*, ‘Conference Room Booking Application using Flutter’, in *2020 International Conference on Communication and Signal Processing (ICCSP)*, Chennai, India, Jul. 2020, pp. 0348–0350. doi: 10.1109/ICCSP48568.2020.9182183.

[7] S. Boukhary and E. Colmenares, ‘A Clean Approach to Flutter Development through the Flutter Clean Architecture Package’, in *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, Las Vegas, NV, USA, Dec. 2019, pp. 1115–1120. doi: 10.1109/CSCI49370.2019.00211.

[8] A. Biessek, *Flutter for beginners: an introductory guide to building cross-platform mobile applications with Flutter and Dart 2*. 2019.

[9] K. J. Harms, N. Rowlett, and C. Kelleher, ‘Enabling independent learning of programming concepts through programming completion puzzles’, in *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, Atlanta, GA, Oct. 2015, pp. 271–279. doi: 10.1109/VLHCC.2015.7357226.

[10] Y. Du, A. Luxton-Reilly, and P. Denny, ‘A Review of Research on Parsons Problems’, in *Proceedings of the Twenty-Second Australasian Computing Education Conference*, Melbourne VIC Australia, Feb. 2020, pp. 195–202. doi: 10.1145/3373165.3373187.

[11] J. Sweller, ‘Cognitive Load Theory: Recent Theoretical Advances’, in *Cognitive Load Theory*, J. L. Plass, R. Moreno, and R. Brunken, Eds. Cambridge: Cambridge University Press, 2010, pp. 29–47. doi: 10.1017/CBO9780511844744.004.

[12] B. J. Ericson, L. E. Margulieux, and J. Rick, ‘Solving parsons problems versus fixing and writing code’, in *Proceedings of the 17th Koli Calling International Conference on Computing Education Research*, Koli Finland, Nov. 2017, pp. 20–29. doi: 10.1145/3141880.3141895.

[13] R. Zhi, M. Chi, T. Barnes, and T. W. Price, ‘Evaluating the Effectiveness of Parsons Problems for Block-based Programming’, in *Proceedings of the 2019 ACM Conference on International Computing Education Research*, Toronto ON Canada, Jul. 2019, pp. 51–59. doi: 10.1145/3291279.3339419.

[14] P. Ihantola and V. Karavirta, ‘Two-Dimensional Parson’s Puzzles: The Concept, Tools, and First Observations’, *JITE:IIP*, vol. 10, pp. 119–132, 2011, doi: 10.28945/1394.

[15] S. Giordano and P. Mainkar, *Google Flutter Mobile Development Quick Start Guide*. Place of publication not identified: Packt Publishing, 2019. Accessed: Dec. 01, 2021. [Online]. Available: https://www.safaribooksonline.com/library/view/title/9781789344967/?ar?orpq&email=^u

[16] B. J. Ericson, J. D. Foley, and J. Rick, ‘Evaluating the Efficiency and Effectiveness of Adaptive Parsons Problems’, in *Proceedings of the 2018 ACM Conference on International Computing Education Research*, Espoo Finland, Aug. 2018, pp. 60–68. doi: 10.1145/3230977.3231000.