Master Thesis

Title

Investigating Impact to Compilability of Python Code Snippets on Stack Overflow due to Python Version Upgrades

Supervisor
Professor Katsuro Inoue

Author
Shiyu Yang

2022/2/2

Software Engineering Laboratory, Department of Computer Science
Graduate School of Information Science and Technology, Osaka University

令和４年度 Master Thesis

Investigating Impact to Compilability of Python Code Snippets on Stack Overflow due to Python Version Upgrades

Shiyu Yang

Abstract

Stack Overflow is a popular Q&A site for programmers that have accumulated a wealth of code snippets. However, as time goes on, code snippets can become unavailable due to obsolete programming languages and other reasons. Such unavailable code may mislead answer seekers and cause unexpected problems. In December 2008, Python 3.0 was released. Python 3.0 is a new language version that is not backward compatible with Python 2. This means that code snippets written in Python 2 on Stack Overflow may not be compiled directly by Python 3. If not identified or documented clearly, it may mislead answer seekers and cause unexpected problems. In this work, we use PyComply to parse Python code snippets on Stack Overflow extracted from SOTorrent. Using the PyComply parsing results, we investigate the impact of the Python version upgrade on the compilability of Python code snippets on Stack Overflow. We found that Python version upgrades had an impact on the compilability of Python code snippets on Stack Overflow, as evidenced by: 1)about 40% of the Python code snippets on Stack Overflow whose compilability changed with the Python version in this study are uncompilable for Python 3. 2)The release of the new Python version inhibits the development of the old versions. 3)The trend of code snippets responding to newer versions increases over time.

Keywords

Stack Overflow
Python Version Evolution
Compilability

Contents

1 Introduction

Stack Overflow is a Q&A site for programmers, which provides a platform for programmers to exchange information and share knowledge. Here, users can post questions in the form of posts with different topics divided by titles and tags to make it easy for other users to search for content they are interested in. Users can also post answers as well as comments in the posts. The content of the posts is mainly text and code snippets. Users usually describe their questions or answers by attaching code snippets to their posts. As of January 2022[1], Stack Overflow has accumulated over 22 million posts, 33 million answers, and 84 million comments. These numbers are increasing every day.

This vast amount of information on the Internet has changed the way developers seek knowledge. Ready-to-use code snippets also provide developers with an easy way to find daily programming problems.

While it is convenient to search for available code snippets on Stack Overflow, recent studies have shown that code snippets can be toxic [10], obsolete [15] as well as low quality [14] and lead to software quality issues [15], license violations [2] or migration of security vulnerabilities [5], [13]. Therefore, more research needs to be done on identifying these problems and how to reduce the impact on Stack Overflow users. There are many reasons why code snippets in a Stack Overflow post become unavailable [15]. One of the reasons is the obsolescence of the code snippet due to the outdated programming language used in the code snippet [15].

Programming languages are not set in stone. Existing programming languages are constantly evolving to meet new needs and thus gain longevity. Popular programming languages often use versions to indicate their evolution, with newer versions usually representing more mature forms of the language. Many programming languages address language evolution by maintaining backward compatibility, which means that programs compiled with an earlier language version can be compiled with a later version and exhibit the same behavior as the previous version. However, the release of Python 3.0 made the Python language break this rule; Python 3 series versions are not backward compatible with Python 2 series versions. The lack of backward compatibility for Python 3 can cause problems for Stack Overflow users. For example, a user found a code snippet that meets his/her needs and wants to reuse it in his/her project. However, this code snippet is written in Python 2 and may not be compilable if used directly in the user's project written in Python 3. Therefore, it is necessary to provide insights on how to track or alleviate this problem.

To our knowledge, no studies have investigated the Python language versions of Python snippets

[1] `https://data.stackexchange.com/`

on Stack Overflow. We are interested in investigating the compilability of Python code snippets on Stack Overflow for each Python version, and what impact Python version upgrades have on the compilability of Python code snippets. Knowledge of the compilability of Python code snippets may provide insight into new research directions and tool support.

In this paper, we extracted 2,475,559 code snippets from SOTorrent whose post tag contained "Python" (we consider such code snippets as code written in Python) and filtered 307,788 code snippets whose compilability changed with the Python version for research. We used a Python compliance analyzer, PyComply, to parse the code snippets we studied to understand the impact to compilability of Python code snippets on Stack Overflow due to Python version upgrades. We organized our study by answering the following research questions:

- **RQ1:What are the available Python version ranges for Python code snippets on Stack Overflow?**

  We found that this study's available version ranges of Python code snippets whose compilability varies with Python version changes in this study can be divided into three categories. The code snippets whose available version range spans Python 2 and Python 3 account for 42.0% (307,788 in total), those whose available version range is only in Python 2 account for 39.6%, and those whose available version range is only in Python 3 account for 18.4%. Code snippets are relatively concentrated in the range of versions containing 2.7 and 3.6.

- **RQ2:How Stack Overflow users respond to each Python release, and what are the differences between Python 2 and Python 3?**

  We found that new versions are released and get a response shortly afterward. The release of new Python version inhibits Stack Overflow users ' response to older versions. It is impossible to compare the differences in user response to each version of Python 2 and Python 3 releases.

- **RQ3:How are Python 2-only compilable Python code snippets and Python 3-only compilable Python code snippets distributed on Stack Overflow?**

  The number of Python 2-only compilable Python code snippets increased year by year starting in 2008, peaked in 2015 and began to decline year by year after that. There is an overall upward trend in Python 3-only compilable Python code snippets. But overall, among the code snippets investigated by RQ3, the Python 2-only compilable Python code snippets are about twice as large as the Python 3-only compilable Python code snippets.

Paper Organization: The rest of the paper is organized as follows. Section 2 presents the background. Section 3 introduces our study approach. Section 4 presents the results of our research questions. Section 5 discusses the implications and limitations of our study. Finally, Section 6 concludes the paper.

2 Background

2.1 Stack Overflow

Stack Overflow is a popular Q&A site for programmers. Developers can post questions, answer questions, and search and browse for the content of their interest on the site. When posting a question on Stack Overflow, questioners can indicate to viewers what the question is about by adding information such as a title and tags. In addition, code snippets and other references (e.g., URLs or images) related to the question can be pasted into the body of the post, allowing the questioner to describe the question in detail. Each question can receive multiple answers from different answerers. Answerers can also post answers with code snippets and other references to explain their answers. Figure 1 shows an example of a post on Stack Overflow. The questioner asked how to change the name of objects in Django admin and attached a code snippet and image to detail the question.

In the process of posting and answering questions, Stack Overflow collects a significant number of code snippets. Developers often try to find code snippets on Stack Overflow that meet their needs to reuse in their projects. Regarding these code snippets on Stack Overflow, Wu et al. [14] studied how developers utilize source code from Stack Overflow. They found that one of the top 3 barriers that make it difficult for developers to reuse code on Stack Overflow is low code quality. From this point, Stack Overflow needs to improve or verify the quality of code snippets.

Zhang et al. [15] conducted an empirical study of obsolete answers on Stack Overflow. As time passes, specific knowledge in answers may become obsolete. If not identified or documented clearly, such obsolete answers may mislead answer seekers and cause unexpected problems. Therefore, it is necessary to provide insights on how to track or alleviate this problem.

2.2 SOTorrent

SOTorrent is an open dataset based on the official Stack Overflow data dump [4]. The official Stack Overflow data dump collects and organizes questions, answers, and user information on Stack Overflow in posts units.

SOTorrent provides access to the version history of Stack Overflow content at the level of whole posts and individual text or code blocks [4]. A post can contain both text and code blocks, depending on how the author formats the content. In our study, the definition of Stack Overflow code snippets refers to code blocks.

SOTorrent contains all tables from the official Stack Overflow data dump [3], marked in gray

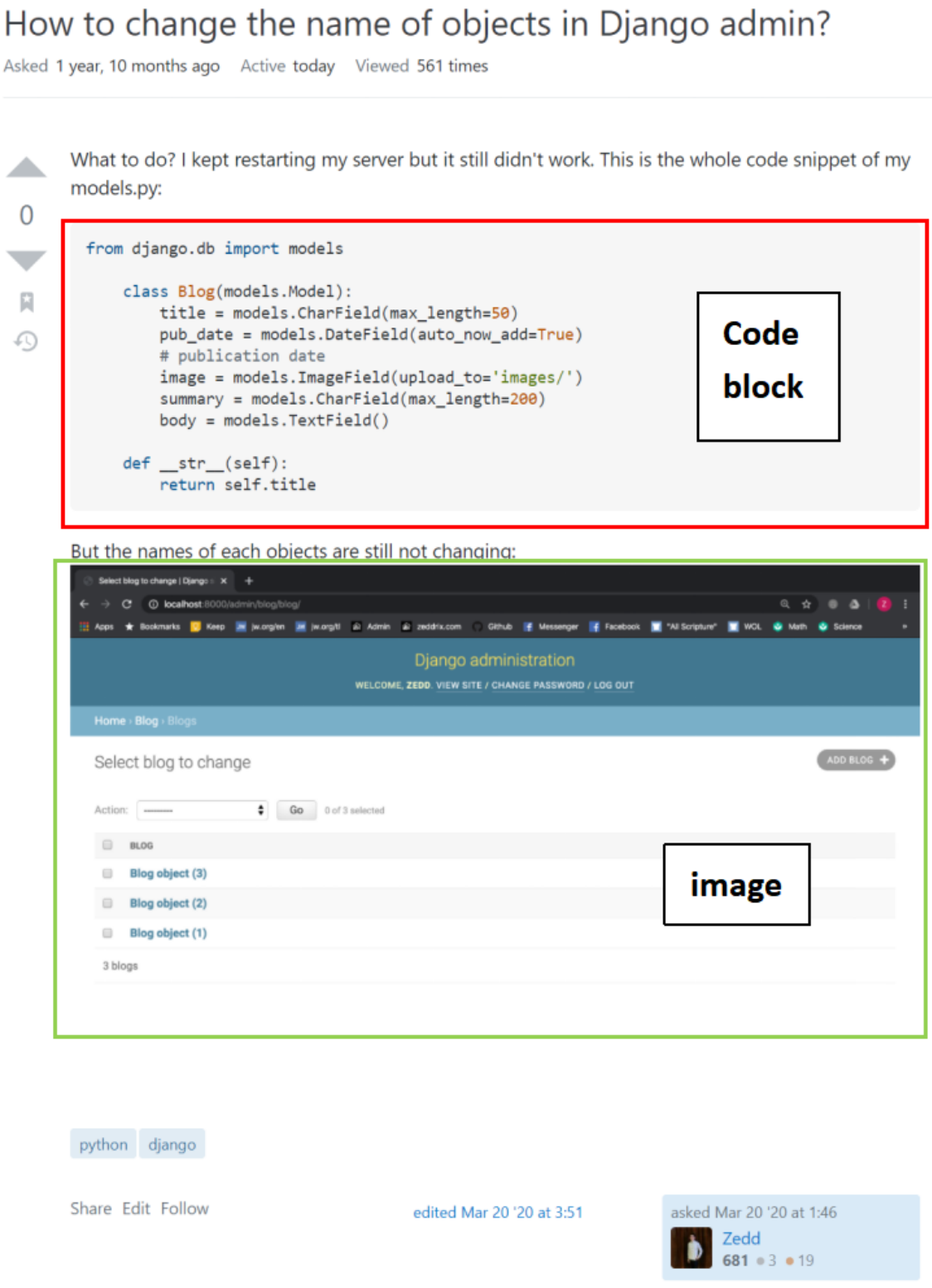


Figure 1: An example of a question and its accepted answer on Stack Overflow (https://stackoverflow.com/questions/60767482/how-to-change-the-name-of-objects-in-django-admin)

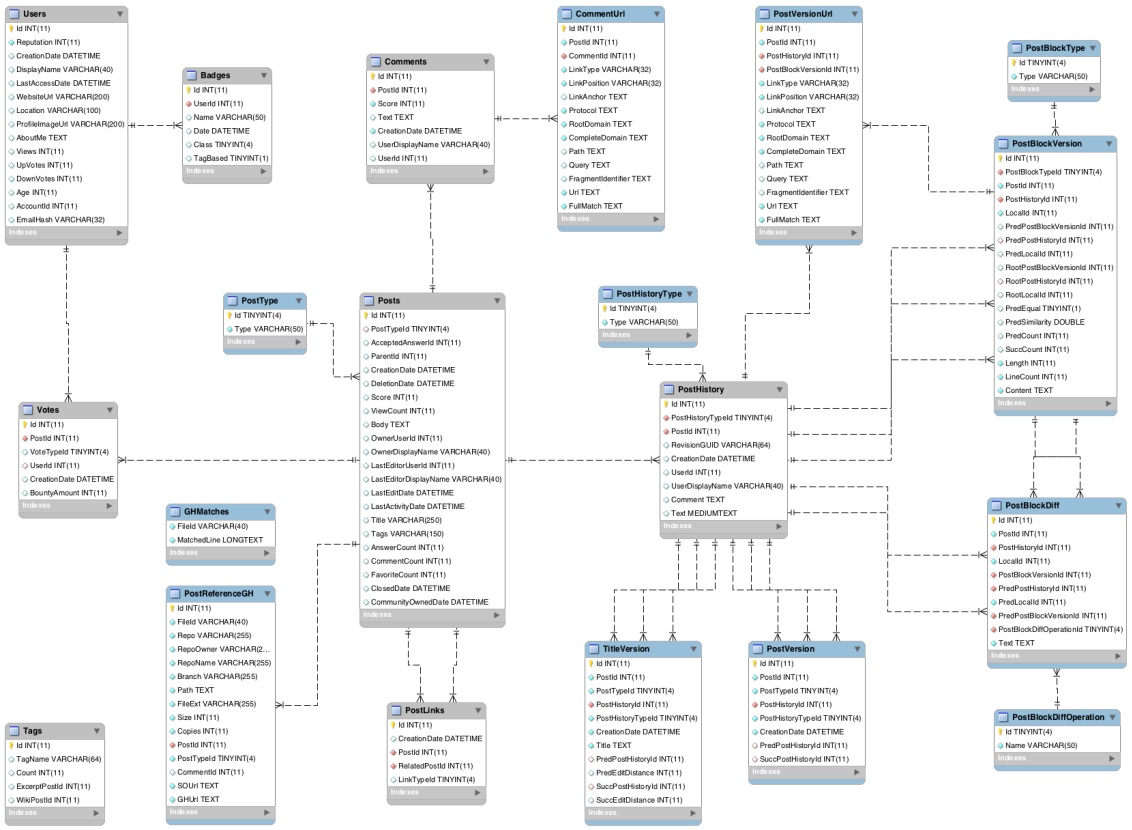


Figure 2: Database schema of SOTorrent [3]

in the SOTorrent database schema as shown in Figure 2. Additionally, the tables created by SOTorrent authors based on the official data dump tables are blue. In the following sections, we will describe how to obtain data from the relevant tables in SOTorrent for our study.

Several other studies have used SOTorrent. Manes et al. [7] investigated how often GitHub developers re-use code snippets from the SO forum, as well as what concepts they are more likely to reference in their code. For their goal, they mined the SOTorrent dataset that provides connectivity between code snippets on the SO posts with software projects hosted on GitHub.

Manes et al. [8] investigated the change histories of snippets on Stack Overflow and GitHub. We studied SO snippets that have been reused in GitHub projects by mapping data sources, SOTorrent and GHTorrent, and building a new dataset that provides a mapping of these data to the revision history of the reused code snippets along with code diffs.

Rahman et al. [11] investigated how often insecure responses appeared in Stack Overflow posts related to Python. In this study, SOTorrent was used to extract code snippets and identify approved responses.

Abric et al. [1] presented an initial exploration into the value of duplicate questions and their answers on Stack Overflow, using SOTorrent to investigate the nature of users who ask duplicate questions by knowing their past credibility and activity.

2.3 Reasons for the obsolescence of code snippets on Stack Overflow

Zhang et al. [15] manually derived and categorized the reasons why answers on Stack Overflow are obsolete, as shown below:

(1) Third Party Library: An answer becomes obsolete due to third-party libraries, Application Programming Interfaces (APIs), or frameworks becoming obsolete.

(2) Programming Language: Answer obsolescence is caused by obsolete programming language features and/or its standard APIs.

(3) Reference: References in an answer are obsolete.

(4) Tool: Tool information is obsolete, such as an old version.

(5) Mobile OS: An answer becomes obsolete due to an obsolete mobile platform.

(6) Non-mobile OS: An answer becomes obsolete due to an obsolete nonmobile OS platform.

(7) Protocol: An answer is obsolete because a protocol is updated.

Most of the above also apply to analyzing and explaining the reasons for the obsolescence of code snippets on Stack Overflow. Taking the Third Party Library as an example, APIs are often used in software development because they allow developers to call commonly used functions from external programs. There are many posts on Stack Overflow about problems encountered when using APIs. API developers can also use these posts to understand how users use the API, and what features they want the API to provide. However, the APIs included in those posts are not always correct in the latest version of the library. Those obsolete APIs are detrimental to users who want to reuse the code snippets. For example, users find a code snippet in a Stack Overflow post that meets their needs and reuse it in their project. However, the user may not realize that the APIs used in the code snippet are outdated. Using such outdated APIs has the potential to cause software quality problems.

Nishimura [9] investigated whether the APIs in the code snippets included in the Stack Overflow posts were available in the latest library versions, targeted APIs included in five Java libraries. As a result, they found that many of the invalid code snippets containing deprecated APIs were no longer valid shortly after 2017.

2.4 Python Language Evolution and Backward Compatibility

Programming languages need to evolve in response to external and internal factors continually. Without evolution, the language may become less competitive and even deprecated. Versions are a common way for popular programming languages to express evolution, and later versions usually represent a more mature form of the language. Many popular programming languages have maintained backward compatibility during their evolution to avoid raising concerns among developers about software product compatibility, meaning that software developed in an earlier version of the language can be compiled with a later version and express the same behavior as the previous version.

However, Python breaks the backward compatibility rule and is one of the notable few exceptions. Python 3 versions starting with 3.0 are not backward-compatible with Python 2 versions from 2.0 to 2.7. In other words, programs developed in Python 2 may not be interpreted by Python 3 without modification.

With its simplicity, readability, and extensibility, the Python versions has grown linearly since 2004, and many programmers have welcomed and loved it. Beginning with version 2.0 released in October 2000, the Python language continued to evolve until version 2.5. Python 2.6 was then released in October 2008, followed by Python 3.0 in December of that year. Python 3.0 is not backward compatible with previous Python versions. Despite the release of Python 3.0, the Python team decided to support both development branches due to the large number of users continuing to use Python 2. Since then, versions of the Python language have been split into two series.

In addition, the Python team announced in 2008 that Python 2 would be supported until 2015. Python 2.7 was released in July 2010 as the last version of the Python 2 series. However, in 2014, the date for ending support for Python 2 was extended to 2020 to accommodate those who were still unable to migrate to Python 3. Eventually, official support for Python 2.7 ended on January 1, 2020. This marks the end-of-life of Python 2. With official support for Python 2 no longer available, many projects also announced to stop supporting Python 2 (including TensorFlow, Pandas, Numpy, Jupyter Notebook, Cython, etc.). If you're still using Python 2, it's time to upgrade to Python 3. Suppose Python users continue to use unsupported modules. In that case, you are taking a risk with the security of your organization and data, because vulnerabilities will appear at some point, and no one will fix them.

Also, as mentioned before, a large number of code snippets have been accumulated on Stack Overflow. There may be code snippets written in various Python versions among these code snippets. When Python 2 is completely deprecated in the future, code snippets written in Python 2

on Stack Overflow will also become obsolete. Therefore, it is necessary to investigate the Python code snippets on Stack Overflow to determine their availability for each Python version.

2.5 PyComply

To investigate the impact of the transition from Python 2 to Python 3 on Python applications, Malloy et al. [6] developed a Python compliance analyzer, PyComply. It is based on an approach that exploits grammar convergence to generate parsers for each of the major versions in the Python 2 and Python 3 series and conducted an empirical study on the Qualitas corpus, a selection of Python applications.

Input to PyComply is the Python grammar for the version under study together with a Python program or test case; output from the tool includes the statistical information such as pass/fail result for each file or the number of Python 3 features that PyComply recognized. The core of PyComply is the grammar formalism used to define the Python syntax, along with the parser actions inserted into the grammar to facilitate identifying the Python 3 features.

The Python developers make a test suite available for each Python version. In addition to using correctness preserving grammar transformations to build their parsers, they also validated the parsers by comparing the number of test cases that their PyComply parsers pass with the number of test cases that the Python parsers passed numbers were the same. Moreover, the fact that their parsers recognize the same test cases that the Python parsers recognize substantiates the validity of their investigation.

3 Study Approach

This study wants to know how the compilability of python code snippets on Stack Overflow has changed due to python version upgrades. Therefore, we surveyed the compilability of python code snippets on Stack Overflow, focusing on the following three three research questions:

(1) RQ1:What are the available Python version ranges for Python code snippets on Stack Overflow?

(2) RQ2:How Stack Overflow users respond to each Python release, and what are the differences between Python 2 and Python 3?

(3) RQ3:How are Python 2-only compilable Python code snippets and Python 3-only compilable Python code snippets distributed on Stack Overflow?

In designing the study approach to answering these research questions, we first utilized SOTorrent, mentioned in Section 2, to extract our desired Python code snippets from Stack Overflow. In the second step, we preprocessed the Python code snippets and parsed the Python code snippets we studied using PyComply, as mentioned in Section 2. Finally, we explain each research question regarding the purpose of the study and the definitions required for the experiment.

3.1 Data Collection

This subsection describes how we gathered the dataset to address our research questions.

3.1.1 Data source

To understand the impact of Python version upgrades on the availability of python code snippets on Stack Overflow, we first need to obtain code snippets written in Python among the code snippets included in the questions and answers posted on Stack Overflow.

As we introduced in Section 2, when a questioner posts a question on Stack Overflow, he can show viewers the relevant content of the question by adding information such as title and tags. Also, due to editing, the content of the currently posted code snippet may differ from that of the original posting. In addition, the content of the now posted code snippet may vary from that of the initial posting due to editing. To investigate the compilability of the current code snippet, we focused on the most recent posted data currently available to users. Based on these, we used the following two criteria to identify the code snippets required for this study:

(1) Code snippets from posts containing the tag ”Python”.

Table 1: Posts

Field Name	Data Type	Interpretation
Id	Integer	PostID
PostTypeId	TinyInt	ID to identify the type of post
CreationDate	DateTime	Date of post
Body	Text	Body of a post
Tags	VarChar(150)	Tags of post

(2) Posting data is up to date.

We use SOTorrent to get Python code snippets on Stack Overflow. SOTorrent has been continuously updated with many versions since its creation. At the time we started our study, the latest version of SoTorrent was SOTorrent20_03 as of March 15, 2020.

3.1.2 Data Processing

We first built a database using the SOTorrent dataset to get Python code snippets from Stack Overflow posts. In this study, we use the data stored in Posts, PostBlockVersion, and PostBlockType in the tables shown in Figure 2. The contents of these three tables[2] are as follows:

(1) Posts: HTML content of most recent version of Stack Overflow posts.

(2) PostBlockVersion: Version history on the level of post blocks, which are either text (1) or code (2) blocks (extracted from Table PostHistory).

(3) PostBlockType: Available post block types (text or code).

We do not need to use all the fields of these tables. Tables 1, 2, and 3 show the fields in each table used in this study and their interpretations.

After constructing the database, we extracted code snippets from all posts containing a tag "Python" from the database. We created a new table PythonPost in the database to store the following information about these code snippets: the posting date of the post including the extracted code snippet, the body of the post, the content of the code snippet, the tags attached to the post, and whether the post is the latest version or not. Finally, we extracted the code snippets we needed from this table, totaling 2,475,559.

[2] https://empirical-software.engineering/sotorrent/

Table 2: PostBlockVersion

Field Name	**Data Type**	**Interpretation**
Id	Integer	Version ID of the post
PostBlockTypeId	TinyInt	The ID of the post type
PostId	Integer	Post ID that is the source of the post version
Content	Text	Content of a post
MostRecentVersion	Boolean	Whether it is the latest after editing

Table 3: PostBlockType

Field Name	**Data Type**	**Interpretation**
Id	TinyInt	The ID of the post type
Name	VarChar(50)	Name of the post type

3.2 PyComply Analysis

This subsection introduces how PyComply parses a Python code snippet to identify its Python version. Next, we present PyComply analysis in two parts, preprocessing and PyComply parsing.

3.2.1 Preprocessing

As mentioned in Section 2, PyComply was initially designed to parse Python files in applications. Our research objects are individual code snippets on Stack Overflow. To accommodate our research needs, we first need to perform a simple preprocessing of the code snippets we obtained in the previous steps. The steps are as follows：

(1) Select the code snippets with the *MostRecentVersion* value of ”true” from PythonPost and save them as separate python files.

(2) Handle formatting issues that cause code snippets not to be parsed by PyComply: redundant indentation, inconsistent indentation, etc.

In addition, before the formal PyComply parsing, we need to build the corresponding PyComply for all Python 2 and Python 3 versions investigated in this study, i.e., generate and build the relevant parsers and scanners for each Python version.

When Malloy et al. [6] ran PyComply for the eight major releases in Python 2 and the seven major releases in Python 3 and examined the pass rates for the applications, they found that the

pass rates for versions 2.1, 2.3, 3.2, and 3.4 were the same as the corresponding previous version for all applications. To investigate whether a similar situation might arise when parsing Python code snippets on Stack Overflow with PyComply, we performed a PyComply parsing test. We ran PyComply for the same eight major releases in Python 2 and the seven major releases in Python 3. When we examined the pass rates of Python code snippets, we found that the pass rates for versions 2.1, 2.3, 3.2, and 3.4 were the same as the corresponding previous version for all Python code snippets. This result is consistent with the findings of Malloy et al. [6]. Therefore, we omitted these versions in this study, showing only the six and five major versions of Python 2 and Python 3.

3.2.2 PyComply parsing

In this study, a code snippet is considered compilable for a Python version if it can pass PyComply parsing for that Python version. Otherwise, the code snippet is deemed uncompilable for that Python version.

We parsed 2,475,559 code snippets obtained earlier with PyComply corresponding to each Python version, and recorded the pass/fail results of each snippet as shown in Table 4. We removed 1,173,905 code snippets during this process because they could not be compiled as Python code snippets.

The first column of Table 4 is the posting year of the python code snippets. To facilitate the observation of the change of python code snippets on Stack Overflow over time, we group the python code snippets based on the year of posting. One row for each year and one column for each Python version, ranging from 2.0 to 2.7 in series 2 and from 3.0 to 3.6 in series 3. The last column shows the number of Python code snippets in each year. The data in all but the first and last columns show the percentage of these Python code snippets that passed PyComply for the corresponding year.

According to the data in Table 4, we found that the Python code snippets have high pass rates for all Python versions. This indicates that a significant fraction of the code snippets we parsed were compilable for all Python versions. This may be the case because they lack some features that would make the compilability of code snippets vary between Python versions. While these code snippets are useful for users, this study focuses on code snippets that are (only) compilable in certain versions, i.e., code snippets whose compilability changes depending on the version of Python. Additionally, we found that the magnitude of change in the data in the table is too small to easily observe changes in code snippets over time or Python versions. Therefore, we believe

Table 4: Pass rates for Python code snippets for each year for Python versions 2.0 through 3.6

Year	ver2.0	ver2.2	ver2.4	ver2.5	ver2.6	ver2.7	ver3.0	ver3.1	ver3.3	ver3.5	ver3.6	# code snippets
2008	92%	93%	95%	97%	97%	98%	82%	82%	83%	83%	83%	917
2009	91%	92%	94%	96%	96%	97%	78%	78%	80%	80%	81%	7,588
2010	90%	91%	94%	96%	96%	96%	80%	80%	81%	81%	82%	17,055
2011	90%	91%	93%	96%	96%	97%	79%	79%	80%	80%	81%	30,619
2012	89%	90%	92%	95%	96%	97%	79%	79%	80%	80%	81%	54,062
2013	88%	89%	92%	95%	96%	97%	80%	80%	81%	81%	82%	91,413
2014	87%	88%	91%	95%	96%	96%	81%	81%	83%	83%	84%	116,229
2015	86%	87%	90%	95%	96%	96%	83%	83%	85%	85%	86%	141,352
2016	85%	86%	89%	95%	95%	96%	86%	86%	87%	87%	88%	169,301
2017	84%	85%	88%	94%	95%	96%	90%	90%	91%	92%	93%	210,517
2018	83%	84%	87%	93%	94%	95%	94%	94%	94%	95%	97%	233,781
2019	83%	84%	86%	92%	93%	94%	95%	95%	95%	96%	98%	195,983
2020	82%	83%	86%	92%	93%	93%	95%	95%	95%	96%	99%	32,837

that excluding code snippets compilable for all Python versions can make the changes in the data clearer when studying specific research questions. We removed 993,866 code snippets that were compilable for all Python versions. The Pass rates for the remaining 307,788 code snippets are shown in Table 5. The first column of Table 5 is the posting year of the python code snippets. One row for each year, and one column for each Python version, ranging from 2.0 to 2.7 in series 2 and from 3.0 to 3.6 in series 3. The last column shows the number of Python code snippets in each year. The data in all but the first and last columns show the percentage of these Python code snippets that passed PyComply for the corresponding year.

3.3 Case Study

As we described in Section 2, in 2008, the successive release of Python 2.6 and Python 3.0 started the path of branching Python versions. Python 2 and Python 3 are no longer released linearly like other programming languages. Therefore, when investigating the research questions, we analyzed the timelines of Python 2 and Python 3 separately to avoid the interference caused by the overlap in the release schedule of the Python 2 and Python 3 series.

In addition, we used the same dataset when investigating RQ1, RQ2, and RQ3. We obtained 307,788 code snippets by filtering out code snippets that are compilable for all Python versions and those that are uncompilable for all Python versions, as mentioned earlier. Table 6 records the pass and fail results of the dataset parsed by the PyComply of each Python version.

Table 5: Pass rates for Python code snippets for each year for Python versions 2.0 through 3.6 (Removed the code snippets compilable for all python versions)

Year	ver2.0	ver2.2	ver2.4	ver2.5	ver2.6	ver2.7	ver3.0	ver3.1	ver3.3	ver3.5	ver3.6	# code snippets
2008	68%	72%	81%	89%	89%	90%	25%	25%	28%	28%	30%	220
2009	68%	70%	80%	85%	86%	88%	23%	23%	29%	29%	31%	2,150
2010	65%	67%	76%	84%	85%	87%	26%	26%	30%	30%	33%	4,689
2011	65%	67%	76%	85%	86%	88%	25%	25%	30%	30%	32%	8,674
2012	62%	64%	74%	84%	86%	88%	27%	27%	31%	31%	35%	15,682
2013	59%	61%	71%	84%	86%	88%	30%	30%	34%	34%	38%	25,980
2014	54%	56%	67%	82%	85%	87%	33%	34%	38%	38%	42%	32,392
2015	48%	51%	63%	81%	84%	86%	38%	39%	43%	43%	47%	38,354
2016	42%	45%	57%	79%	82%	85%	45%	45%	49%	50%	54%	42,720
2017	29%	33%	47%	74%	78%	82%	57%	58%	61%	63%	68%	47,501
2018	16%	20%	35%	65%	70%	74%	67%	68%	70%	74%	83%	46,476
2019	9%	13%	28%	59%	64%	69%	72%	72%	74%	78%	91%	36,783
2020	5%	10%	25%	55%	61%	65%	72%	73%	74%	77%	95%	6,167

Table 6: PyComply parsing results for Python code snippets for Python versions 2.0 through 3.6(Removed the code snippets compilable and uncompilable for all python versions)

Version	Failed	Passed	# code snippets
ver2.0	193,302	114,486	307,788
ver2.2	183,894	123,894	307,788
ver2.4	145,069	162,719	307,788
ver2.5	77,753	230,035	307,788
ver2.6	66,748	241,040	307,788
ver2.7	56,880	250,908	307,788
ver3.0	158,135	149,653	307,788
ver3.1	156,784	151,004	307,788
ver3.3	145,660	162,128	307,788
ver3.5	141,555	166,233	307,788
ver3.6	122,155	185,633	307,788

3.3.1 RQ1:What are the available Python version ranges for Python code snippets on Stack Overflow?

This study would like to investigate the lifespan of SO code snippets that are not compilable in some Python versions. The available python version ranges for Python code snippets may represent the survival period of code snippets to some extent.

One thing to note about the available python version ranges for Python code snippets is that the available python version ranges may not be continuous but are subject to breaks. For example, if a code snippet is compilable for Python 2.4, uncompilable for Python 2.5, but compilable for Python 2.7. The available python version range for that code snippet is 2.4-2.7. Just because a code snippet is uncompilable for a newly released Python version does not mean that it is entirely deprecated, and there are many other Python versions available simultaneously. We only need to focus on the earliest available Python version and the latest available Python version for that code snippet. Based on the above, we define the criteria for available Python version ranges as follows:

(1) The available version range of a code snippet is from the oldest python version to the latest python version that the code snippet can pass PyComply parsing.

(2) There can be breaks between available python version ranges.

Figure 3 is a schematic diagram of the available Python version ranges for Python code snippets in this study, with the left and right graphs representing Python 2 and Python 3, respectively. The two series do not interfere with each other. The available version ranges in Python 2 are observed without considering the compilability of Python code snippets for each version of Python 3 and vice versa. The first row of both graphs shows the Python versions in the figure, each marked with a color. A rectangular block of color represents a range of Python versions in the rest of the charts, marked with the color corresponding to the Python version that the range starts with. There are 21 possible ranges in the Python 2 series and 15 possible ranges in the Python 3 series. Taking Python 2 on the left as an example, the available version range represented by the orange-colored part of line eight is 2.2-2.4. Python code snippets in this range are not compilable for Python 2 versions before Python 2.2 and after Python 2.4, but only for Python 2 versions in the range 2.2-2.4.

Based on the above criteria, we investigate the available version ranges of Python code snippets within the two series Python 2 and Python 3, respectively. Only the respective series' versions are considered as these version ranges are obtained. Those code snippets whose available python version ranges span Python 2 and Python 3 will get the version range within Python 2 and within Python 3, respectively. We combine the version ranges in the respective series of Python 2 and

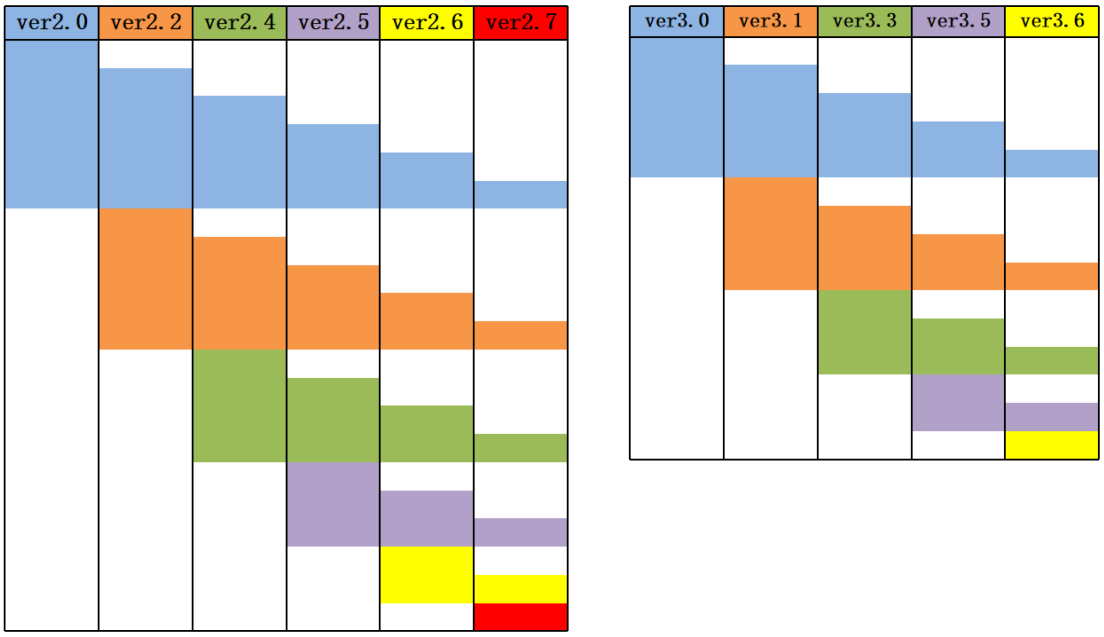


Figure 3: Schematic diagram of the available Python version ranges

Python 3 in pairs. If there are duplicates of code snippets in the Python 2 range and Python 3 range in a combination, it means that the code snippets in those duplicates are compilable for both Python version ranges. This way, we can get Python code snippets that are compilable across Python 2 and Python 3.

When we combine the available version ranges of Python 2 and Python 3, we use forms like [2.0,3.0-3.3] to represent our combinations of version ranges within the Python 2 and Python 3 series. [2.0,3.0-3.3] means that the code snippets are compilable for Python 2.0 and versions in the range Python 3.0 to Python 3.3. We use ∅ to represent that there are no Python 2 versions available or no Python 3 versions available.

These combinations can be divided into three categories:

(1) Ranges span Python 2 and Python 3：contains both the range of Python 2 and Python 3, such as [2.0,3.0-3.3]

(2) Only in Python 2：Contains only the range of Python 2, not the range of Python 3, like [2.0, ∅]

(3) Only in Python 3：Contains only the range of Python 3, not the range of Python 2, like [∅, 3.6]

Table 7: The release date of each version of Python

Version	Release Date
v2.0	2000.10.16
v2.2	2001.12.21
v2.4	2004.11.30
v2.5	2006.09.19
v2.6	2008.10.1
v2.7	2010.7.3
v3.0	2008.12.3
v3.1	2009.6.27
v3.3	2012.9.29
v3.5	2015.9.13
v3.6	2016.12.23

3.3.2 RQ2:How Stack Overflow users respond to each Python release, and what are the differences between Python 2 and Python 3?

We wondered how authors of Python code snippets on Stack Overflow would react and choose when faced with the constant release of new versions of Python. And how this reaction behaves differently between Python 2 and Python 3. By investigating the authors' responses of Python code snippets on Stack Overflow to each version, we can get a sense of how Python versions are trending and how they interact with each other.

To do this, we first need to determine the release date of each python version. As shown in Table 7, we obtained the release date of each python version from Python.org[3]. The first column in the table is the Python version, and the second column is the Python version release date.

To observe how Stack Overflow users respond to each Python release, we can investigate how the Python code snippets responding to that Python version have evolved since each Python version release. We refer to the Python code snippets that are compilable for a certain Python version after it is released as the code snippets responding to that version, which is defined in detail as follows:

(1) The posting date for code snippets responding to a certain Python version needs to be after the release of that Python version.

(2) Code snippets that respond to a certain Python version need to be compilable for that version,

[3] `https://www.python.org/`

and uncompilable for the Python versions before that version. As shown in Figure 3, the available python version ranges for code snippets responding to Python 2.4 are within the four ranges in green, starting with 2.4.

Since it is impossible in this study to determine whether code snippets that are compilable for Python 2.0 are uncompilable for Python versions before Python 2.0, therefore, we do not investigate Python 2.0 in this research question.

3.3.3 RQ3:How are Python 2-only compilable Python code snippets and Python 3-only compilable Python code snippets distributed on Stack Overflow?

While developers can upgrade to Python 3, code snippets written in Python 2 on Stack Overflow may not be modified by their authors to make them compilable for Python 3. While studying obsolete answers on Stack Overflow, zhang et al. [15] found that only a tiny proportion of such answers are updated afterward when an obsolete answer is identified. Soni et al. [12]. explored how comments affect answer updates on Stack Overflow, using the SOTorrent dataset. Their results show that a large number of answers on Stack Overflow are not updated, even when they receive comments that warrant an update.

Therefore, we need to find out if Python 2-only compilable Python code snippets on Stack Overflow. If so, compare it to the Python 3-only compilable code snippets and analyze the differences. As shown in Table 7, the set of contents in the last column is the Python 2-only compilable Python code snippets, i.e., Python code snippets are compilable for Python 2 and uncompilable for Python 3. The set of contents in the last row is the Python 3-only compilable Python code snippets, i.e., Python code snippets that are compilable for Python 3 and uncompilable for Python 2.

4 Case Study Results

We present the results of our analysis and provide answers to the research questions.

4.1 RQ1:What are the available Python version ranges for Python code snippets on Stack Overflow?

To investigate the range of Python versions available for Python code snippets on Stack Overflow, we need to obtain the value of each of the available version range combinations for Python 2 and Python 3.

First, We investigated the available version ranges of Python code snippets within the two series, Python 2 and Python 3, respectively. The results are shown in Table 8, Table 9. The first column of Table 8 and Table 9 shows the available Python version ranges in the Python 2 series and the available Python version ranges in the Python 3 series, respectively. The second column is the number of Python code snippets in each range.

Table 8: The available python version range for Python code snippets in Python 2

Version Range	# code snippets
2.0	9
2.0-2.4	139
2.0-2.5	22
2.0-2.7	114,316
2.2-2.4	1
2.2-2.7	9,488
2.4	11
2.4-2.5	7
2.4-2.7	38,807
2.5	9
2.5-2.7	67,386
2.6-2.7	11,043
2.7	9,868

Figure 4 provides a more intuitive view of the distribution of code snippets across the available python version ranges. The first row of the two diagrams shows the individual Python versions in the figure. A rectangular block of color represents an available Python version range in the rest of the diagram. We divided the number of Python code snippets within each available Python version range into six groups, marked with different colors. The correspondence between the number of code snippets and the colors is shown in the legend on the right side of the figure. For example,

Table 9: The available python version range for Python code snippets in Python 3

Version Range	# code snippets
3.0-3.3	147
3.0-3.6	149,506
3.1-3.3	1
3.1-3.6	1,350
3.3	54
3.3-3.5	169
3.3-3.6	10,901
3.5-3.6	4,307
3.6	19,569

the color of the 2.0-2.2 range in the figure is gray, indicating that there are no compilable code snippets in that Python version range.

Second, we used the available python version ranges for Python code snippets as shown in Table 8, Table 9 to combine the available version ranges of Python 2 and Python 3 and list the possible combinations. For each of the three categories of combinations as mentioned in Section 3, we investigate using the following methods:

(1) Ranges span Python 2 and Python 3： Taking [2.0,3.0-3.3] as an example, using the code snippets of 2.0 and 3.0-3.3 in Table 8 and Table 9 for comparison. The duplicate items of the two ranges are the code snippets contained in the range [2.0,3.0-3.3] that we need.

(2) Only in Python 2：Taking [2.0, ∅] as an example, we first use Table 6 to obtain code snippets that are uncompilable for all Python 3 versions. We then compare them with the code snippets in the 2.0 range in Table 8 to extract the duplicates. The duplicates are the code snippets we need.

(3) Only in Python 3：Taking [2.0, ∅] as an example, we first use Table 6 to obtain code snippets that are uncompilable for all Python 3 versions. We then compare them with the code snippets in the 2.0 range in Table 8 to extract the duplicates. The duplicates are the code snippets we need. For example, we first use Table 6 to obtain code snippets that are uncompilable for all Python 2 versions. We then compare them with the code snippets in the 2.0 range in Table 9 to extract the duplicates. The duplicates are the code snippets we need.

We obtained the results shown in Figure based on the above methods5. The first column of the figure shows the available Python version ranges in the Python 2 series in Table 8, and the first

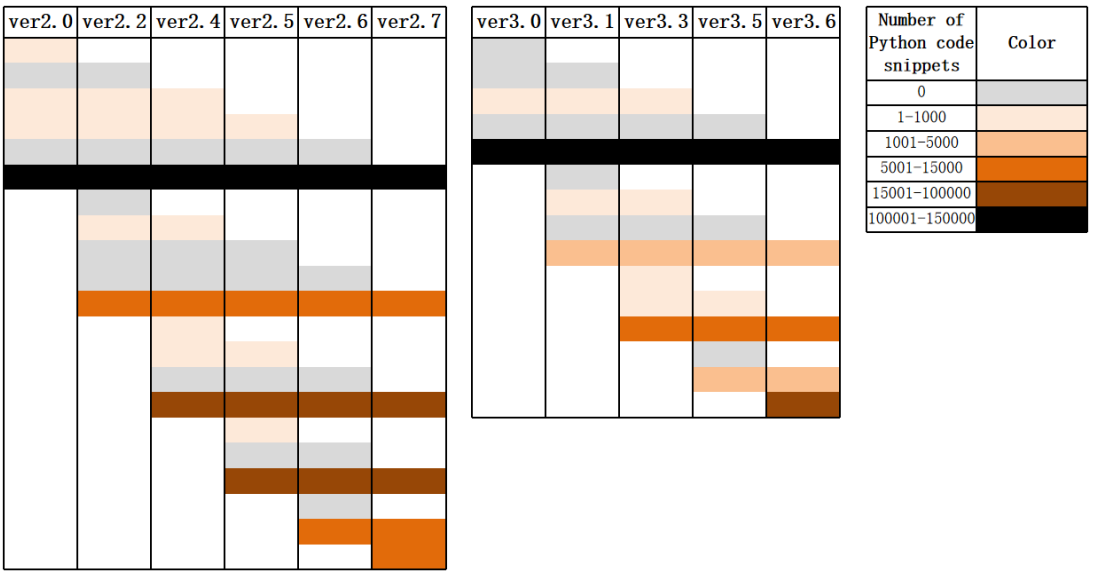


Figure 4: Available python version range for Python code snippets on Stack Overflow

row shows the available Python version ranges in the Python 3 series in Table 9. The values in the table represent the number of code snippets in each available Python version range. We divided the number of Python code snippets within each Python available range into five groups, marked with different colors. The correspondence between the number of code snippets and the colors is shown in the legend on the right side of the figure. In total, there are 46 non-zero available Python version ranges. The 13 ranges in the last column of the table belong to category 2, Only in Python 2. The total number of code snippets for this category is 121,784. The eight ranges in the last row belong to category 3, Only in Python 3. The total number of code snippets for this category is 56,682. The remaining 25 ranges are in category 1, ranges span Python 2 and Python 3. The total number of snippets for this category is 129,322.

In the figure, the data in the last column, which corresponds to the Python 3 series version range, is ∅, indicating that the code snippets in this column are uncompilable for all versions of Python 3. It is only compilable for Python 2 versions, which belong to category 2 mentioned above. The last row of data corresponding to the Python 2 series version range is ∅, and the code snippet is only compilable for Python 3 versions, which belongs to category 3. The rest of the data belongs to category 1, and the available Python version spans Python 2 and Python 3.

We first analyze the data of category 2 and category 3. Most of the code snippets in category 2 are in the range [2.0-2.7, ∅]. This means that code snippets in this range are compilable for all

Range	3.0-3.3	3.0-3.6	3.1-3.3	3.1-3.6	3.3	3.3-3.5	3.3-3.6	3.5-3.6	3.6	∅
2.0	0	1	0	0	0	0	0	0	0	8
2.0-2.4	0	0	0	0	0	0	0	0	0	139
2.0-2.5	0	0	0	0	0	0	0	0	0	22
2.0-2.7	103	60	0	99	0	155	9,340	0	0	104,559
2.2-2.4	0	0	0	0	0	0	0	0	0	1
2.2-2.7	0	8,450	0	0	0	0	42	0	0	996
2.4	0	0	0	0	0	0	0	0	0	11
2.4-2.5	0	0	0	0	0	0	0	0	0	7
2.4-2.7	23	34,059	0	4	0	3	335	0	0	4,383
2.5	0	0	0	0	0	0	0	0	0	9
2.5-2.7	12	58,313	0	7	0	10	332	0	0	8,712
2.6-2.7	2	8,713	0	0	0	0	98	0	0	2,230
2.7	0	7,915	1	1,198	0	0	47	0	0	707
∅	7	31,995	0	42	54	1	707	4307	19,569	-

Number of Python code snippets	Color
1-500	
501-5000	
5001-10000	
10001-100000	
100001-200000	

Figure 5: Available version ranges span Python 2 and Python 3

versions of Python 2. Similarly, most code snippets in category 3 are in the range [∅, 3.0-3.6]. In addition, in category 3, most of the remaining code snippets except for [∅, 3.0-3.6] are in the range containing version 3.6, such as [∅, 3.5-3.6], [∅, 3.6]. Although category 2 is not so obvious, we can also see from the figure that, the code snippets except for the range [2.0-2.7, ∅] are mostly concentrated in the range containing version 2.7, such as [2.5-2.7, ∅] and [2.6-2.7, ∅]. Looking next at the data for category 1, they also show a similar pattern to the previous category 2 and category 3, as shown in the figure. Most of the code snippets are concentrated in the range of versions containing 2.7 and 3.6, such as [2.7, 3.1-3.6], [2.4-2.72.7, 3.3-3.6].

Answer to RQ1: We found that the available version ranges of Python code snippets whose compilability varies with Python version changes in this study can be divided into three categories. The code snippets whose available version range spans Python 2 and Python 3 account for 42.0% (307,788 in total), those whose available version range is only in Python 2 account for 39.6%, and those whose available version range is only in Python 3 account for 18.4%. Code snippets are relatively concentrated in the range of versions containing 2.7 and 3.6.

4.2 RQ2:How Stack Overflow users respond to each Python release, and what are the differences between Python 2 and Python 3?

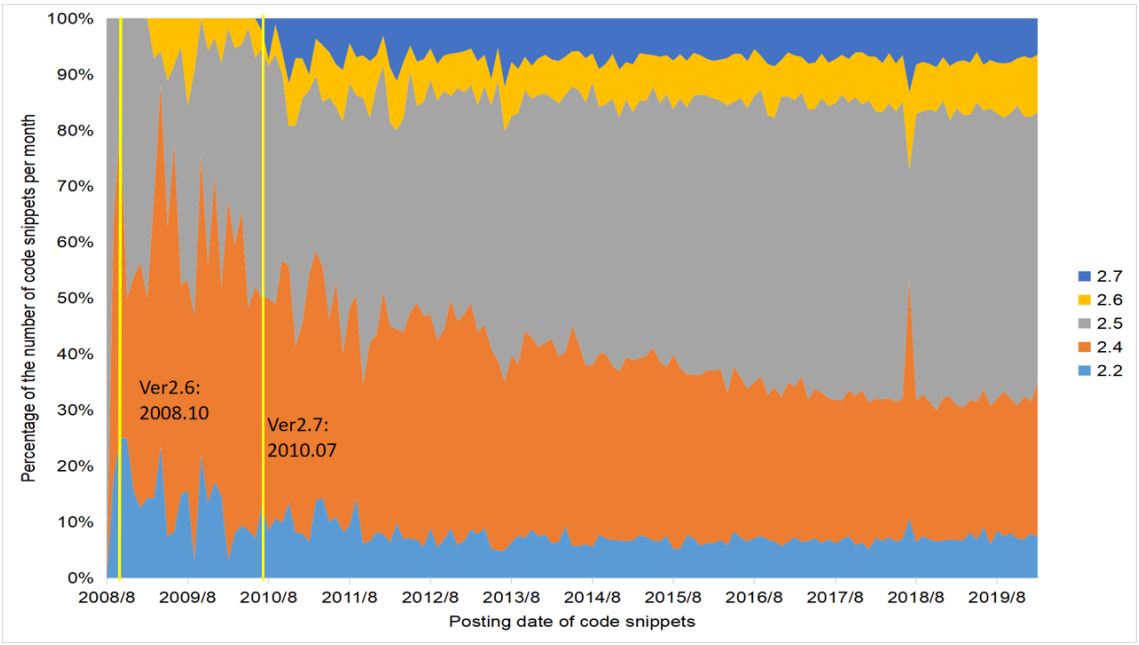
To investigate how Stack Overflow users respond to each Python release and the differences between Python 2 and Python 3, we studied the growth of Python code snippets responding to each Python version release based on the release date of each version in Table 7.

To improve the experimental accuracy in this research question, we divided the code snippets responding to each version by month. Figure 6 is the percent stacked area chart for Python 2 and Python 3. The horizontal axis represents the posting date of code snippets. The vertical axis shows the percentage of the number of code snippets per month. The release date of each Python version is marked with a yellow line in the figure.

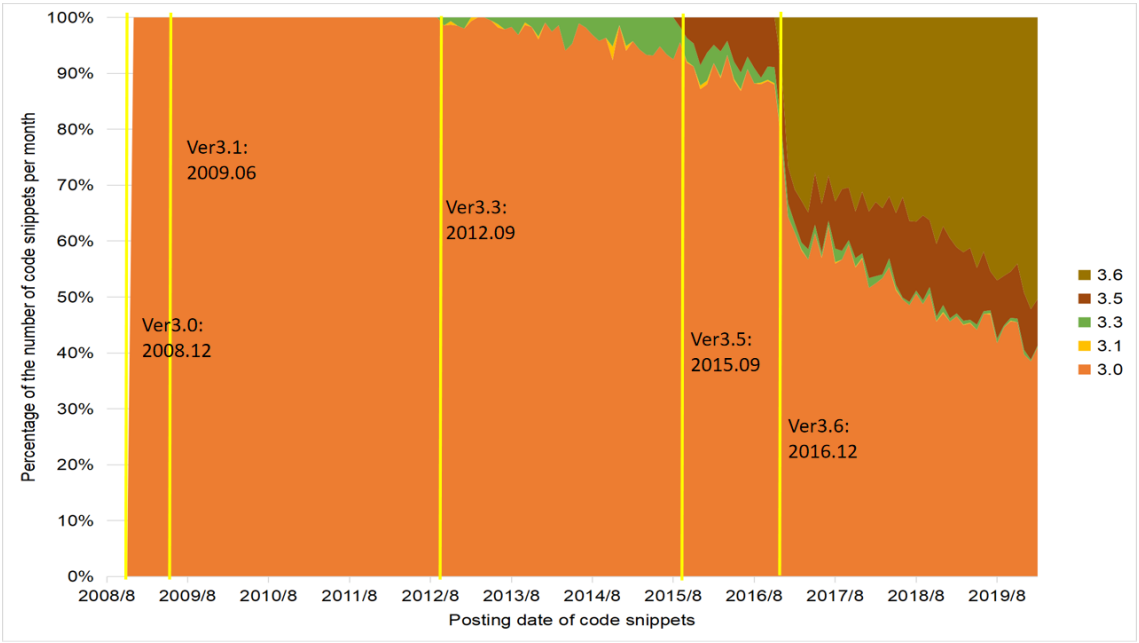
Figure 6a shows the growth of code snippets responding to each version of Python 2. Since August 2008, 2.2 and 2.4, which were released in October 2000 and December 2001 respectively, have shown an overall decreasing trend in their share of the total. 2.5 was released in September 2006 and took a more dominant position alongside 2.4 about two years after its release. 2.6 was released in October 2008, and code snippets responding to its release appeared about six months later. 2.6 also impacted 2.5 for a while after it gained a response, causing its share to continue to drop. 2.7 was released in July 2010 and quickly gained response after its release, and its response code snippets began to grow. 2.7 also impacted other Python 2 releases after its releases, such as 2.5 between August and December 2010, and saw a relatively significant drop.

Figure 6b shows the growth of code snippets responding to each version of Python 3 versions. 3.0 was released in December 2008, and 3.0 had a monopoly for quite some time as the Python 3 series was just getting started. 3.1 was released six months later, but it had almost no presence. 3.3 was released in September 2012. Although 3.3 was quickly responded to and gave 3.0 some impact, it was still not very strong. Immediately after 3.5 was released in September 2015, 3.0 and 3.3 dropped significantly. But up until here, 3.0 still held a great advantage. Until December 2016, the release of 3.6 brought a massive hit to versions such as 3.0, which saw a steep drop in share. 3.5 also took a hit but regained its vigor and grew modestly after a while. Together with 3.6, the newer Python 3 versions gradually took over the dominance of 3.0.

We found it impossible to compare the differences in user response to each version of Python 2 and Python 3 releases. However, some findings deserve our attention. After the response from users, 3.6 developed rapidly.While 2.6, 2.7 and 3.5 show rapid development after receiving responses for a short period of time, they do not grow rapidly afterwards. 3.1 and 3.3 have almost no obvious development after receiving the response, and the proportion is close to 0.



(a) Growth of Python code snippets responding to Python 2 versions



(b) Growth of Python code snippets responding to Python 3 versions

Figure 6: Growth of Python code snippets responding to each Python version

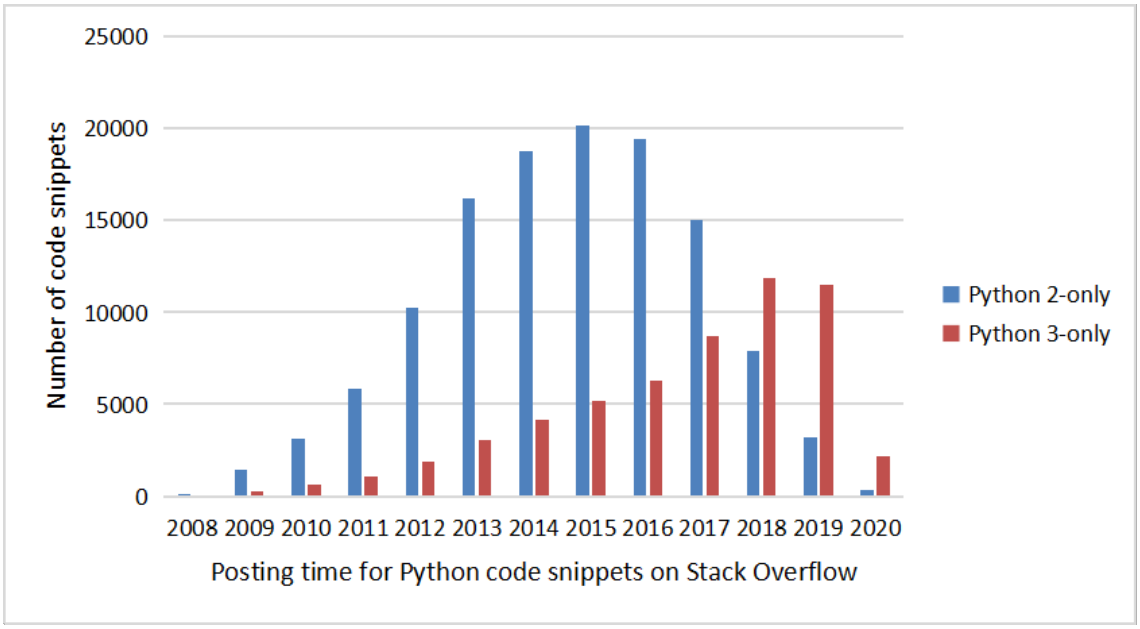


Figure 7: Distribution of Python 2-only compilable Python code snippets and Python 3-only compilable Python code snippets

Answer to RQ2: We found that new versions are released and get a response shortly afterward. The release of new Python version inhibits Stack Overflow users ' response to older versions. It is impossible to compare the differences in user response to each version of Python 2 and Python 3 releases.

4.3 RQ3:How are Python 2-only compilable Python code snippets and Python 3-only compilable Python code snippets distributed on Stack Overflow?

To investigate RQ3, we utilized the code snippets compilable only for Python 2 (121,784) and the code snippets compilable only for Python 3 (56,682) obtained in RQ1.

Figure 7 shows the distribution of Python 2-only compilable Python code snippets and Python 3-only compilable Python code snippets. Since SOTorrent only includes data for the first two months or so of 2020, the overall number of Python code snippets in 2020 is less.

As shown in Figure 7, The number of Python 2-only compilable Python code snippets increased year by year starting in 2008, peaked in 2015, and has since begun to decline each year. There is an overall upward trend in Python 3-only compilable Python code snippets.

While the number of Python 3-only compilable Python code snippets is increasing, overall, the number of Python 2-only compilable Python code snippets far exceeds the number for Python 3.

Answer to RQ3: The number of Python 2-only compilable Python code snippets increased year by year starting in 2008, peaked in 2015, and began to decline year by year after that. There is an overall upward trend in Python 3-only compilable Python code snippets. But overall, among the code snippets investigated by RQ3, the Python 2-only compilable Python code snippets are about twice as large as the Python 3-only compilable Python code snippets.

5 Discussion

5.1 Findings

The purpose of this work is to understand the impact of Python version upgrades on the compilability of Python code snippets on Stack Overflow. We investigated the effects of Python 2 and Python 3 internal version evolution on the compilability of Python code snippets on Stack Overflow separately. We also compared and analyzed the similarities and differences between the two series. We found that:

(1) The available version ranges of Python code snippets whose compilability varies with Python version changes in this study can be divided into three categories. The code snippets whose available version range spans Python 2 and Python 3 account for 42.0% (307,788 in total), those whose available version range is only in Python 2 account for 39.6%, and those whose available version range is only in Python 3 account for 18.4%. Code snippets are relatively concentrated in the range of versions containing 2.7 and 3.6.

(2) We found that new versions are released and get a response shortly afterward. The release of new Python versions inhibits Stack Overflow users' response to older versions. It is impossible to compare the differences in user response to each version of Python 2 and Python 3 releases.

(3) The number of Python 2-only compilable Python code snippets increased year by year starting in 2008, peaked in 2015 and began to decline year by year after that. There is an overall upward trend in Python 3-only compilable Python code snippets. But overall, among the code snippets investigated by RQ3, the Python 2-only compilable Python code snippets are about twice as large as the Python 3-only compilable Python code snippets.

Our study only investigated the compilability of code snippets on Stack Overflow and did not delve into the syntactic features of the code snippets that affect their compilability changes. Future research should focus on investigating the factors that influence these changes.

5.2 Implications

Through investigation, we found that the Python version upgrade impacted the compilability of quite a few Python code snippets on Stack Overflow. Stack Overflow, as well as Stack Overflow users, should pay attention to this issue.

5.2.1 Suggestions for Stack Overflow

An automated tool can be built to identify the Python language version of existing code snippets in posts on Stack Overflow or help users identify the Python version used in the posted code snippet in real-time as they create their questions or answers. In RQ3, we found that code snippets only compilable for Python 2 accounted for about 40% of the total number of code snippets we surveyed (307,788 total). Although the Python team discontinued support for Python 2 on January 1, 2020, there are still newly posted code snippets that are only compilable for Python 2 until 2020. An automated tool can be developed to identify possible Python versions of code snippets by analyzing their syntactic features as they are entered. An example is the tool Vermin[4], produced by Morten Kristensen. It functions by parsing Python code into an abstract syntax tree (AST) concurrently detects the minimum Python versions needed to run code.

5.2.2 Suggestions for Stack Overflow Users

We recommend that Stack Overflow users provide information about the Python version used to write the code snippet when attaching code snippets to illustrate their questions or give answers. In RQ1, we observed that many of Stack Overflow's Python code snippets have different available Python versions ranges and are not compiled by all Python versions. If users actively provide the Python version of the code snippet when posting or answering questions.

Users searching Stack Overflow for the required code snippet should also pay more attention to the information about the Python version in the posts. Alternatively, before utilizing the code snippet, identify the possible Python version of the snippet.

5.3 Threats to Validity

Our study is subject to limitations and threats to validity.

Limitations of PyComply: Parsing metrics of PyComply [6] parsing metrics are based on (static) syntactic observations, coarse-grained, and roughly categorize each Python file into compliant or not, without attempting to estimate the degree of compliance. While we believe that the level studied here is sufficient for our purposes, we should note that the parsing results of these PyComply cannot assert that any code snippet is 100% compliant with the Python version.

Limitations of Python code snippets: As we mentioned in Section 3, about 47.4% (1,173,905) of the Python code snippets we obtained from SOTorrent could not pass the PyComply parsing for all Python versions, i.e., these code snippets are not compilable for all Python versions. These

[4] `https://github.com/netromdk/vermin`

code snippets contain programming errors, pseudocode, and other issues unrelated to the Python version upgrade that caused the parsing failure. However, there may also be featured in these code snippets related to Python version upgrades but are masked by errors caused by other unrelated issues. Because of technical and time constraints, we abandoned processing and studying this part of the code snippets in this study. This may pose a threat to the validity of our results.

External validity: The generality of our findings poses a danger to external validity. We focused on Stack Overflow in this study, and our findings may not apply to other Q&A sites because of the differences in mechanisms. To mitigate this threat, we should study more Q&A sites in the future.

Furthermore, unlike Malloy et al.'s analysis using Qualitas corpus [6], the Python files contained in it are parsed on a per-application basis. Our research objects are unrelated code snippets, and many of them are only short fragments, lacking syntactic features that can be used as metrics. This makes our findings less generalizable and perhaps not applicable beyond the Q&A site.

6 Conclusion

In this study, we want to investigate the effect of Python version upgrades on the compilability of Python code snippets on Stack Overflow. By analyzing the compilability of Python code snippets on Stack Overflow for different Python versions, We found that Python version upgrades had an impact on the compilability of Python code snippets on Stack Overflow, as evidenced by: 1)about 40% of the Python code snippets on Stack Overflow whose compilability changed with the Python version in this study are uncompilable for Python 3. 2) The release of new Python version inhibits Stack Overflow users ' response to older versions. 3) The trend of code snippets responding to newer versions increases over time. Based on our findings, we offer the following suggestions: 1) An automated tool can be built to help users identify the Python language version of code snippets in Stack Overflow posts. 2) We recommend that Stack Overflow users provide information about the Python version used to write the code snippet when attaching code snippets to illustrate their questions or give answers. 3）Users searching Stack Overflow for the required code snippet should also pay more attention to the information about the Python version in the posts.

There are two possible directions for future work. First, we want to investigate factors that affect the compilability of code snippets on Stack Overflow due to Python version upgrades. For example, the syntax usage of specific Python versions. In addition, we would like to investigate the reasons for the failure of the code snippets filtered out in this study that are not compilable for all Python versions.

Acknowledgement

Over the course of my researching and writing this paper, I would like to express my thanks to all those who have helped me.

First of all, I would like to thank my supervisor, Professor Katsuro Inoue, who gives me the opportunity to study and work in Software Engineering Laboratory, and provided important suggestions and guidance to my research.

I would like express my gratitude to Assistant Professor Tetsuya Kanda. His guidance helped me in all the time of research and writing of my thesis. He is a very learned and responsible teacher. After every day ' s tiring and busy work of his own, he still devoted his considerate care and immense vigor to the supervision of my writing thesis, including his suggestions on wording, his help in forming the structure, and the efforts to the refinement of my ideas in my thesis.

I want to thank Associate Professor Makoto Matsushita. He gave me a lot of valuable comments and suggestions on my research.

I am grateful to Kazumasa Shimari, Shi Qiu, for their valuable comments and suggestions in my research. Without their help, it would not have been possible for me to complete my thesis.

I also would like to thank Mrs. Mizuho Karube, who helped me a lot with my life in Japan. Without her help, I would not have been able to concentrate on study and research.

I would like to thank all the members in Inoue laboratory for creating such an excellent research environment with their passion and creativity.

Finally, I would like to give my heartfelt thanks to my parents, for their endless love and care for me. Whatever I need and wherever I go, they are always there supporting me without any requirement in return. I love them.

References

[1] Durham Abric, Oliver E. Clark, Matthew Caminiti, Keheliya Gallaba, and Shane McIntosh. Can duplicate questions on stack overflow benefit the software development community? In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, pages 230–234, 2019.

[2] L. An, O. Mlouki, F. Khomh, and G. Antoniol. Stack overflow: A code laundering platform? In *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 283–293, Los Alamitos, CA, USA, feb 2017. IEEE Computer Society.

[3] Sebastian Baltes, Lorik Dumani, Christoph Treude, and Stephan Diehl. Sotorrent: Reconstructing and analyzing the evolution of stack overflow posts. MSR '18, New York, NY, USA, 2018. Association for Computing Machinery.

[4] Sebastian Baltes, Christoph Treude, and Stephan Diehl. Sotorrent: Studying the origin, evolution, and usage of stack overflow code snippets. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, pages 191–194, 2019.

[5] Felix Fischer, Konstantin Böttinger, Huang Xiao, Christian Stransky, Yasemin Acar, Michael Backes, and Sascha Fahl. Stack overflow considered harmful? the impact of copy amp;paste on android application security. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 121–136, 2017.

[6] Brian A. Malloy and James F. Power. Quantifying the transition from python 2 to 3: An empirical study of python applications. In *Proceedings of the 11th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '17, page 314–323. IEEE Press, 2017.

[7] Saraj Singh Manes and Olga Baysal. How often and what stackoverflow posts do developers reference in their github projects? In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, pages 235–239, 2019.

[8] Saraj Singh Manes and Olga Baysal. Studying the change histories of stack overflow and github snippets. In *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*, pages 283–294, 2021.

[9] Kotaro Nishimura. Investigating the validity of code snippets containing the java api posted on stack overflow. https://sel.ist.osaka-u.ac.jp/lab-db/Bthesis/contents.en/169.html.

[10] Chaiyong Ragkhitwetsagul, Jens Krinke, Matheus Paixao, Giuseppe Bianco, and Rocco Oliveto. Toxic code snippets on stack overflow. *IEEE Transactions on Software Engineering*, 47(3):560–581, 2021.

[11] Akond Rahman, Effat Farhana, and Nasif Imtiaz. Snakes in paradise?: Insecure python-related coding practices in stack overflow. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, pages 200–204, 2019.

[12] Abhishek Soni and Sarah Nadi. Analyzing comment-induced updates on stack overflow. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, pages 220–224, 2019.

[13] M. Verdi, A. Sami, J. Akhondali, F. Khomh, G. Uddin, and A. Karami Motlagh. An empirical study of c++ vulnerabilities in crowd-sourced code examples. *IEEE Transactions on Software Engineering*, (01):1–1, sep 5555.

[14] Yuhao Wu, Shaowei Wang, Cor-Paul Bezemer, and Katsuro Inoue. How do developers utilize source code from stack overflow? *Empirical Software Engineering*, 24, 04 2019.

[15] Haoxiang Zhang, Shaowei Wang, Tse-Hsun Chen, Ying Zou, and Ahmed E. Hassan. An empirical study of obsolete answers on stack overflow. *IEEE Transactions on Software Engineering*, 47(4):850–862, 2021.