“INSERT_TITLE_HERE”

Investigating the communication of placeholders and placeholder-characters in code examples in the programming trade.
This exam work has been carried out at the School of Engineering in Jönköping at JU in the subject area Informatics. The work is a part of the Bachelor of Science continuation program (New Media Design). The author takes full responsibility for opinions, conclusions and findings presented.

Examiner: Ulf Seigerroth

Supervisor: Johan Kohlin

Scope: 15 credits (first cycle)

Date: 29 May 2017
Abstract

To improve the communication between programming instructors and learners; placeholders, placeholder-characters and placeholder characteristics are investigated via two focus groups. One focus group with instructors, the other with learners. The combined and analyzed results show that placeholder-characters can confuse more than they help, more often for programming beginners. The semantics of a placeholder is the most crucial part of understanding its purpose and strengthens its characteristics. A placeholder using a different letter case from the surrounding code is easier to differentiate and find in a block of code.
Summary

The study’s purpose is:

To find out if digital characters could improve the communication between programming instructors and learners by standardizing the implication “placeholder” or “replace this”. What characteristics could be used to communicate placeholders?

To answer the purpose, it has been broken down into two research questions.

1. What opinions do programming instructors and learners have about using placeholder-characters to improve the communication between them?
2. What characteristics could a placeholder have to better communicate its purpose?

To answer the research questions, two focus groups were held. One with instructors and one with learners. They are then asked to discuss placeholders, placeholder-characters and their characteristics. Later transcribed, categorized and analyzed into findings that are discussed and concluded upon.

The participants mostly believed that placeholder-characters can confuse more than they are able to help, more often the case for beginners. That the placeholder’s semantics of the label is crucial for understanding the purpose of it. Using a different letter-casing for placeholders could make it easier to find and understand that it deviates from the rest of the code in the concerned code example, preferably “SCREAMING_SNAKE_CASE”. A placeholder-character is considered viable to use as a global standard if it complies with the following five requirements:

- It is universally used (A standard).
- Available easily (found on any keyboard).
- Shows what and what not to replace (keep quotation marks).
- Is understood by programming-software (displayed and compiled correctly).
- Not used for other purposes in coding already (has another implication).

Otherwise it is not considered supposable for implementation.

Keywords

Placeholder, placeholder-character, code example, semantics, communication, programming.
Contents

1 Introduction ............................................................................................................. 1
  1.1 PURPOSE AND RESEARCH QUESTIONS....................................................... 2
  1.2 DELIMITATIONS....................................................................................... 3
  1.3 OUTLINE ............................................................................................... 3

2 Method and implementation .............................................................................. 4
  2.1 CORRELATION BETWEEN RESEARCH QUESTIONS AND CHOICE OF METHOD 4
    2.1.1 CORRELATION BETWEEN THE FIRST RESEARCH QUESTION AND CHOICE OF METHOD .... 4
    2.1.2 CORRELATION BETWEEN THE SECOND RESEARCH QUESTION AND CHOICE OF METHOD .... 5
  2.2 PROCESS................................................................................................. 5
    2.2.1 FOCUS GROUPS PROCESS .................................................................... 5
    2.2.2 APPROACH .................................................................................. 6
  2.3 DATA COLLECTION AND DESIGN ......................................................... 6
  2.4 DATA ANALYSIS................................................................................... 6
  2.5 CREDIBILITY ....................................................................................... 7

3 Theoretical framework ......................................................................................... 9
  3.1 DEFINITIONS.......................................................................................... 9
    3.1.1 INSTRUCTORS AND LEARNERS ....................................................... 9
    3.1.2 CODE SAMPLES AND EXAMPLES ................................................... 9
    3.1.3 PROGRAMMING LANGUAGES ......................................................... 9
    3.1.4 PLACEHOLDER ........................................................................... 9
    3.1.5 PLACEHOLDER-CHARACTER.......................................................... 10
    3.1.6 PLACEHOLDER-TITLE CONVENTIONS ...................................... 11
    3.1.7 UNICODE .................................................................................. 11
    3.1.8 DICTIONARIES AND ENCYCLOPEDIAS ....................................... 11
    3.1.9 API DOCUMENTATION AND CODE EXAMPLES ......................... 11
    3.1.10 BRACKETS AND PARENTHESES ............................................... 12
    3.1.11 SEMANTICS ............................................................................ 12
## Contents

3.2 **COMMUNICATING PROGRAMMING** .......................................................... 12

4 **Findings and analysis** .................................................................................. 14
   4.1 **FOCUS GROUPS** ..................................................................................... 14
      4.1.1 **TEACHER-GROUP** ........................................................................ 14
      4.1.2 **STUDENT-GROUP** ........................................................................ 16

5 **Discussion and conclusions** ....................................................................... 18
   5.1 **DISCUSSION OF METHOD** .................................................................... 18
   5.2 **DISCUSSION OF FINDINGS** ................................................................. 19
      5.2.1 **TEACHER-GROUP’S FINDINGS DISCUSSED** .............................. 19
      5.2.2 **STUDENT-GROUP’S FINDINGS DISCUSSED** .............................. 20
      5.2.3 **COMBINED FINDINGS DISCUSSED** ........................................... 21
   5.3 **CONCLUSIONS** ..................................................................................... 23
   5.4 **FURTHER RESEARCH** .......................................................................... 24
   5.5 **IMPLICATIONS** .................................................................................... 25

6 **References** .................................................................................................. 26

7 **Appendices** .................................................................................................. 28
   7.1 **APPENDIX 1 — FOCUS GROUPS SCRIPTS** ......................................... 28
   7.2 **APPENDIX 2 — FOCUS GROUPS INTRODUCTION** ............................ 30
The title of this study is not a mistake, it is intended. This study is about investigating a solution for a potential problem in the programming business. To primarily see if a placeholder-character could improve the communication between programming instructors and learners. And secondarily discover the perceived characteristics of a placeholder.

Code.org (using 2010 statistics), claimed that there will be 1 million more programming jobs than graduated computer science students by 2020, and later analyzed by Davis (2016). The Swedish Public Employment Service “Arbetsförmedlingen” clearly shows that in the next ten years there will be a continued difficulty to recruit IT-architects, software- and system developers according to their forecast “Var finns jobben 2017?” (What jobs are needed in 2017?), (Arbetsförmedlingen, 2017). It is apparent that it is a trade on the rise with a need for programmers, and there is always room for improvement in the programming learning process. When a person wants to learn how to program or implement a code example from a peer, they need to moderately understand how to write their code. A way of teaching code, programming language syntaxes and API’s are through code and code examples. Placeholders in code examples are what this paper is aiming to improve. Programmers need to overcome obstacles when they learn programming. Ko, Myers and Aung (2004) bring up six different types of learning barriers in End-User Programming Systems in their report. They are: design-, selection-, coordination-, use-, understanding- and information barriers. The main focus of this paper is on the understanding barriers, which they explain briefly as, “I thought I knew how to use this, but it didn’t do what I expected...”. Basically, when a learner tries to evaluate their program’s behavior related to their expectations.

There should generally be a lesser amount of understanding barriers if we want to improve communication in the area. This paper aims to ease these “Understanding barriers” by investigating the need for placeholder-characters in code samples. Placeholder-characters today are non-standardized and appear in many different forms or even non-existent as later shown in 4. Findings and analysis.

Following is an example (Figure 1) of a current use of a placeholder from a code example found on the online programming forum Stack Overflow. Where the learner is expected to replace ‘YOUR_API_KEY’ with something else, something unique/individual, in this case it could be replaced with ‘apiKey27901’ for example, to potentially make it work. Here the placeholder-characters are non-existent and the implication of “replace this” is only conveyed through its own characteristic text.

This in comparison to how Microsoft make use of placeholder-characters in a specific code sample (Figure 2). They use less-than (LT) and greater-than (GT) signs to imply...
that the learner should replace the contained text (including the LT-, GT-signs) with something else.

```
11 # Your account key goes here.
12 account_key = '<your account key>'
```

Figure 2, Example of code sample found at https://text-analytics-demo.azurewebsites.net/Home/SampleCode

The problem here is that there isn’t a standard placeholder-character or characteristic that instructors can conform to. A learner that uses sample codes like these preceding ones can’t promptly know for sure what text currently is used as a placeholder. In certain cases it could be hard to demonstrate what naming convention or format the replaced text should be replaced with, because the text itself is used as a placeholder and not the placeholder-characters instead. This also applies to teachers who use code samples in their education.

### 1.1 Purpose and research questions

As mentioned before, there is a problem with the way programmers communicate solutions with each other. It is also a problem between programming instructors and learners, whenever a teacher wants to teach a new programming syntax to its learners. Therefore, the purpose for this study is:

> To find out if digital characters could improve the communication between programming instructors and learners by standardizing the implication “placeholder” or “replace this”. What characteristics could be used to communicate placeholders?

To answer the purpose, it has been broken down into two research questions.

To find out if digital characters could improve the communication between programming instructors and learners by standardizing the implication “placeholder” or “replace this”. We must know if a standardization of this kind would be meaningful for programming instructors. Therefore, the first research question for this study is:

1. What opinions do programming instructors and learners have about using placeholder-characters to improve the communication between them?

To examine what the digital characters could be, the study must find out if any currently available characters could be. Therefore, the second research question for this study is:

2. What characteristics could a placeholder have to better communicate its purpose?
1.2 Delimitations

The informants in this study were chosen from the School of Engineering at JU, students and teachers with convenient access to the date and place of the focus group interviews.

The characters studied are limited to the ones that can be found in the Unicode Character Database due to Unicode’s substantial comprehensiveness in the world.

1.3 Outline

In chapter 2. **Method and implementation**, there is an overview of the correlation between the research questions and choice of method, the work process and data collection.

In chapter 3. **Theoretical framework**, a theoretical background and several explanations are described.

In chapter 4. **Findings and analysis**, the collected data is described and analyzed to help answer the research questions.

In chapter 5. **Discussion and conclusions**, a summarizing description of the study’s results is given. The results and methods used are discussed and conclusions are written. Implications and examples of further research are brought up at the end of this chapter.

After chapter 5. the references are documented in a bibliography and appendices follow it.
2 Method and implementation

2.1 Correlation between research questions and choice of method

2.1.1 Correlation between the first research question and choice of method

This paper’s first research question was about a problem that may be encountered in a programming teachers’ everyday life but doesn’t have a standardized solution yet. This work is a case study using the qualitative method Focus Groups. According to Clifford, Cope, Gillespie and French (2016), a focus group is usually 6-12 people who meet in an informal setting to talk about a particular topic with them as a group. If a focus group has 3-4 participants that also have some expertise or special knowledge in the subject area, it is rather called a “mini-focus group”. Mini-focus groups are small focus groups and are endorsed by Kreuger (1994, p. 17).

“The comparative advantage of focus groups as an interview technique lies in their ability to observe interaction on a topic. Group discussions provide direct evidence about similarities and differences in the participants’ opinions and experiences as opposed to reaching such conclusions from post hoc analyses of separate statements from each interviewee. “ (Morgan, 1997, p. 10)

For the first of two focus groups, teachers (instructors) at the School of Engineering at JU were invited to voice their thoughts about this issue and its potential solution. A group of 4 teachers and tutors who all teach programming in some way, participated in the study. The paper’s purpose reaches into a higher level of computer knowledge where the participants should be able to know how programming works and understand the concept of placeholders.

The second focus group out of the two were held with various programming students (learners) at the School of Engineering at JU. The aim was to use the learners as a contrast to the instructors, to find out their opinions for the first research question. By holding a second focus group, a more nuanced information can be gathered. A clearer difference between the two subject groups could appear when two relatively different groups are interviewed. This way they complement each other (Jacobsen, 2002, p. 179).

The method “focus groups” was chosen for several reasons. Mostly because of the difficulties that other conventional methods would bring for this report’s subject. A quantitative method, such as a survey or questionnaire would be hard to carry out. Due to the difficulty to gather enough data from a relevant population with this advanced subject, where explaining it to people through only a small chunk of text would be a strenuous task. The potentially large amount of people that would not be able to understand the questions would not be desired. Therefore, a method that allows for easy supplementary explanation when needed was preferred.

There was a possibility to perform separate interviews instead of grouped interviews in the focus groups. Interviews would possibly be able to give richer data, but it was estimated to be of a less generalizability than comparing the two chosen research populations in separated focus groups.
2.1.2 Correlation between the second research question and choice of method

The paper’s second research question is generally about finding out what characteristics placeholders have and what they should have, to improve its communication purpose, which is meant to be replaced.

To collect data from the targeted users, the two different focus groups were shown code examples that include placeholder-characters. At the same time the data collection from the first research question was being accumulated. They were able to discuss the characters’ contemporary meaning, design, understandability and readability among other things. The focus groups were therefore directed to answer both research questions. Many of the same arguments that are made about using other methods than focus groups under the heading 2.1.1 apply to this too.

2.2 Process

2.2.1 Focus groups process

The focus groups that were held, meant to involve at least six participants in each focus group, while at most twelve people would be able to attend, due to how the group dynamics change and more people than that would be unwieldy to manage (Millward, 1995, p. 425). The first focus group which will later be called the “Teacher-group”, consisted of four teachers/tutors (instructors) at the School of Engineering at Jönköping University (JU). It is considered a “mini-focus group” henceforth. The second focus group, later called the “Student-group” consisted of seven students (learners) at the same school (School of Engineering at JU). The population for these two are therefore:

Teacher-group: Teachers/tutors (instructors) at the School of Engineering at JU.

Student-group: Students (learners) at the School of Engineering at JU.

They were chosen because of their availability and logistical reasons, they were all in close proximity of Jönköping University, the place where the focus groups were held. Another point of view that was considered, was that a number of the students were taught by a few of the teachers that were asked to participate in the meetings. This could result in unexpected comparisons between the groups, but may also be in the way of an open approach, where the data is possibly restricted to fewer different experiences. The participants were found via the schools’ contact list, course lists, teachers and friends. These were then contacted through e-mail. The first positive responders were then considered interested in participating in the study. The process concluded then by finding the best common date and meeting place for everybody.

The focus groups were aimed to consist of equal amounts of men and women, because men and women tend to act differently in mixed groups, than in same sex groups (Millward, 1995). It was also important to keep the subject narrow because themes that are too broad have tendencies to derail (Jacobsen, 2002, p. 174).

The focus groups were actively moderated by asking questions and letting the participants theorize freely, also letting them formulate new questions that the researcher hasn’t thought about.
2.2.2 Approach

The deductive approach served this study well since the concept was to find out more about a potential problem and test the hypothesis with the help of empirical data (Jakobsson, 2011). A deductive method is when a focal point for the scientific paper has been chosen before the research is made; the inductive approach is vice versa, when the research defines the study instead (Blomkvist & Hallin, 2014). The deductive process has gotten some critique. Its biggest weakness is that it is often narrow in its data collection. The researcher is usually focused on fulfilling the hypothesis to see other interesting information, but the focus can be more open. The inductive process is not perfect either; it is said that it is impossible and naïve to think that a person can have a completely open mind in reality. It is also common now to argue whether how open or not a study’s approach is, on a spectrum, rather than saying if it is purely deductive or inductive. This study is aimed to be more on the open side of the deductive side of the spectrum (Jacobsen, 2002, pp. 42-45).

2.3 Data Collection and Design

The focus groups were to be one hour at most and the audio was recorded by the moderator using two separate recording devices. The two groups (Teacher-group and Student-group) were meant to be held with the same conditions from the moderator, in an attempt to reach the same consistency. Though a focus group is depending on the feedback from the participants and when two different groups are held at different times they can’t fully reach the same consistency. The first mentioned group (Teacher-group) was held in a conference room at JU and the second in a group room at JU.

Pen and papers were to be handed out to the participants at the beginning of the interviews. The papers were blank to let the participants write and draw in order to help them document their thoughts. An introduction of the problem was given to the participants, a similar background as in chapter 1. Introduction was used (found in Exhibit A of Appendix 2 — Focus groups introduction). After the introduction, they were asked semi-structured questions and able to discuss among themselves. When the participants were familiarized with the subjects and some questions have been brought up, they were shown some code examples and placeholder-characters. The participants were made aware that the solution to the problem might not exclusively be able to be solved by existing bracket characters, so that they aren’t fully biased to the currently used characters (in the shown examples).

After the participants gave their feedback, the blank papers handed out in the beginning of the interviews were collected for additional documentation. Many of the decisions that were taken for the focus group design was gathered from Krueger and Casey’s “Designing and Conducting Focus Group Interviews” (2002).

2.4 Data Analysis

The data analysis is broken down into three parts later. 1 Description; 2 Systemization and Categorization; and 3 Combination; Jacobsen (2002, p. 216), Dey’s Qualitative Data Analysis (1993) and Miles and Huberman’s Qualitative Data Analysis (1994).
1. **Description**, in this phase, the qualitative data will be described as detailed as possible. The interviews are registered thoroughly. Rich on details, analyses and variations.

2. **Systemization and Categorization** is when the data is reduced and systemized into a perspicuous information that is easier to analyze. This is to be able to get an overview of the rich qualitative data. An exhaustive description is often too dense for others than the researcher to understand. A systemization is thereby necessary to mediate the findings.

3. **Combination** is when the data has been systemized it is generally ready for interpretation. Looking for intentions, reasons and trying to generalize and structure the data. In this phase, the data is analyzed for things that have been seen or heard. The more hidden and perhaps the most interesting conditions are hopefully found at this stage.

The analysis is found in chapter 4. **Findings and analysis**.

2.5 **Credibility**

A quantitative research method for the first research question would possibly be able to produce a more credible research if asking the right questions in a questionnaire or finding a trustworthy source of statistics. This was difficult to approach, due to this study’s first research question is considering a subject that is very new to the field and a questionnaire would probably be very advanced for most participants. Let alone, difficult to target correctly. The subject is hard to explain to a broad audience. That is why the qualitative method “focus groups”, with an advanced and targeted audience, was chosen. Even though a quantitative approach would enable a more credible generalization of the data.

Qualitative research has its drawbacks. The data is not statistically represented and not easily generalized in the same way quantitative data can be. It is also dependent on the researcher's/interviewer's skills, orientation and interpretation when researching. The research may have difficulties to maintain an objective view because it is based on a personal interaction. It may also have bigger differences between sessions of data collection because of (in this case) the interviewer's semi-structured questions in the interviews compared to what it could have if a quantitative approach was selected instead.

Qualitative research is not limited to numerical results; therefore, it can explore topics in detail with great depth compared to quantitative research (ORISE, 2017).

It was argued that a content analysis (a quantitative subcategory of the qualitative method document analysis) could be carried out for exploring today's online use of placeholders and their characteristics. But there was a big problem with using this method in a deductive study. In a content analysis, the researcher is searching for specific traits, characters, words et cetera. The study has already locked down the scope for what is considered data and what isn’t. This doesn’t leave much room for unexpected data to appear.

The questions in the focus groups must be challenged, their intentions are to measure the research questions, and if they aren’t, the internal validity must also be questioned. They might be asking about something else outside the papers' research questions and therefore gathering data that is not applicable to this study’s purpose. The focus groups
might not be moderated correctly or other factors such as, the interviews surroundings or the partiality of the participants might not be ideal.

To know if this study could be applied to other fields the external validity needs to be questioned. The quantitative data from the API documentations content analysis needs to have a large enough population to be able to generalize. If the sample group is too small, it would be hard to measure correctly and the statistics would not be able to be generalized fairly.

Because the focus groups are held in Jönköping, Sweden and only interviews participants of Swedish descent, it is in a culturally biased setting. It would take more work to research further on the rest of the world and its cultural and lingual differences. The research’s credibility about the test subjects should bear this in mind.

The study’s qualitative data is hard to replicate because of the individuals that are interviewed in the focus groups. They are part of a specific contemporary culture that might be tough to find again. The knowledge they had before participating in the focus groups may be very different compared to any succeeding researches that use similar populations as this papers’ focus group-populations.

The participants of the focus groups might come to know the moderator beforehand even though Krueger (1994) claimed that participants of a focus group shouldn’t know the moderator because they might be influenced if they associate the moderator with some specific opinion or orientation. But Hylander (1998) argues that there are instances when you can’t avoid people from knowing each other. The focus groups that participated in this study, are in that type of setting Hylander wrote about. Since a physical meeting that took place with a lot of participants that have met each other before through school, it was hard to avoid the fact that they know each other with the chosen data collection population.
3 Theoretical framework

3.1 Definitions

Below, the different wordings and constructs are set and explained further to set common definitions for the paper.

3.1.1 Instructors and learners

An instructor is according to the dictionary Merriam-Webster (2017) “One who instructs”. Cambridge Dictionary (2017) explains it as “A person whose job is to teach people a practical skill”. In this study, the plural of the word instructor is used to imply teachers, tutors, mentors, educators, instructresses, forum-posters and the authors of API documentations.

The word “learner” is used instead of words such as students, pupils, educatees, scholars, apprentice, forum-post readers or API documentation readers; to aggregate these words into one. Merriam-Webster (2017) explains the word as to someone who gains knowledge or understanding of or skill in by study, instruction, or experience. Cambridge Dictionary (2017) describes it as “a person who is still learning something”. The word “reader” is also used in this text to indicate a learner.

3.1.2 Code samples and examples

The terms “sample code”, “code snippets”, “code sample” and “code examples” are used interchangeably in this paper to refer to the samples of code that are used in the instructing process of programming.

3.1.3 Programming languages

A programming language is “a set of rules, words etc. that are used for writing computer programs” according to Cambridge Dictionary (2017). In this study, there are several programming languages used in body text and figures. The study uses the abbreviations of these languages throughout. Some are: “PHP”, a recursive acronym for “PHP: Hypertext Preprocessor” and “JS” that stands for JavaScript.

3.1.4 Placeholder

A placeholder is a word often used in mathematics and computing. It is “a symbol or piece of text in a mathematical expression or computer instruction that can be replaced by particular pieces of information” according to the Cambridge Dictionary (2017). Wiktionary (2016) defines it as “Something used or included temporarily or as a substitute for something that is not known or must remain generic; that which holds, denotes or reserves a place for something to come later”. Placeholders appear often in digital file naming, digital templates and code examples/snippets in programming lectures, forum posts and API documentations. Two examples are given to explain the use of a placeholder.

1. An example of when a placeholder is used in a PHP code example from Google’s YouTube Data API documentation (2016) is found in Figure 3.
Theoretical framework

// Replace with section title of your choice.
$SECTION_TITLE = 'YOUR_SECTION_TITLE';

Figure 3. Code example from YouTube Data API.

The green text “YOUR_SECTION_TITLE” is meant to be replaced with something like “Inspiring videos” and keeping the single quotes (‘ ’) since they are necessary for the PHP programming syntax.

2. The following (Figure 4) is an example where placeholders are used in digital file naming when a teacher wants to receive a digital file, named a certain way.

“When you hand in the assignment you should name it as:
‘Javascript-Lab-1-FIRSTNAME_LASTNAME.js’
to simplify the grading process.”

Figure 4. Digital file naming example.

Here the text “FIRSTNAME” and “LASTNAME” is meant to be replaced with the students actual first name and last name, to end up with something like this: “Javascript-Lab-1-Sven_Parker.js”.

The term “placeholder” in this paper, is not to be confused with:

• HTML form fields placeholders, the often grey text that is present in some form fields right until something is typed inside the field.
• The wildcard character, such as an asterisk (*), percent symbol (%) or question mark (?) that is usually used in software searches so that the full name need not be typed. It is a single character that is used to represent several characters or an empty string. (Computer Hope, 2017)
• A digital type of bookmark/placemark/placement that is used to keep track of where a user is inside of an e-book.
• Placeholder names such as John Doe or metasyntactic variables such as Foo, Bar and Baz. Even though this papers’ placeholders could act in a similar way.
• Placeholder-text used in design purposes. A text that fills space temporarily, intended for another text in the future. As “Lorem Ipsum” is used.
• The U+FFFD REPLACEMENT CHARACTER used to replace an unknown, unrecognized or unrepresentable character.
• The templating placeholder used in many templating programming languages. Used when a code that contains template-placeholders is compiled and replaced completely by the computer, according to its languages syntax (often seen as “{{ }}”).

3.1.5 Placeholder-character

A placeholder-character is either a digital character or a set of digital characters that are used to signify that the contained text should be replaced. Examples of these are: Quotation marks “ or ’, parentheses (), curly brackets {}, square brackets [] or lesser-than and greater-than signs < >. In use it could look like this: “[FIRSTNAME]”. Where both square brackets act as placeholder-characters and “FIRSTNAME” acts as the placeholder-title in this example.
3.1.6 Placeholder-title conventions

There isn’t really a standard convention for placeholder-titles that instructors do or can conform to. There are multiple attempts to standardize the naming conventions, but no clear way that a learner knows what should be replaced. People might use “screaming snake case” (API_KEY), “kebab case” (api-key) or “camel case” (apiKey) with or without a combination of quotation marks (“ or ’), round parentheses ( ), curly brackets { } or square brackets [ ] to signal where to input individual data. The words “api key” are used as examples in parentheses to show the named letter cases. Another way of signaling it is to change color of the text which should be replaced, such as “this” (text is red). Though using colors to indicate placeholders, disadvantages accessibility.

3.1.7 Unicode

Unicode is a non-profit, 501(c)(3) charitable organization that develops and maintains the Unicode Standard which has been adopted by the main industry leaders. The Consortium works closely with W3C and ISO, which is responsible for maintaining ISO/IEC 10646, the International Standard synchronized with the Unicode Standard (The Unicode Consortium, 2017).

3.1.8 Dictionaries and Encyclopedias

Merriam-Webster, Wiktionary and Cambridge Dictionary are all information databases that offer a variety of reviewed content. These three are considered quite dependable because of their established backgrounds and high status. Even though Wiktionary is community-driven, all opinions and content must be backed up by references. It is the world’s voice in different from the dictionaries, where there is an educated team that manages the dictionary.

3.1.9 API documentation and code examples

API stands for Application Programming Interface and generally denotes a set of clearly defined methods of communication between various software components.

Publicly available (free) API’s usually come with a documentation that describes what services the API offers and how to use those services. The documentation is traditionally found in documentation files/websites, blogs or online forums.

API documentations can contain multiple code examples to show how the API could be used. To show how the learner could write its program for it to work with the API.

“As showed by vast empirical research, a major obstacle faced by developers when learning APIs is the lack of examples of usage.”

(Montandon, et al., 2013)

Developers usually view code examples as recommendations on how to use an API and it is considered harder to learn if there aren’t many code examples available. So, it is recommended that an API documentation contains usage examples for the learner.
3.1.10 Brackets and Parentheses

The difference between a “bracket” and a “parenthesis” can be a bit confusing. Generally, “parenthesis” refers to round brackets ( ) and “brackets” to square brackets [ ] or curly brackets { }. It may bear a cultural difference in what the custom is. People are used to hearing these referred to simply as “round brackets”, “square brackets” or “curly brackets”. This study uses the term “brackets” to refer to all types of brackets and parenthesis.

3.1.11 Semantics

Semantics is “the study of meanings in a language” according to Cambridge Dictionary (2017). A collective word that includes several teachings and studies. Placeholders usually has a label built up by word or words. The choice of wording is an important characteristic in the understanding of placeholders. A placeholder-label that explains through its chosen words what it is supposed to be replaced with is considered semantically sound, it is descriptive towards its purpose. Merriam-Webster’s definition of semantic is: “of or relating to meaning in language” (2017).

3.2 Communicating programming

There are several ways to learning programming today. Study online or offline via a school or teach yourself by looking up how to program online, on a great number of different forums, code-sources, documentations or websites. These approaches are all, in one way or another, depending on actual written code for the learner to be able to write its own code. It is thereby crucial to be able to communicate programming and code in a desirable way.

If a programmer already knows the basics of a programming language or its way of thinking, they often search on a search engine (such as Google) to find a solution to their current problem.

"Developers are mainly using Web searches to opportunistically solve software development problems (82% of Web searches). Opportunistic searches are ad hoc and are done to remember syntax details, clarify implementation details or fix bugs, and learn new concepts. On the other hand, non-opportunistic searches (only 18% of Web searches) are done following a systematic process and are performed to find open source projects." (Gallardo-Valencia & Sim, 2013, p. 253)

Gallardo-Valencia and Sim observed 24 software developers at 3 different companies and studied these types of searches which brought forward 4 main categories according to them, they are as following: Remember/Fact Finding, Clarification, Learn, Look for Tool/Open Source Project. Where the first three approaches are considered opportunistic and the fourth non-opportunistic. This shows that a lot of the educative searches performed by software developers are done to clarify implementation details or fix bugs, remember syntax details and learn new concepts.

Developers compose around 12 search queries per weekday on average according to Sadowski, Stolee and Elbaum (2015). Thereby there is clearly a necessity to ameliorate this process further.
Ko, Myers and Aung (2004) bring up six learning barriers in end-user programming systems. One barrier that this study aims to ease is the “Understanding barrier”. It is when a learner did not expect the same output as to what their program output. It can happen when a placeholder is incorrectly replaced or when the programming syntax is not followed completely.
4 Findings and analysis

4.1 Focus groups

To answer the research questions, two separate focus groups were held. The first research question was answered by asking programming instructors and learners about what opinions they have about using placeholder-characters to improve the communication between them. The second research question was answered by asking the same focus group participants about their thoughts on placeholders. Finding out what characteristics a placeholder could have to better communicate its purpose.

The focus groups were semi-structured with a script, found as Appendix 1 — Focus groups scripts. The researcher acted as the moderator and held both 45-minute focus groups in English during the same day, Friday the 28th of April, but at different times. The Teacher-group was at 1:00 pm – 1.45 pm in the conference room “The Triangle” at JU, and the Student-group was at: 4.00 pm – 4.45 pm in a group room (E1112) at JU. They were transcribed to include thoughts and useful information and are therefore not verbatim transcripts. These transcripts are rich on details so the data was categorized and reduced into overviews in Figure 5 and Figure 6. They are confidential and therefore the names have been turned into random initials marked in bold.

4.1.1 Teacher-group

The instructors’ focus group was held with four instructors of different kinds, all active at the School of Engineering at JU. Half were teachers and the remaining half were tutors. One woman and three men. Originally there were eleven possible participants that were contacted and asked to participate, that gives a turnout of around 36%.

The participants of the Teacher-group gave their opinions on the issue at hand. These opinions were then categorized and put into a perspicuous table so that it could be analyzed easier. The reduced data for the Teacher-group is displayed in Figure 5. Here the participants’ standpoint was recorded as either a “Yes” or “No whether they agree or disagree with the essence of the statement/category. The participants’ names are shortened into a random letter for confidential matters. The amount of “Yes’s” and “No’s” in each row is measured in the columns of totals. If an opinion on a statement didn’t come through, the standpoint was documented as “Unspoken” and didn’t count to either “Yes” or “No”. The statements are in no particular order except for a descending order from the total amount of “Yes” they have received, where the first four statements are from scripted questions that have an introductory intention. The rest of them are categories that were extracted from the raw opinions that the participants have expressed in one way or another. When an engagement is registered, it is added to the second last row, “Total amount of engagements”. Then a percentage out of all the possible engagements are calculated in the last row, “Percentage of engagements”.

---
The group was unanimous about two things, that the semantics of the placeholder label is crucial for understanding a placeholder and that the semantics are more important than eventual placeholder-characters. They were nearly in consensus about five more categories: “Uppercase with underscores is a good way to imply a placeholder”; “Implying ownership in the placeholder as ‘YOUR’ would be a good solution”; if decided to, “Two consecutive curly brackets ‘{{ }}’ would be a good candidate for...
Findings and analysis

placeholder-characters; “It is not always obvious if you should keep or remove quotation marks around placeholders” and that “There is a communication problem between instructors and learners”. Half of the group clearly expressed standpoints on five other categories. These categories were: “Uppercase with underscores is the most common way to imply a placeholder”, “The color-coding on placeholder-strings can be confusing for beginners and should be considered” when improving placeholders, “Adding explanatory comments would make it easier” to understand some placeholders, “Adding an introductory protocol for the learner would be good for setting a common placeholder” and “This is a problem that is met by good education”. There are also some less directly agreed upon statements that were expressed. K stated that placeholder-characters are not needed, that it is important that programming-software-tools (that display code) should be able to understand what a placeholder is and that there is no “one” standard to placeholders. B also declared that there could be different definitions of “placeholder”, that can make it hard to understand.

There were differing amounts of engagements. The highest engagement was 16 out of a possible 20 categorized opinions and the lowest had half of that (8 engagements). The percentage between them is 40%. The engagement overall is at 59% on the chosen categories. Do keep in mind that not all opinions were explicitly spoken and that there could be more agreements and disagreements to different subjects.

4.1.2 Student-group

The learners focus group was held with seven students from the School of Engineering at JU. It consisted of two women and five men. Twelve possible participants were originally asked to participate in the focus group, that is a turnout of about 58%.

The participants of the Student-group gave their opinions on the issue at hand. These opinions were then categorized and put into a table the same way as the Teacher-group was, so that it could be analyzed easier. The reduced data for the Student-group is displayed in Figure 6. The same structure as the Teacher-group is used in the Student-groups’ table.
Findings and analysis

Here all the participants agreed upon four statements. They were: “It is not always obvious if you should keep or remove quotation marks around placeholders”, “The semantics of a label is crucial”, even “more important than placeholder-characters” and “Placeholder-characters that aren’t universal would be more confusing” than simplifying. Then there are two statements that about half of the group concur with, “Adding explanatory comments would make it easier” to understand some placeholders and “It is more important that the placeholder-characters are universal than what design they are”. And lastly there were four categorizations that didn’t get that many “Yes’s” compared to the others. P and T agree that “It is hard sometimes to understand when something needs to be replaced”. E stated two more things: “Implying ownership in the placeholder as ‘YOUR’ would be a good solution”, “Uppercase with underscores is the most common way to imply a placeholder”. A meant that “Standardized placeholder-characters should be based per school”, so that every school has a set placeholder-character that is used throughout the education and that it doesn’t have to be standard.

Here there weren’t too much differing amounts of engagements. The highest engagement was 11 out of a possible 14 engagements and the lowest had 7 opinions. The percentage between them is 29%. The engagement overall is at 64% on the chosen categories.
5 Discussion and conclusions

5.1 Discussion of method

The method “focus groups” had both general strengths and weaknesses as in every study, it gave evaluations that emerged later in this study’s analysis. As discussed before in 2. Method and implementation, focus groups is a qualitative method that needs to be analyzed further to bring out the essence of the interviews. That is a drawback to an extent when compared to quantitative data that can be generalized and analyzed more easily, but it is also an advantage in some cases where it is more important to get rich data that contains more overlooked information than what is usually expected in a quantitative approach. That is what this study experienced during the focus groups. There was overlooked info that would make the study’s findings something else than expected. The use of a relatively open approach proved helpful in this. Focus groups gave the advantage of this outcome and could therefore bring in data that would probably be undiscovered in a quantitative method. The report got what it searched for and more.

The focus groups met some difficulties. It was hard to gather participants and figuring out a common time and place for them. Like any meeting, some participants could attend to one suggested time when some others couldn’t join that same date. Also, the population for the instructors that was researched, was very narrow. There wasn’t that many that could be asked to participate in the first case. Broadening the populace would mean that more participants could be found, but would also risk becoming too hard to generalize and analyze when comparing the two focus groups and may be troublesome, for logistical reasons. This affects the study’s validity to the worse, but is, arguably, still great enough to truly represent what this study is measuring.

The focus groups were semi-structured interviews with scripted questions combined with the participants’ freedom to discuss what they had on their mind. Enabling questions that were not scripted beforehand and the possibility of finding out the participants’ unexpected opinions, that helped answering the study’s first research question. This also gave a richer data for answering the second research question, when the participants of the Teacher-group brought this matter up somewhat unscripted.

In the Teacher-group, there were only four that could participate at one of the set dates. This is on the very edge of being able to be called a focus group, it was rather a “mini-focus group”. Mini-focus groups are small focus groups with 3-4 participants that have some expertise or special knowledge in the subject area, as endorsed by Kreuger (1994, p. 17). This differs from the Student-group which interviewed seven participants. The students (learners) that participated were within the decided “learner-population” didn’t have an expertise in the area, but rather a fair knowledge about it. That was desired to be researched upon.

There were a differing number of standpoints between the participants. In the Teacher-group, K had sixteen spoken standpoints and P had half that amount. This could mean that P didn’t get to express that many opinions clearly for many possible reasons. Possibly he/she wasn’t given the space needed to speak their mind or maybe they didn’t have that much opinions to give. Perhaps K took too much space in the meeting and therefore the other participants weren’t given the same possibility to speak. Maybe the moderator should have interrupted more and enquired the participants that didn’t express their opinions as much.
The Student-group had more unanimous opinions than the Teacher-group, but also had less clearly observed statements. The Teacher-group had 20 registered statements and with only four participants and the Student-group had 14 registered statements with almost double the participants. This may have been an impact of the groups different sizes, knowledge or perhaps the meetings environment and the questioning methods used. The groups were to be held by the moderator as similarly as possible to each other, so that they could be analyzed and discussed in parallel.

5.2 Discussion of findings

The purpose and research questions are restated below.

The purpose for this study is:

To find out if digital characters could improve the communication between programming instructors and learners by standardizing the implication “placeholder” or “replace this”. What characteristics could be used to communicate placeholders?

To answer the purpose, it has been broken down into two research questions.

1. What opinions do programming instructors and learners have about using placeholder-characters to improve the communication between them?
2. What characteristics could a placeholder have to better communicate its purpose?

The findings-discussions have been divided up into the two first following subtitles. Then discussed and compared in its entirety under 5.2.3 Combined findings discussed.

5.2.1 Teacher-group’s findings discussed

The first research question is about finding out what opinions programming instructors and tutors have about using placeholder-characters to further improve the communication of placeholders between them. There were mixed opinions about being able to standardize any type of placeholder-characters, but three out of four agreed that placeholders could be improved in its communication. K insisted that it isn’t a problem that needs to be dealt with, that placeholder-characters aren’t needed. And if this was a problem, it would be best dealt by meeting it with good education.

The Teacher-group’s participants were unanimous about the semantics of the placeholder-label being a crucial part of conveying that it is in fact, a placeholder, meant to be replaced by something else. For example, a placeholder-label in the sample code of Figure 14 (found in Exhibit C, Appendix 2 — Focus groups introduction) is: “lecture_php_to_do_app”. A better change to this label would render the placeholder more semantic and easily understandable. Like in Figure 15 (found in Exhibit C, Appendix 2 — Focus groups introduction) where the label is “name_of_database” instead. This was completely agreed upon in the Teacher-group. The labelling of placeholders should rather state what it should be replaced with, for the learner to understand its purpose. Three of the participants also thought that indicating some type of ownership in the placeholder, such as “YOUR_DATABASE_NAME”, would
bring a significant improvement to understanding where to replace something, with something. By using the word “Your”, the ownership is transferred to the one reading the code. Therefore, the reader should be able to understand that the placeholder is a placeholder. There could possibly be more usable words than “Your” that imply some sort of ownership or interchangeability, such as: “My”, “The”, “A” or “Change this”. The important role semantics of placeholder-labels plays, was not planned to be found in the first place, but rather manifested by the focus groups discussions and opinions.

The participants mostly agreed that indicating a placeholder using uppercase lettering and underscores (_) instead of spaces is good (called “screaming snake case”). That it usually stands out from other letter-cases and programming syntaxes so much that your eye would catch it faster. This is an easily followed guideline to give about placeholder-labelling and should be effortless to remember for programming instructors. This way a standard guideline for placeholders could be set, to improve the communication between programming instructors and learners. They also saw “two curly brackets” ({{ }}) as good candidates to be used as placeholder-characters, because these characters currently have a similar meaning in the templating process of some programming languages. In these programming languages for example: “Mustache”, “Handlebars.js”, “Angular JS”, “Django”, “Go” and Laravel’s “Blade”, two consecutive curly brackets are used in templates to later be replaced with some other content. Since two consecutive curly brackets already have a similar implication as this papers’ placeholders are desired to have and there is already a moderate acquaintance of these characters together in the programming community, it could be fitting to use these.

There are some things to be considered for the programming-software-tools that are used when displaying placeholders in code. Half of the group clearly expressed that it was very important how the placeholders are color-coded in the tools that display code. That it could be very confusing for beginners to understand what to replace, if the placeholder (which is supposed to be replaced) and the surrounding quotation marks (that are not supposed to be replaced) are in the same, or visually similar, color. If the quotation marks and the placeholder were in different colors it would be easier to separate the two. It was also registered that half of the group said that adding explanatory comments near the placeholder would make the placeholder easier to understand. Writing comments would enable a more thorough explanation of what is desired in the placeholder’s place, but could lack in visibility and has some restrictions when a comment is not available to give. For example, when a filename as in **Figure 4** contains placeholders, it is very hard to give an explanatory comment in the filename too. There could of course be comments around or even in the file itself. An explanatory comment could also come in the form of an introductory protocol as **B** suggested. Where the learner is given guidelines in the beginning of a course, book, forum post or other instructing mediums that set a common standard for the following code examples of the very same author. **K** agreed strongly to this.

5.2.2 Student-group’s findings discussed

There were mixed opinions about standardizing any type of placeholder-characters for improving the communication between instructors and learners, even if not considering the difficulty in establishing it. Four out of seven thought it would be a good way to solve the problem and one participant disagreed with this statement. The opinion of disagreement **D** had, was about placeholder-characters complicating things for the learner when there currently is no standardized placeholder-character. **D** also thought it would be better to write an explanatory comment for the placeholder
Instead. Two more participants agreed that explanatory comments would make placeholders easier to understand.

The group completely concurred that the semantics of the placeholder-label plays the most important role of understanding a placeholder. And that it is not always obvious if you should keep or remove the quotation marks around placeholders. That is partly what is meant to be solved by using placeholder-characters, but it also adds one more thing to be taught when learning code. They were also in agreement that placeholder-characters that aren’t universal (standard) would be more confusing than facilitative for learners. This was an eye-opener for the study. It was believed that placeholder-characters would make it easier to understand when to replace a placeholder even if the characters weren’t completely universal in their meaning yet. That it would have to be a universal standard to be understood was always a thought in the beginning of this study, that preferably there already were some standard characters that constantly were used as placeholder-characters. But that it wouldn’t be acceptable to imply the placeholder-meaning to some chosen characters if they weren’t used globally (from day one). This is of course a near impossible task to do, standards take often years until they are practiced or understood globally. Though all standards have one thing in common: They all started sometime. It needs to be considered if it is worth the time standardizing placeholder-characters universally or not.

Two participants stated that it is sometimes hard to understand when something needs to be replaced. Probably because of placeholders that are too hard to understand their purpose, through bad use of the placeholder-labels’ semantics, casing, placeholder-characters or related explanatory comments. Only one opinion was recorded about semantic ownership being a good way to imply a placeholder and that uppercase combined with underscores (_) is the most common way to indicate a placeholder. This was probably because of the small space these statements were given in the Student-group. The moderator was not fully prepared for these statements potential importance later in the analysis of the data.

5.2.3 Combined findings discussed

The Student-group had a lot of similarities with the Teacher-group. They both agreed completely that the semantics of the placeholder-label is crucial for the learners understanding of what to replace with and that it is supposed to be replaced. It is therefore utterly important that instructors think extra about what acts as a placeholder, and that its label is clear in its message to the learner. The choice of words in placeholders have a huge impact for the reader/learner. They also thought that if there is a placeholder-character that fills all or some of the following demands in Figure 7 (Not listed in any particular order).

- It is universally used (A standard).
- Made easily available (found on any keyboard).
- Shows what and what not to replace (keep quotation marks).
- Is understood by programming-software (displayed and compiled correctly).
- Not used for other purposes in coding already (has another implication).

Figure 7, Requirements for a viable placeholder-character.

If a placeholder-character fulfilled all the requirements, then it would be improving the communication between instructors and learners. If any of these aren’t met, then it wouldn’t be worth pursuing the notion of a standard placeholder-character. Something
Discussion and conclusions

e else to improve the communication of placeholders would be better to consider in that case.

Will two consecutive curly brackets ({{ }}) meet all the requirements in Figure 7?

- They are already used as templating placeholders for several well-used programming languages in web-development.
- Curly brackets are often easily found on keyboards and already in use in many programming languages. Therefore, typing one more curly bracket is not much more of a nuisance than writing the first one.
- It has been argued that using any type of placeholder-characters would not be making it any easier for the learner to understand what to replace and what not to. That it would still be hard to know that the quotation marks should almost always be kept when used in code. But it can also be argued that if the concept of this is taught out to beginning learners, it would actually help the understanding.
- Because of the curly bracket being a character that is widely used in many programming languages. It should be generally understood, displayed and compiled correctly by already following the set programming syntaxes. A placeholder often takes place inside a string where the content of that string usually isn’t “compiler-breaking”.
- This last requirement is the hardest to meet at the same time as the first requirement is met. Two consecutive curly brackets already have an implication in many languages which is divergent of this study’s placeholders’ implication. It would be hard to differentiate between the two various applications. Maybe a new definition to this study’s implication of placeholders could be used to differ from the current use of the placeholder-character, so that they are easier to keep apart. A suggestion is the word: “Replaceholder”. It implies that, it is not only a temporary place-holding word, but also that it should be replaced by the learner.

Because the first and last requirement are quite contradictory to each other, it is hard to fulfill them both at the same time. Two consecutive curly brackets come very close to meeting all requisites, but their current use isn’t researched enough yet to be standardized. This study was about finding out if there is a basic need for improving the communication between programming instructors and learners by standardizing the meaning “placeholder”. Not including a conclusive research on what placeholder-characters are to be used in a potential standard.

As mentioned before, the groups had mixed or unregistered opinions about using placeholder-characters to improve communication in code. Six participants thought that it would be a good improvement if placeholder-characters are used in code, when not considering how hard it would be to establish globally. One participant strongly disagreed that placeholder-characters would help, and four participants didn’t clearly state their stance on the matter. This means that, with this study’s data, it would not be fully backed up if a placeholder-character was chosen to be used as a standard in the conclusion.

The groups were either for or unstated about using ownership in the placeholder-label. That, words which indicate some kind of ownership to the placeholders’ reader, would be a great and differentiable enough characteristic for placeholders to be understood better, together with a semantically sound label (for example, using “YOUR_AGE” to explain briefly to the learner, that it is supposed to be replaced with their age). They
also mostly concurred with the statement considering using explanatory comments or introductory protocols to further ease the understanding of placeholders. Writing explanatory comments is not a huge nuisance for programming instructors when comparing it to how much more appreciation it would receive from learners. The same goes for introductory protocols that explain in the beginning of courses and books etc. rather than explanatory comments that are usually found in the near vicinity of the concerned placeholder(s).

The characteristics of a placeholder are discussed roughly in both focus groups. What is concluded is that; a placeholder usually has an informative semantic in the label, may contain words that implies ownership, may be in another letter casing compared to the surrounding text and may use placeholder-characters (there are many different placeholder-characters in use). If a placeholder doesn’t differentiate itself from the remaining code enough, it is hard to understand its intended nature. It is therefore essential that a placeholder should use any of these characteristics to imply the meaning “replace this”, where the most important characteristic is: the semantic of the placeholder-label itself.

5.3 Conclusions

Programming instructors and learners, from the School of Engineering at Jönköping University, concluded in some broad opinions about using placeholder-characters to improve communication between the two parties. They predominantly thought that using placeholder-characters, that aren’t globally established yet, would rather confuse than ease the understanding of what is to be replaced or not. It is therefore strategically better to focus on using a common characteristic for placeholders, than alternatively focusing on finding a standard placeholder-character. A placeholder is better off written with semantically sound labels that explain their purpose on their own, instead of using an input that only works for the instructor (Ex. in Figure 14 and Figure 15 in Exhibit C, Appendix 2 — Focus groups introduction). It is also recommended that placeholder-labels are to use a word that gives the ownership to the learner/reader or conveys the implication of an interchangeability. The most preferred word for this is “Your”, but there are several other words that could be utilized for a similar meaning too. This includes: “My”, “The”, “A”, “Insert”, “Here” or “Change this”. An example of a placeholder that applies this could look like: “Your_api_key”.

An instructor should use explanatory comments or give an introductory protocol when using placeholders. This means that if a placeholder is used in a code example; if it is in a course, book, forum post or any other applicable medium; it is easier for the learner to understand what a placeholder is in the corresponding text/code if there is some clarification given by the instructor, in the form of explanatory comments in the vicinity of the placeholder(s) or via an introductory protocol in the beginning (that illustrates the characteristics of placeholders in the authors code examples).

If a placeholder-character is to improve the understanding of placeholders’ purposes, it should meet all following requirements, restated from Figure 7:

- It is universally used (A standard).
- Made easily available(found on any keyboard).
- Shows what and what not to replace (keep quotation marks).
- Is understood by programming-software (displayed and compiled correctly).
- Not used for other purposes in coding already (has another implication).
Discussion and conclusions

It would be hard to meet all the requirements listed under Figure 7 currently and therefore the idea of a global placeholder-character is not considered viable at the time of writing.

The following (Figure 8, A code example that abides by the conclusions made in 5.3 Conclusions.) is a modified example of Figure 3’s code example and placeholder that complies with the conclusions written above.

```java
// Replace YOUR_SECTION_TITLE with the section title of your choice and keep the surrounding quote marks.
$SECTION_TITLE = 'YOUR_SECTION_TITLE';
```

Figure 8, A code example that abides by the conclusions made in 5.3 Conclusions.

Here, the comment (after “/”, in red) is quite explanatory in what is expected by the reader. The placeholder-label is defined to be replaced and an attempt to explain for the reader to keep the surrounding quotation marks is given. The placeholder-label itself (within the green quotation marks) is giving the ownership to the reader and is written using “screaming snake case” (Uppercase lettering with underlines instead of spaces) to deviate itself more from the rest of the code. Note that the conclusive example does not use placeholder-characters.

5.4 Further research

Because of the subjects relatively unexplored area, there are many factors available to research further upon. It is suggested to look for more detailed data about the current use of two consecutive curly brackets application in today's programming languages that use them to signify a templating placeholder. To find out if it could fulfill the last requirement of Figure 7, in a satisfactory way. Or if there are other potential placeholder-character candidates that meet the requisites better.

This study has now laid an introductory ground for further research in the same area. Because of the limited data available for generalization gathered in focus groups, it is suggested that a future study could apply the conclusions of chapter 5.3 to a more generalizable data collection method, perhaps through a quantitative survey that asks a larger population. It is also suggested to research on how to be able to standardize a new set of digital characters and how to implement it in IDE’s, text editors, websites and such (it may be easier in the future to implement new characters if computer keyboards were digital and able to change assigned keys dynamically. If so, this might be considered a good area to research about more). What should the potential placeholder-characters be named? Some suggested words that have come forth amidst this study’s research are: “Replaceholder” or “Da kine”, where the latter is a Hawaiian Pidgin expression and is close to our English placeholder name “whatchamacallit”. This study didn’t include much research on vertical placeholders that include a bigger amount of place-holding text than a one-lined string. It could be worth researching if there is a need for a specific characteristic in that type of placeholder, or what placeholder-characters that could be used in those cases.

In this report, the programming trade is primarily in the spotlight. But there are applications of placeholders outside programming that are not researched thoroughly
yet. To find out if placeholder-characters could and should be implemented for situations other than solely programming.

5.5 Implications

This study concludes in some guidelines for placeholders in code examples to abide by. Instructors that in any way use placeholders in code examples, should evaluate if the label is semantic and giving away ownership to the reader, they should preferably also give an explanatory comment above or below the placeholder in concern.

Businesses/people of influence that constantly use placeholders in code examples should consider having set a standard placeholder-characteristic or abide by the conclusions made in 5.3 Conclusions.
References


Hylander, I., 1998. *Fokusgrupper som kvalitativ datainsamlingsmetod,* s.l.: LIU.


7  Appendices

7.1 Appendix 1 — Focus groups scripts

Teacher-group

1 Welcome

1. Introduce myself
2. I won’t use any names in the report. You may be assured of complete confidentiality.
3. This will be recorded, is it okay with everybody? Record!
4. Will give an introduction first, then some ground rules are given, then questions are asked.

2 Overview of topic

1. Introduce the subject, Exhibit A & B
2. Explain code examples / sample code in Exhibit C
3. You were selected because you have in one way or another taught some programming in school.

3 Ground rules

1. Hand out papers and pens, explain
2. Not give short responses
3. No right or wrong answers, only differing points of view
4. It is ok to answer in Swedish if preferred. I will translate in that case.
5. We’re tape recording, one person speaking at a time
6. Use first names if needed when discussing

4 Questions

1. What is your name? For the transcription. (OPENING)
2. How have you been involved in teaching programming through the years? (INTRO)
   2.1. Taught via courses? online? forums? other?
3. Are you familiar with the term “placeholder”? (TRANSITION)
   3.1. What is a placeholder to you? Give an example?
   3.2. Do the others agree or disagree?
4. Have you ever used code examples in your teaching? (KEY)
   4.1. What was your experience with this? Did you find it difficult in any way? How?
5. Think back on your school years when you were strictly a learner. Have you ever been taught using code examples?
   5.1. Have you ever been taught using placeholders in code examples before?
   5.2. What was your experience with this? Did you find it difficult in any way? How?
6. In your teaching, have you ever used placeholders in your code examples?
   6.1. Do you prefer a way of displaying placeholder content? Please draw
   6.2. Would using placeholder-characters improve the communication between you (teachers) and learners (students)?
7. Anything else I should bear in mind considering placeholders and placeholder-characters?

5 Summary

1. Summarize with confirmation (ENDING)
2. Is this an adequate summary?
3. Is there anything that I might have missed?
4. Thanks, and dismissal
Student-group

1 Welcome (estimate 5’)
   1. Introduce myself
   2. I won’t use any names in the report. You may be assured of complete confidentiality.
   3. This will be recorded, is it okay with everybody? Record!
   4. Will give an introduction first, then some ground rules are given, then questions are asked.

2 Overview of topic (estimate 7’)
   5. Introduce the subject, Exhibit A & B
   6. Explain code examples / sample code in Exhibit C
   7. You were selected because you have learned some programming in school.

3 Ground rules (estimate 3’)
   8. Hand out papers and pens, explain
   9. Not give short responses
   10. No right or wrong answers, only differing points of view
   11. It is ok to answer in Swedish if preferred. I will translate in that case.
   12. We’re tape recording, one person speaking at a time
   13. Use first names if needed when discussing

4 Questions (estimate 23’)
   1. What is your name? For the transcription. (OPENING)
   2. Think back on your school years. How have you been involved in learning programming?
      (INTRO)
      2.1. Taught via courses? online? forums? other?
   3. Are you familiar with the term "placeholder"? (TRANSITION)
      3.1. What is a placeholder to you? Give an example?
      3.2. Do the others agree or disagree?
   4. Has your teacher ever used code examples? (KEY)
      4.1. What was your experience with this? Did you find it difficult in any way? How?
   5. Has your teacher ever used placeholders in their code examples?
      5.1. Was it always clear when a text should be replaced by you?
      5.2. What ways of displaying placeholder content have you encountered? Please draw
      5.3. Would using placeholder-characters improve the communication between instructors (teachers) and you (students)?
   6. Anything else I should bear in mind considering placeholders and placeholder-characters?

5 Summary (estimate 5’)
   14. Summarize with confirmation (ENDING)
   15. Is this an adequate summary?
   16. Is there anything that I might have missed?
   17. Thanks, and dismissal

To think about:
Control reactions to participants, Verbal and nonverbal, Head nodding
Avoid short verbal responses ("that’s good", "excellent")
Use pauses and probes (5 second pause)
"Would you explain further?", "Would you give an example?", "I don’t understand."
7.2 Appendix 2 — Focus groups introduction

Subject intro – Exhibit A

An example (Figure 9) of a current use of a placeholder from a code sample found on the online programming forum Stack Overflow where the learner is expected to replace ‘YOUR_API_KEY’ with something else, something unique/individual, in this case it could be replaced with ‘apiKey27901’ for example, to make it potentially work. Here the placeholder-characters are non-existent and the implication of “replace this” is only conveyed through its own characteristic text.

```
key: 'YOUR_API_KEY'},
```

Figure 9. Example of code sample found at http://stackoverflow.com/a/26065356/5415663

In comparison to how Microsoft make use of placeholder-characters in a specific code sample (Figure 10). They use less-than (LT) and greater-than (GT) signs to imply that the learner should replace the contained text (including the LT-, GT-signs) with something else here.

```
11 # Your account key goes here.
12 account_key = '<your account key>'
```

Figure 10. Example of code sample found at https://text-analytics-demo.azurewebsites.net/Home/SampleCode

The following (Figure 11) is an example where placeholders are used in digital file naming when a teacher wants to receive a digital file, named a certain way.

“When you hand in the assignment you should name it as: Javascript-Lab-1-FIRSTNAME_LASTNAME.js to simplify the grading process.”

Figure 11. Digital file naming example.

Here the text “FIRSTNAME” and “LASTNAME” is meant to be replaced with the students actual first name and last name, to end up in something like this: “Javascript-Lab-1-Sven_Parker.js”.

The problem here is that there isn’t a standard placeholder-character that instructors can conform to, a learner that uses sample codes like these preceeding ones can’t promptly know for sure what text currently is used as a placeholder. In certain cases it is also hard to demonstrate what naming convention or format the replaced text should be replaced with, because the text itself is used as a placeholder and not the placeholder-characters instead. This also applies to teachers who use code samples in their education.
Subject intro continued – Exhibit B

From this javascript code example in Figure 12 (from Google’s YouTube Data API):

```javascript
// Replace with section title of your choice.
var sectionTitle = 'YOUR_SECTION_TITLE';

var sectionTitle = 'Inspiring videos';
```

**Figure 12.** From original code to a desired replacement.

To this possible change in Figure 13, where [] are used as placeholder-characters:

```javascript
var sectionTitle = '[Section title]';

var sectionTitle = 'Inspiring videos';
```

**Figure 13.** From original code using placeholder-characters, to a desired replacement.
A code example/sample – Exhibit C

A code example is a bit of code given by an instructor to show a possible solution to the problem at hand.

**Figure 14** is what a code example could look like (PHP):

```php
<?php
// Connect to the database using object oriented style
$connection = new mysqli( 'localhost', 'root', '', 'lecture_php_to_do_app' );
?>
```

**Figure 14.** Code example from a teacher at School of Engineering, JU.

**Figure 15** is the same code example but with `{ }` used as placeholder-characters:

```php
<?php
// Connect to the database using object oriented style
$connection = new mysqli( 'localhost', 'root', '', '{name_of_database}' );
?>
```

**Figure 15.** Code example from a teacher at School of Engineering, JU.