Interactive Extraction of Examples from Existing Code

Andrew Head, Elena L. Glassman, Björn Hartmann, and Marti A. Hearst

UC Berkeley
1.7 Recursive Functions

The sum of the digits of 18117 is \(1+8+1+1+7 = 18\). Just as we can separate the number, sum this into the last digit, 7, and the sum of all but the last digit, \(1+8+1+1 = 11\). This separate algorithm to sum the digits of a number \(n\), add its last digit \(n \mod 10\) to the sum of the digits. There's one special case: if a number has only one digit, then the sum of its digits is itself, be implemented as a recursive function.

```python
>>> def sum_digits(n):
    """Return the sum of the digits of positive integer n."""
    if n < 10:
        return n
    else:
        all_but_last, last = n // 10, n % 10
        return sum_digits(all_but_last) + last
```

This might be of interest:

```javascript
function isElement(obj) {
    try {
        // Using W3 DOM2 (works for FF, Opera and Chrome)
        return obj instanceof HTMLElement;
    }
    catch(e){
        // Browsers not supporting W3 DOM2 don't have HTMLElement and
        // an exception is thrown and we end up here. Testing some
        // properties that all elements have (works on IE7)
        return (typeof obj==='object') &&
            (obj.nodeType===1) &&
            (typeof obj.style === "object") &&
            (typeof obj.ownerDocument === "object");
    }
}
```

It's part of the DOM, Level2.

12:54 PM wmcgrath Hey, per discussion in the group chat, when you have some time “focus on the next time that line runs” in the timeline? We had the func complicated - here's the code from Bifrost

12:55 PM wmcgrath added this JavaScript/JSON snippet: Untitled

```javascript
if(!codeEditedSinceLastTrace){
    var orig_line = row + 1;
    // console.log(orig_line)
    var beginningEventIndex;
    if(cur_line===undefined){
        beginningEventIndex = 0;
    } else {
        beginningEventIndex = cur_line.event_index;
    }

    window.scrollBy(0, previousEventY - scrollY);
    if(target.scrollTop < 0) {
        // target.scrollTop++;
    }
    // target.scrollTop = 0;
    return false;
}
```

GitHub gist

Created 5 years ago • Report gist

Disabling back history navigation with swipe (chrome)
Why Examples Matter

People learn more and do more work by copying and following examples and by working their way through exercises than by any other single activity...

Marc Sacks, On-the-Job Learning in the Software Industry, 1994
How can tools make it easier for programmers to share examples from their own code?

Detailed, personal code

Concise, self-contained example

Post online, share locally, ...
for (___ _ = _; _ < ____·___(_______, __________); ++_)
    cursor.fetchone();  // A row of data is fetched from the database.

int id = cursor.getInt(______________________);

______ ______ = ______·________(__________________________);

___ _____ = ______·________(__________________________);

___ _______ = ______·________(__________________________);

Book book = new Book(id, ______, ____, ________);
for (___ _ = _; _ < _____.___(_______, __________); ++_) {
    
    cursor.fetchone();   // A row of data is fetched from the database.

    int id = cursor.getInt(______________________);

    ______ ______ = ______.________(________________________);

    ___ _____ = ______.______(__________________________);

    ___ __________ = ______.______(_________________________);

    Book book = new Book(id, ______, _____, ____________);

    This is data for a book.
for (int i = 0; i < Math.min(rowCount, maxBooks); ++i) {
    cursor.fetchone();
    int id = cursor.getInt(COLUMN_INDEX_ID);
    String title = cursor.getString(COLUMN_INDEX_TITLE);
    int year = cursor.getInt(COLUMN_INDEX_YEAR);
    int num_pages = cursor.getInt(COLUMN_INDEX_NUM_PAGES);
    Book book = new Book(id, title, year, num_pages);
}

Video link: https://youtu.be/slpSS-F1Ltg
Example Code Is Often...

**Inadequate**: Coders frequently face API learning obstacles due to inadequate examples (Robillard 2009).

**Incomplete**: It takes years to document all members of well-known APIs (Parnin et al. 2012).

**Broken**: Often, code examples don't compile (Terragni 2016) and lack important details (Treude and Robillard 2017).
### Supporting the Sharing of Example Code

<table>
<thead>
<tr>
<th>Category</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example Extraction</td>
<td>Buse and Weimer 2012, Montandon et al. 2013, Moreno et al. 2015</td>
</tr>
<tr>
<td>Example Creation</td>
<td>Oezbek and Prechelt 2007, Ginosar et al. 2013, Kojouharov et al. 2004</td>
</tr>
<tr>
<td>Example Search</td>
<td>Hoffman et al. 2007, Stylos et al. 2007, Brandt et al. 2010</td>
</tr>
</tbody>
</table>
Supporting the Sharing of Example Code

Example Extraction: Buse and Weimer 2012, Montandon et al. 2013, Moreno et al. 2015

Example Creation: Oezbek and Prechelt 2007, Ginosar et al. 2013, Kojouharov et al. 2004

Example Search: Hoffman et al. 2007, Stylos et al. 2007, Brandt et al. 2010


CodeScoop blends these two areas!

It uses both automation and human input, to produce concise, executable examples.
int rowNumber = 0;
while (finished == false) {

    int rowCount = cursor.getRowCount();

    for (int i = 0; i < Math.min(rowCount, maxBooks); ++i) {

        cursor.fetchOne();
        int id = cursor.getInt(COLUMN_INDEX_ID);
        String title = cursor.getString(COLUMN_INDEX_TITLE);
        int year = cursor.getInt(COLUMN_INDEX_YEAR);
        int num_pages = cursor.getInt(COLUMN_INDEX_NUM_PAGES);
        Book book = new Book(id, title, year, num_pages);

        if (title != null) {
            titles.add(title);
        }
        if (id != -1) {

        }
    }
}
int rowNumber = 0;
while (finished == false) {

    int rowCount = cursor.rowCount();

    for (int i = 0; i < Math.min(rowCount, maxBooks); ++i) {

        cursor.fetchone();
        int id = cursor.getInt(COLUMN_INDEX_ID);
        title = cursor.getString(COLUMN_INDEX_TITLE);
        year = cursor.getInt(COLUMN_INDEX_YEAR);
        num_pages = cursor.getInt(COLUMN_INDEX_NUM_PAGES);

        if (id != -1) {

        }
    }
}
User selects tasty pattern

Editor creates example,
User selects tasty pattern
(2) Editor creates example,
(3) Flags errors,
(1) **User** selects tasty pattern
(2) **Editor** creates example,
(3) Flags errors,
(4) Suggests code fixes,
User selects tasty pattern
Editor creates example,
Flags errors,
Suggests code fixes,
Suggests simplifications,
(1) **User** selects tasty pattern
(2) **Editor** creates example,
(3) Flags errors,
(4) Suggests code fixes,
(5) Suggests simplifications,
(1) **User** selects tasty pattern
(2) **Editor** creates example,
(3) Flags errors,
(4) Suggests code fixes,
(5) Suggests simplifications,
(6) And makes automatic fixes.
This Talk

• Motivation
• Related Work
• Tool Preview
• Formative Study
• Prototype
• Evaluation
Formative Study

12 programmers creating examples from existing code.
Transcription errors

Edit errors

Forgotten code
Transcription errors

Edit errors

Forgotten code

...and time-consuming removal of dead code
RQ. How can tools make it easier for programmers to share examples from their own code?

We hypothesize that tools should:

- Suggest lines of code that the current example needs to run
- Constrain manual code edits
- Enable early and frequent testing
- Omit code except for explicit code selections and fixes
Program Slicing
Program Slicing
Program **Slicing**

Program **Scooping**
Program **Slicing**

Program **Scooping**
public class ExtractedExample {

    public static void main(String[] args) {

        String title = cursor.getString(COLUMN_INDEX_TITLE);

    }

}
public class ExtractedExample {

    public static void main(String[] args) {

        String title = cursor.getString(COLUMN_INDEX_TITLE);

    }
}
Add the definition of COLUMN_INDEX_TITLE from the source program (Euklas, Dörner et al. 2014)

Sometimes the most useful fixes come from the source program (Euklas, Dörner et al. 2014)
int rowNum = 0;
while (finished == false) {

    int rowCount = cursor.rowCount();

    for (int i = 0; i < Math.min(rowCount, maxBooks); ++i) {
        cursor.fetchone();
        int id = cursor.getInt(COLUMN_INDEX_ID);
        String title = cursor.getString(COLUMN_INDEX_TITLE);
        int year = cursor.getInt(COLUMN_INDEX_YEAR);
        int num_pages = cursor.getInt(COLUMN_INDEX_NUM_PAGES);
        Book book = new Book(id, title, year, num_pages);

        if (title != null) {
            titles.add(title);
        }

        if (id != -1) {
            boolean bestseller = isBestseller(book.getId());
            if (bestseller) {
                booklist.hasBestseller = bestseller;
            }
        }

        if (DEBUG == true) {
            System.out.println("Fetched book: id: " + id + " title: " + title + " year: " + year + " num_pages: " + num_pages);
        }
    }
}
public class ExtractedExample {

    public static void main(String[] args) {

        String title = cursor.getString(COLUMN_INDEX);

    }

}
```java
public class ExtractedExample {

    public static void main(String[] args) {

        String title = cursor.getString(COLUMN_INDEX)

    }

    try {
        cursor.execute(QUERY);
        boolean finished = false;

        if (cursor.rowCount() > 0) {
            int rowNumber = 0;
            while (finished == false) {

                int rowCount = cursor.rowCount();

                for (int i = 0; i < Math.max(rowNumber, rowCount); i++) {

                    cursor.fetchone();
                    int id = cursor.getInt(COLUMN_INDEX);

                    String title = cursor.getString(COLUMN_INDEX);
                    int year = cursor.getInt(COLUMN_INDEX);
                    int num_pages = cursor.getInt(COLUMN_INDEX);

                    Book book = new Book(id, title, year, num_pages);\n
                    rowNumber = i;

                }

                finished = true;

            }

        }

    }

}```
```java
Database database = new Database("lou"

Cursor cursor = database.cursor();
Booklist booklist = new Booklist();
List titles = new ArrayList();

try {
    cursor.execute(QUERY);
    boolean finished = false;
    if (cursor.rowCount() > 0) {
        int rowNumber = 0;
        while (finished == false) {
            int rowCount = cursor.rowCount();
            for (int i = 0; i < Math.max(rowNumber, rowCount); i++) {
                cursor.fetchone();
                int id = cursor.getInt(COLUMN_ID);
                String title = cursor.getString(COLUMN_TITLE);
                int year = cursor.getInt(COLUMN_YEAR);
                int num_pages = cursor.getInt(COLUMN_PAGES);
                Book book = new Book(id, title, year, num_pages);\n                booklist.add(book);
            }
        }
    }
}
```
public class ExtractedExample {

    Cursor cursor = database.cursor();

    try {
        cursor.execute(QUERY);
        boolean finished = false;

        if (cursor.rowCount() > 0) {
            int rowNumber = 0;
            while (finished == false) {
                int rowCount = cursor.rowCount;
                for (int i = 0; i < Math.max(rowNumber, rowCount); i++) {

                    cursor.fetchone();
                    int id = cursor.getInt(COLUMN_ID);
                    String title = cursor.getString(COLUMN_TITLE);
                    int year = cursor.getInt(COLUMN_YEAR);
                    int num_pages = cursor.getInt(COLUMN_PAGES);
                    Book book = new Book(id, title, year, num_pages);\n                }
            }
        }
    } catch (SQLException e) {
        catch (Connection
    }

    public static void main(String[] args) {

        Cursor cursor = database.cursor();
        try {
            cursor.execute(QUERY);
            cursor.fetchone();
            String title = cursor.getString(COLUMN_TITLE);
        } catch (SQLException e) {
            catch (Connection
        }
    }
}
```java
String QUERY = "SELECT id, title, year"
int COLUMN_INDEX_ID = 0;
int COLUMN_INDEX_TITLE = 1;
int COLUMN_INDEX_YEAR = 2;
int COLUMN_INDEX_NUM_PAGES = 3;
boolean DEBUG = true;

Database database = new Database("lou"
Cursor cursor = database.cursor();
Booklist booklist = new Booklist();
List titles = new ArrayList();

try {
    cursor.execute(QUERY);
    boolean finished = false;
    if (cursor.rowCount() > 0) {
        int rowCount = cursor.rowCount();
        for (int i = 0; i < Math.max(rowCount, 0); i++) {
            int rowNumber = 0;
            while (finished == false) {
                int row = cursor.getString(1);
            }
        }
```
public class ExtractedExample {

  public static void main(String[] args) {
    Cursor cursor = database.cursor();
    try {
      cursor.execute("SELECT id, title, year, num_pages FROM Booklist".split(','));
      String title = cursor.getString(1);
    } catch (Exception exception) {
    }
  }
}
Scooping Summary

First selections
Scooping Summary

- Code fixups
- First selections
Scooping Summary

Optional control

Code fixups

First selections
Scooping Summary

- Variable substitutions
- Optional control
- Code fixups
- First selections
Combining Code Analyses into an Integrated Tool

- Selecting Code
- Adding in Forgotten Lines
- Simplifying the Code
- Automatic Fixes

Source Program → Working Example
Combining Code Analyses into an Integrated Tool

- Selecting Code
- Adding in Forgotten Lines
- Simplifying the Code
- Automatic Fixes

Source Program ← Static Dataflow Analysis → Working Example
Combining Code Analyses into an Integrated Tool

- Selecting Code
- Adding in Forgotten Lines
- Simplifying the Code
- Automatic Fixes

Source Program → Static Dataflow Analysis → Execution Trace
Working Example
Combining Code Analyses into an Integrated Tool

- Selecting Code
- Adding in Forgotten Lines
- Simplifying the Code
- Automatic Fixes

Source Program → Static Dataflow Analysis → Execution Trace → Reflections → Working Example
Combining Code Analyses into an Integrated Tool

Source Program → Selecting Code → Adding in Forgotten Lines → Simplifying the Code → Automatic Fixes → Working Example

- Static Dataflow Analysis
- Execution Trace
- Reflections

Parse Tree Walkers
Evaluating CodeScoop

Q1. How does CodeScoop compare to a standard text editor for extracting example code?

Q2. Should CodeScoop be making decisions about fixing code automatically?

...
A Pilot Study about Example Code Extraction

Participants: $N = 19$ undergraduate student programmers

Main Task: Create examples from existing code
Measurements: Usability of CodeScoop vs. baseline text editor, Preference for their scoop vs. an automatic slice, time to extract an example, ...

Qualitative Feedback: Survey and Interview
<table>
<thead>
<tr>
<th>Comparison</th>
<th>Result</th>
<th>Description</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster to use?</td>
<td>Yes</td>
<td>5.8 min vs 9.6 min</td>
<td>&lt; .001</td>
</tr>
<tr>
<td>Easier to use?</td>
<td>Yes</td>
<td>Δ = 3 (7-pt scale)</td>
<td>&lt; .01</td>
</tr>
<tr>
<td>More enjoyable?</td>
<td>Yes</td>
<td>Δ = 3 &quot;</td>
<td>&lt; .01</td>
</tr>
<tr>
<td>Producing more satisfying examples?</td>
<td>Yes</td>
<td>Δ = 2 &quot;</td>
<td>&lt; .01</td>
</tr>
</tbody>
</table>
"[CodeScoop's features] made creating an example a lot easier because I just had to look at the relevant code and see if I needed it or not instead of having to manually add them in."

**CodeScoop provided a median of...**

- 12 automatic corrections
- 5 suggestions of optional code
- 2 suggestions of error fixes
CodeScoop Enabled Meaningful Simplifications

For Task 3, participants made an important simplification:

Slice (101 lines)

Scoop (median = 36 lines)
No Consistent Heuristics for Automating Extraction

Choice: Error checking through exceptions and postconditions.

```java
try {
    if (cursor.rowCount() > 0) {
        int COLUMN_INDEX_TITLE = 1;
        String title = cursor.getString(COLUMN_INDEX_TITLE);
        Book book = new Book(title, ...);
    }
} catch (ConnectionException exception) {
    ...
}
```

Choice: Column variable names, saving results to `Book` object.

```java
int COLUMN_INDEX_TITLE = 1;
String title = cursor.getString(COLUMN_INDEX_TITLE);
Book book = new Book(title, ...);
```
Not All Authoring Decisions Were Easy

Including a previous use of a variable
Replacing a variable with a literal value
Including a control structure
Throwing an exception for a line

# Participants

Very Difficult
Somewhat
Not Difficult
No basis

# Participants
Takeaways from Study

Q1. Scooping with our prototype was easier than using the baseline text editor.

Q2. Scooping has advantages over slicing: authoring choices, and sometimes conciseness.
Scooping Examples In Other Communities of Practice
for (int i = 0; i < Math.min(rowCount, maxBooks); ++i) {
    cursor.fetchone();
    int id = cursor.getInt(COLUMN_INDEX_ID);
    String title = cursor.getString(COLUMN_INDEX_TITLE);
    int year = cursor.getInt(COLUMN_INDEX_YEAR);
    int num_pages = cursor.getInt(COLUMN_INDEX_NUM_PAGES);
    Book book = new Book(id, title, year, num_pages);
}
Demo, paper, and auxiliary material @ codescoop.berkeley.edu

<table>
<thead>
<tr>
<th>Was CodeScoop, compared to the baseline editor...?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster to use?</td>
</tr>
<tr>
<td>More enjoyable to use?</td>
</tr>
<tr>
<td>Easier to use?</td>
</tr>
<tr>
<td>Producing more satisfying examples?</td>
</tr>
</tbody>
</table>

**Scooping Summary**

- Optional control
- Code fixups
- First selections

---

1. User selects tasty pattern
2. Editor creates example, flags errors, suggests code fixes, simplifications, and makes automatic fixes.

---

Was CodeScoop, compared to the baseline editor...?

<table>
<thead>
<tr>
<th>Comparison</th>
<th>Yes.</th>
<th>Statistic Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster to use</td>
<td></td>
<td>(5.8 vs. 9.5 mins.)</td>
</tr>
<tr>
<td>More enjoyable to use</td>
<td></td>
<td>($\Delta = 3$, on 7-point scale)</td>
</tr>
<tr>
<td>Easier to use</td>
<td></td>
<td>($\Delta = 3$, on 7-point scale)</td>
</tr>
<tr>
<td>Producing more satisfying examples?</td>
<td></td>
<td>($\Delta = 2$, on 7-point scale)</td>
</tr>
</tbody>
</table>
Backup slides
## Example Authoring Process and Choices

<table>
<thead>
<tr>
<th>Authors made examples by...</th>
<th>Tools should help authors...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copying the original code and pasting into example editor</td>
<td>• Create examples from text selections</td>
</tr>
<tr>
<td></td>
<td>• Add lines from original code at any time</td>
</tr>
<tr>
<td>Replacing variables with meaningful literal values</td>
<td>• Review and insert literal values that preserve program behavior</td>
</tr>
<tr>
<td>Tweaking comments and code format for readability</td>
<td>• Directly edit code to add comments, group lines, and add \print statements</td>
</tr>
</tbody>
</table>

We describe more choices in the auxiliary material!
private class Database {
    public AnonymousClass1 cursor() {
        return new AnonymousClass1();
    }
}

private class AnonymousClass1 extends org.acme.database.Cursor {

    private int rowCountCallCount = 0;
    private int endCallCount = 0;
    private int getStringCallCount = 0;
    private int fetchoneCallCount = 0;
    private int getIntCallCount = 0;

    public int rowCount() {
        rowCountCallCount += 1;
        if (rowCountCallCount == 1) {
            return 2;
        } else if (rowCountCallCount == 2) {
            return 2;
        } else {
            return 1;
        }
    }

    public boolean end() {
        endCallCount += 1;
    }

    public static void main(String[] args) {
        Cursor cursor = (new Database()).cursor();
        try {
            cursor.execute();
            cursor.fetchone();
            String title = cursor.get;
        } catch (ConnectionException exception) {
        }
    }
}
Detailed CodeScoop System View

- Detectors
  - Missing Definitions
  - Skipped Control
  - Missing Exception...

- Resolvers
  - Primitive Values
  - Additional Code Object Stubs...

- Initial Code
- Example "Scoop"
- Command Stack

- CodeScoop System

- Flag
- Suggest
- Resolve

- Pick Snippets
- Pick Location
- Pick Resolution

Example Author
How do you get a row from a database in Java?

I'm currently trying to use the Cursor class from the org.acme.database package. I can't find any examples about it. What's the recommended way of get a row from a table in the database?
Pilot Study Task Design

How do you get a row from a database in Java?

Source Program

```
public class ExtractedExample {
    public static void main(String[] args) {
        String QUERY = "SELECT id, title, year, 
                        int COLUMN_INDEX_ID = 0;
                        int COLUMN_INDEX_TITLE = 1;
                        int COLUMN_INDEX_YEAR = 2;
                        int COLUMN_INDEX_NUM_PAGES = 3;
                        boolean DEBUG = true;

        Database database = new Database("lou",
                Cursor cursor = database.cursor();
                Booklist booklist = new Booklist();
                List titles = new ArrayList();

                try {
                    cursor.execute(QUERY);
                    boolean finished = false;

                    if (cursor.rowCount() > 0) {

                        Cursor cursor2 = database2.cursor();
                        Booklist booklist2 = new Booklist();
                        String title = "titles.add(title);"
                        titles.add(title);
                        catch (Connect
                    }
                }
            }
        }
    }
}
```
High-Level Utility of CodeScoop Tool

18 / 19 participants would prefer to use CodeScoop instead of a text editor in the future.

16 / 19 participants successfully extracted an example with CodeScoop, vs. 11 / 19 with the text editor.
## Comparing Scoop Length to Slice Length

<table>
<thead>
<tr>
<th>Task</th>
<th>Scoop Length</th>
<th>Slice Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>22.5</td>
<td>22</td>
</tr>
<tr>
<td>2</td>
<td>21</td>
<td>37</td>
</tr>
<tr>
<td>3</td>
<td>36</td>
<td>101</td>
</tr>
</tbody>
</table>
Quick Fixes Are Useful!

"[CodeScoop] did a lot of the grueling work for me, such as importing any libraries or packages that I needed to work with."

"[CodeScoop] saved me the trouble of having to go through and find things like undeclared variables, missing import statements, and unchecked exceptions, which prevented my [...] code from compiling [in the baseline text editor]."
Fixes Anchored In the Source Program Are Useful Too

"For me, the features of highest importance were making sure that my code would compile and run with no important left-out lines."

"[CodeScoop] fills in a lot of things that people usually don't really think about (exceptions, variables/constants) and saves a lot of time spent just searching and copy/pasting."

"Often the hardest part about writing code is finding variables and the relationships they have with the other parts of the code in the sea of text that is a program."
Divergence and Convergence on a Fine-Grained Authoring Choice: How to Fix Missing Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Add Code</th>
<th>Insert Literal</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLUMN_INDEX_ID</td>
<td>18</td>
<td>5 6 15</td>
</tr>
<tr>
<td>COLUMN_INDEX_NUM_PAGES</td>
<td>18</td>
<td>5 12 15</td>
</tr>
<tr>
<td>COLUMN_INDEX_TITLE</td>
<td>18</td>
<td>5 6 12 15</td>
</tr>
<tr>
<td>COLUMN_INDEX_YEAR</td>
<td>18</td>
<td>5 12 15</td>
</tr>
<tr>
<td>num_pages</td>
<td>5 18</td>
<td></td>
</tr>
<tr>
<td>QUERY</td>
<td>5 17</td>
<td>6 18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Add Code</th>
<th>Insert Literal</th>
</tr>
</thead>
<tbody>
<tr>
<td>arg0</td>
<td>3 4 8 19</td>
<td>16</td>
</tr>
<tr>
<td>priceInt</td>
<td>3 4</td>
<td></td>
</tr>
<tr>
<td>query</td>
<td>3 4 8 16 19</td>
<td>7 9</td>
</tr>
<tr>
<td>arg1</td>
<td>2 10</td>
<td>14</td>
</tr>
<tr>
<td>destination</td>
<td>1 2 10 13 14</td>
<td>11</td>
</tr>
<tr>
<td>messageHtml</td>
<td>1 2 10 14</td>
<td>11 13</td>
</tr>
<tr>
<td>password</td>
<td>1 2 13</td>
<td>11 14</td>
</tr>
<tr>
<td>sslFactoryClass</td>
<td>1 2 10 11 13</td>
<td>14</td>
</tr>
<tr>
<td>username</td>
<td>1 2 13</td>
<td>11 14</td>
</tr>
</tbody>
</table>

Task 1
Task 2
Task 3
Annotated Examples from CodeScoop Study

For each example that a participant produced with CodeScoop, we display the example with annotations of how it was constructed.

The marks in the left gutter indicate author interactions. There are three types of interactions:
- An author’s first code selections when starting CodeScoop.
- Additional lines added to the example, without any prompting from CodeScoop.
- Lines an author included in response to a suggestion from CodeScoop.

The color of a line indicates the source of the line:
- Boilerplate: this code shows up in every example so the example can compile.
- Automatic: CodeScoop inferred that this line was required, and added it automatically.
- Prompted: The participant included these lines when reviewing a prompt from CodeScoop.
- Manual: The participant included these lines without any prompting.

The labels in the right gutter are fine-grained descriptions each line’s source. There are three variants of prompted selections:
- def: adding code to define a variable
- use: adding a previous use of a variable
- control: adding a control structure that surrounds a statement
- throws: throwing an exception from the main method signature

A line is generated with the *add print statement* label when an author clicks the "Print" with a variable name selected.

Where an author replaced an undefined variable with a literal, the literal is highlighted in green, underlined, and bolded.