Experiences with a Single-Page Application for Learning Programming

Minoru Uehara
Toyo University, Kawagoe, Saitama, Japan
uehara@toyo.jp

Abstract. In recent years, the need for programming has increased. Self-study in programing is important for mastering it. However, many sites for learning programming only support personal computers. JavaScript Development Environment (JDE) was developed in our previous research. JDE is a single-page application that allows a user to learn programming on a smartphone. It imposes a small load on the server, thus many people can use it at the same time. Because it does not use a computer, it can be used in a classroom. Furthermore, JDE is suitable for distance learning (or e-learning). The number of remote classes has increased in Japan in 2020, as a result of the coronavirus pandemic. In this paper, we introduce an example of distance learning using JDE and verify the effectiveness of JDE.

1 Introduction

Recently, the necessity of programming education has been increasing [1]. Programming education is a part of science, technology, engineering, and mathematics (STEM) education. Among STEM subjects, programming education has a substantial impact [2]. Programs can visualize teaching materials and can explain the causal relationship between phenomena.

In primary education, many teaching materials have been developed and are actually used, using visual languages such as Scratch and Google Blockly. However, there are few examples of its use in higher education institutions, such as universities. There are several reasons for this.

(R1) Most programmers develop applications using text-based languages, rather than visual languages. Therefore, most of the teaching materials mentioned above are not suitable.
(R2) Many sites for learning programming have limited application. Programmers learn new languages on such sites, and many of them provide remote execution environments. However, in many cases, those execution environments only support simple input/output (I/O). They are fully usable to teach grammar and basic concepts, but they cannot teach advanced contents. More advanced contents are more difficult to learn using these environments.

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2021
L. Barolli et al. (Eds.): BWCCA 2020, LNNS 159, pp. 55–66, 2021.
https://doi.org/10.1007/978-3-030-61108-8_6
(R3) The cost of providing a learning environment is high if the application is not restricted. Because it is necessary to construct a dedicated learning environment on its own, its development costs are added to the tuition fees. It is difficult to provide services when the number of learners cannot be predicted. For example, when using graphics other than HTML or using a special device, then remote execution is difficult.

(R4) Otherwise, learners will need to prepare their own learning environment. If the learners’ skills are inadequate, then they will fail to set up the environment in the early stages of learning.

Many universities avoid these methods for reasons R1 or R3. However, reasons R2 and R4 can be partially relaxed. Many universities, despite R2, use learning sites partially as teaching materials for home study. Additionally, despite R4, a classroom equipped with personal computers (PCs) provides an exercise environment that is difficult for students to construct.

In our previous research [15], we developed the JavaScript Development Environment (JDE). JDE is a single-page application (SPA) that allows a user to learn programming on a smartphone. It imposes a small load on the server, so many people can use it at the same time. Because it does not use a PC, it can be used in regular classrooms (those that are not equipped with PCs). That is, problems R2 and R4 (above) can be partially solved at the same time.

The number of remote classes has increased in Japan in 2020, as a result of the coronavirus (COVID-19) pandemic. JDE is suitable for distance learning. In this paper, we introduce an example of distance learning using JDE and verify the effectiveness of JDE.

The paper is organized as follows. Section 2 describes related research. Section 3 describes JDE in detail. Section 4 introduces actual classes using JDE. Section 5 describes the experience gained through the lessons. We conclude in Sect. 6.

2 Related Work

In this section, we describe related work.

2.1 Web-Based Integrated Development Environments

Web-based integrated development environments (Web IDEs) are increasing in popularity because of the spread of cloud computing. Arvue [3] is a Java-based Web IDE. Eclipse Che is based on Eclipse. Adinda [4] is collaborative. Cloud9 can be used as a web IDE. However, Cloud9 is basically an editor and does not include a language-specific development kit. Yanagisawa et al. are developing a web-based programming environment [5–9].

Web IDEs are also available on Chromebooks. However, they are not suitable for smartphones because of their small screens. Furthermore, some web-based editors do not display a virtual keyboard, thus they are not available for smartphones.
2.2 Code-Sharing Sites

Code-sharing sites provide the functions of code sharing and code execution. However, not all code-sharing sites provide these features; they can be classified into three types depending on the combination.

1. Sites that just share code (e.g., GitHub).
2. Sites that can execute shared code (e.g., CodePen and JSFiddle.net). CodePen specializes in front-end XX.
3. Sites on which code can be executed but cannot be shared (e.g., Paiza.io).

Of the above systems, (2) and (3) can be used as Web IDEs.

2.3 Chrome Extensions

A Chrome extension is an application that is used by incorporating it into the Chrome web browser. Chrome extensions include many development tools.

Web Maker is a Chrome extension that can be used as a development tool. It is considered to be a standalone version of CodePen and works offline. However, it is not available on Chrome for smartphones.

2.4 Single-Page Applications

An SPA is an application composed of a single web page (i.e., an HTML file). Strictly speaking, the back end exists on the server side, so the application on the client side consists of a single page. The views are dynamically generated on the client side, allowing the application to exhibit excellent response time.

An SPA communicates asynchronously with the back end. Asynchronous communication can hide latency, and thereby shorten the response time. Increased responsiveness improves the user experience and makes an SPA easy to use.

An SPA is often developed using SPA frameworks, such as React and Vue.js. Currently, JDE (described below) does not use these SPA frameworks. Rewriting JDE using an SPA framework is a topic for the future.

2.5 Flip Teaching

Flip teaching [10, 11] is a type of blended learning that mixes distance learning and face-to-face learning. Basically, self-study is performed using prepared video materials, and later reviewed by face-to-face classes.

Khan Academy offers a variety of materials that can be used in flipped classrooms. There are many cases of reversing programming [12].

3 JavaScript Development Environment

We developed JDE in a previous study [15]. This section describes its design and implementation.
3.1 Features

JDE has the following features.

(F1) JDE can execute code in a web browser.
(F2) The JDE screen is suitable for smartphones.
(F3) The operation of JDE is suitable for smartphones.

These characteristics endow JDE with several advantages. JDE is usable offline. JDE does not overload the server, and can therefore be used by many users at the same time. JDE is usable on smartphones, so students can study while commuting between school and home.

According to the Japan Student Services Organization, the average commuting time for Japanese college students is 1 h [13]. According to another survey, half of the students use the railway. Therefore, students can use their smartphones to study while commuting to school.

3.2 Screen

Figure 1 shows a part of the screen running JDE on the iPhone X. The JDE screen consists of four parts: (1) menu, (2) code field, (3) output field, and (4) console field.

Various buttons are lined up in the menu. The user inputs the code in the code field and edits it. Virtual keyboards on smartphones are not suitable for entering code, so JDE provides some buttons that help users to enter code. A virtual keyboard does not allow the user to enter a tab for indentation, so a Tab button is used for this purpose. The Enter button will enter an indentation after a line break. The +return button adds a return on the last line. The For button creates a for statement. For example, nested loops can be created from a list of variable names. The If button creates an if statement. The Syntax button assists with entering reserved words. The Op button assists with entering operators. The Brackets button creates brackets of various types, and positions the cursor between them. The Snippets button inserts frequently used code.

Values generated by code execution can be added to the console and displayed. The final result of the code is displayed in the output field as the return value of the function. If no return value is specified, then the value displayed is undefined.

3.3 Library

One of the problems with learning programming on smartphones is code input. An example library was developed to eliminate the need for code input (No Code Input).

Figure 2 shows the screen of the example library. The example library is a regular web application, not an SPA. Students can use sections of code registered by the teacher. Currently, more than 200 sample sections of code are registered.

The copy is performed as follows. First, the sample code is saved in the local storage of the browser with its tag name, which is an identifier that corresponds to the file name. JDE loads the sample code from local storage by using the tag name.
3.4 Summary

JDE is an SPA JavaScript development environment. JDE can be run only in a web browser and can be used offline. Students need to access the server to browse the example library, but once saved, they can then browse the library offline.

JDE is designed to operate on a smartphone. Therefore, using JDE, teachers can usually teach in regular classrooms. The number of Wi-Fi connections is usually limited in these classrooms. However, because JDE runs on a smartphone, students can connect to the server with IP telephony (4G). Once they download the SPA, it can be used offline.

4 A Classroom Using JDE

JDE has been used in the actual course “Introduction to Programming” since 2017. The course is offered in the first (Spring) semester. (In Japan, the school year starts in April.)
It has a total of 15 lessons with two credits. In these lessons, both lectures and exercises are offered.

The purpose of the course is for students with no programming experience to learn the basics of programming. Although it uses JavaScript as its programming language, it emphasizes the learning of language-independent principles. The contents include basic concepts, such as data, variables, I/O, conditionals, loops, arrays, functions, and combinations thereof.

The contents of each lesson are as follows.

1. Introduction
2. Basic Concepts
3. Getting started with JavaScript/Output
4. Expressions/Variables/Input
5. Conditional Statements and Logic
6. Loops
7. Nested Loops
8. Computational Complexity
9. Arrays
10. Objects
The number of students taking the course in 2017, 2018, 2019, and 2020 was 78, 82, 72, and 78, respectively. Although it is not a compulsory subject, many students choose it. The same course is offered in three courses in parallel. Students are automatically assigned to a course by student ID number. Therefore, the number of students enrolled in each course is approximately one-third of the entire capacity of 260.

The features of this course are as follows.

(P1) JavaScript is used as the programming language.
(P2) JDE is used as the development environment.
(P3) Students can also practice on smartphones.
(P4) Classes can be held in regular classrooms, instead of PC rooms.
(P5) New concepts are combined with known concepts.
(P6) Many examples are used.

JDE satisfies the features of P1–P4. In the course, we use many examples for P5 and P6. For example, if students learn loops after learning conditional statements, they also learn applications in which a loop is combined with a conditional statement. After more classes, the number of combinations increases. Many sample sections of code are used to learn the combinations.

The sample code can be found in the text distributed by the learning management system (LMS) called manaba [14]. This LMS prohibits the use of script tags and style tags, for security reasons. Therefore, it is necessary to manually copy and paste code from that text into JDE to run it. At that time, extra blank lines may be copied. Therefore, in 2019, we added a function to delete blank lines.

Copying and pasting with a smartphone is not always easy. It is difficult to specify a sample code section accurately when it extends over 100 lines. Therefore, in 2019, we developed the example library, which avoids the need for students to enter the code. Students can also modify an example and try it out.

Features added in 2019 are detailed in Ref. [15]. For example, the Open button has been added to create a window and display HTML. It is difficult to use the Run button, which executes JavaScript, to execute the I/O code of a Form.

5 Experiences of a Classroom Using JDE

In 2020, many universities offered classes online because of the coronavirus pandemic. We also held classes online. In this section, we describe our experiences of an online class using JDE.

Online classes can be classified into two types: live-streaming classes and on-demand-type classes. The former streams lessons using a video conferencing system,
such as Zoom, Google Meet, Microsoft Teams, or Cisco Webex. This type of class is suitable for conducting traditional face-to-face lessons online. The latter distributes lesson videos using video on demand (VoD), such as YouTube or Microsoft Stream. This type is suitable for flipped classes: in the flipped classroom, video materials are originally delivered on demand.

The subject “Introduction to Programming”, described in Sect. 4, uses on-demand classes. With this type of class, the content distribution method is important.

The materials are shared with students on Google Drive, to limit the audience. The file size of a 90-min MP4 video (Full HD) is around 1 GB. Assuming that approximately 80 people download it at the same time, 80 GB of traffic will be generated at the beginning of the class, which is difficult to serve with an on-premises file server. Ideally, a VoD platform should be used. However, any platform that can handle a large amount of traffic can be used instead. For example, cloud-based online storage can be used as an alternative platform. Therefore, Google Drive can be used for content distribution. Our university and many others use G Suite for Education.

JDE traffic will be described next. According to our previous research [15], JDE can be downloaded by each user in around 300 ms. The load on the server is proportional to the number of users, which we measured from the server log.

Table 1 shows the results of classifying URLs from the logs for a period of 2 weeks (July 8–July 22, 2020). Here, the URL indicates an address pattern. The URL “/” matches addresses of all pages on this site. The JDE URL is “/jde”, which includes the addresses of the JavaScript library and the example library. The corresponding URLs are “/jde/js” and “/jde/lib”. Therefore, the number of accesses to “/jde/”, excluding the library, is only 392, at most. Students accessed it only five times on average. Because the classes were held twice every 2 weeks, the students only accessed it 2.5 times per class. By implementing JDE as an SPA, we were able to reduce the number of downloads significantly. However, the problem is that JavaScript libraries were downloaded more often than SPAs. This JavaScript library displays line numbers in the code field, but it is not absolutely necessary. Traffic can be reduced by removing unnecessary libraries.

Table 1. URLs accessed and the number of accesses to each.

| URL   | Accesses |
|-------|----------|
| /     | 26843    |
| /jde  | 3731     |
| /jde/js | 1820   |
| /jde/lib | 1519   |

Figure 3 shows the dates of accesses. Classes were held every Friday, including July 10 and July 17. Although access was distributed, peaks occurred on class days. The number of accesses was almost equal to the number of students taking the course. Therefore, it follows that almost all the students had access. Because the system time
zone is Coordinated Universal Time (UTC), accesses on class days were divided between July 16 and 17. In general, the number of accesses on weekends was small.

![Fig. 3. Access dates.](image)

Figure 4 shows the distribution of access times. Because the time zone is UTC, the time difference from Japanese Standard Time (JST) is +9. Class hours 16:30–18:00 (JST) are 7:30–9:00 (UTC). The first peak occurred during regular school hours. The next peak was at 14:00 (UTC) or 23:00 (JST). Students have a habit of learning before bed. The on-demand method is ideal for the lifestyle of such students.

![Fig. 4. Access times.](image)
Table 2 shows the types of client operating systems. JDE can be used on smartphones, but many students use PCs. However, the total number of iPhone and iPad accesses exceeded the number of Macintosh accesses. The Linux accesses were from IP addresses starting with “64” octets. According to Table 3, they were from Google and were not student accesses.

Table 2. Client operating systems and the number of accesses.

| Client    | Accesses |
|-----------|----------|
| Windows   | 211      |
| Macintosh | 47       |
| iPad      | 37       |
| iPhone    | 27       |
| Android   | 5        |
| Other(Linux) | 65       |

Table 3. Client IP addresses.

| Address pattern | Carrier | Accesses |
|-----------------|---------|----------|
| 126.77.137.x    | Softbank | 37       |
| 64.233.172.x    | Google  | 17+12+12 |
| 118.156.139.x   | KDDI    | 15       |
| 110.132.0.x     | Jupiter | 12       |

Table 3 shows the IP addresses of the clients with the most accesses. The last octet is hidden. These were all class A addresses. The “Carrier” column indicates the administrator, as displayed by WHOIS. All except Google are mobile carriers. Google uses multiple addresses to collect files, such as favicons, and should be excluded.

The results in Table 2 suggest that students were using JDE from their PCs. However, the results in Table 3 suggest that a significant number of students used mobile carriers, but not all of them did so.

JDE can be used on smartphones, but the communication charges for smartphones are high, and students always act to reduce them. When students watch a 1-GB lesson video on their smartphone, their communications charges are very high, and therefore many students watch the video on their PC instead. The PC is connected to a flat-rate home Internet connection and includes a large screen, with video and JDE displayed side by side at the same time. The above suggests that there are few advantages to using a smartphone for home learning. Rather, it is more advantageous to use it in a regular classroom.
6 Conclusions

This paper described experiences of distance learning using JDE. Students downloaded JDE only 2.5 times per class on average. Therefore, the load on the server was very small. Furthermore, student accesses were distributed outside of class days. It was often accessed late at night and during class hours.

With JDE, students can study both in the classroom and at home. However, with home learning, smartphones have no advantage over PCs. Therefore, the advantage of smartphones is that they can be used in a regular classroom.

The topics for future work are as follows.

JDE will be combined with other services. For example, JDE will make mutual copies of code with LMSs such as manaba. This will allow students to test programming problems.

We will also improve the user interface, using React or Vue.js. JDE can be used as a smartphone application by rebuilding it with Flutter or React native XX.

Acknowledgments. We thank Edanz Group (https://en-author-services.edanzgroup.com/ac) for editing a draft of this manuscript.

References

1. Cabinet Office, Programming Education. http://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1375607.htm. Accessed 4 July 2019
2. Vee, A.: Coding Literacy: How Computer Programming Is Changing Writing. The MIT Press, Cambridge (2017)
3. Aho, T., et al.: Designing IDE as a service. Commun. Cloud Softw. 1(1) (2011)
4. van Deursen, A., Mesbah, A., Cornelissen, B., Zaidman, A., Pinzger, M., Guzzi, A.: Adinda: a knowledgeable, browser-based IDE. In: Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, (ICSE 2010), vol. 2, pp. 203–206. ACM, New York (2010). http://dx.doi.org/10.1145/1810295.1810330
5. Yanagisawa, H., Uehara, M., Mori, H.: Web-based collaborative development environment for designing processors. GESTS Int. Trans. Comput. Sci. Eng. 30(1), ISSN 17386438 (2006)
6. Yanagisawa, H., Uehara, M., Mori, H.: Interface implementation using ajax for web-based instruction set simulator. In: Proceedings of 2nd International Workshop on Telecommunication Networking, Applications and Systems (TeNAS2008), pp. 1511–1516 (2008)
7. Yanagisawa, H., Uehara, M., Mori, H.: Web-based collaborative development environment for an ISA simulator. In: Cooperative Internet Computing, pp. 125–138 (2008). https://doi.org/10.1142/9789812811103_0009
8. Yanagisawa, H.: Evaluation of a web-based programming environment. In: 2012 15th International Conference on Network-Based Information Systems, Melbourne, VIC 2012, pp. 633–638 (2012). https://doi.org/10.1109/nbis.2012.67
9. Yanagisawa, H., Kondo, K.: Implementation of web-based interactive interface as software execution environment. In: 2013 16th International Conference on Network-Based Information Systems, Gwangju, pp. 383–388 (2013). https://doi.org/10.1109/nbis.2013.61
10. Lage, M., Platt, G., Treglia, M.: Inverting the classroom: a gateway to creating an inclusive learning environment. J. Econ. Educ. 31, 30–43 (2000)
11. Baker, J.W.: The “Classroom Flip”: using web course management tools to become the guide by the side, p. 15. Communication Faculty Publications (2000)

12. Çakıroğlu, Ü., Öztürk, M.: Flipped classroom with problem based activities: exploring self-regulated learning in a programming language course. J. Educ. Technol. Soc. JSTOR, 20(1), 337–349 (2017). www.jstor.org/stable/jeductechsoci.20.1.337. Accessed 21 July 2020

13. Japan Student Services Organization, “Student Life Survey Report 2019. (in Japanese). https://www.jasso.go.jp/about/statistics/gakusei_chosa/__icsFiles/afieldfile/2020/03/16/data18_all.pdf. Accessed 07 July 2020

14. AsahiNet, manaba. http://manaba.jp/. Accessed 07 July 2020

15. Uehara, M.: JavaScript development environment for programming education using smartphones. In: 2019 Seventh International Symposium on Computing and Networking Workshops (CANDARW), Nagasaki, Japan, pp. 292–297 (2019). https://doi.org/10.1109/candarw.2019.00058