Lesson 8: Code Reviews

45 minutes

Overview

How do software engineers obtain feedback from peers?

Software engineers use code reviews to develop code that is easy for others to read, maintain, and modify over time. Students identify the importance of commenting and program structure to improve readability. Students learn about code reviews and practice giving and receiving feedback.

Standards

Full Course Alignment

CSA Conceptual Framework

- **MOD-2** - Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attirbutes and/or behaviors of the object or concept

Agenda

Warm Up (10 minutes)
- **CS Bingo**

Activity (30 minutes)
- **Style and Documentation**
- **The Code Review Process**

Wrap Up (5 minutes)
- **Impacts of Bad Code**
- **Assessment: Check for Understanding**
- **AP Classroom Topic Questions**

Objectives

Students will be able to:

- Conduct a code review to give and receive feedback
- Explain why it is important to use good programming style
- Write clear comments to help the reader understand the code

Preparation

- Create code review groups as indicated on the Code Reviews resource
- Print copies of the CS Bingo Cards (one card for each student)
- Print the CS Bingo Definitions resource and mix them up
- Print copies of the student version of the Code Reviews resource (one for each student)
- Check the **Teacher's Lounge** for verified teachers on the CSA Forum to find additional strategies or resources shared by fellow teachers

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the students

- **CS Bingo Cards** - Handout
- **Code Reviews** - Resource
- **Commits and the Backpack** - Video
- **Software Engineering: Code Reviews** - Video

- **U1L8 Extra Practice** - Handout

Vocabulary

- **code review** - the process of examining code and providing feedback to improve the quality and functionality of the program
- **comment** - a text note that is ignored by the compiler to explain or annotate the code
- **documentation** - written descriptions of the purpose and functionality of code
- **programming style** - a set of guidelines for formatting program code

Teaching Guide

Warm Up (10 minutes)

CS Bingo

Remarks

> We have learned a lot of new terms so far! Let's review some of these terms through a game of CS Bingo.

Distribute: Give each student a unique CS Bingo Card.

Do This: Review the instructions for playing CS Bingo.

Do This: Play the music clip to cue the CS Bingo activity. Choose a random definition and read it aloud to the class. Give students a moment to check their bingo card for the correct term, then state the correct term. Repeat until a student yells "Java."

Teaching Tip

To allow for multiple students to "win" the game, continue pulling definitions until a couple more students win as well.

Activity (30 minutes)

Style and Documentation (15 minutes)

Remarks

> We have learned a lot about writing code, and our programs are becoming more complex as we learn new concepts. As our programs grow, we must write code that is easy to read and navigate. Software engineers follow guidelines to write their code and use various tools to keep track of their work, access

> engineers follow guidelines to write their code and use various tools to keep track of their work, access code that they want to reuse in a different program, and request and give feedback.

Do This: Review the lesson objectives.

Group: Place students in pairs.

Do This: Direct students to Level 1 on Code Studio. Students use Levels 1 through 4 to explore good and bad programming styles, comments, how to commit code, how to save classes to the Backpack, and how to import classes from the Backpack with their partners.

1-4 **Programming Style, Commits, and the Backpack**

1 2 3 4

Discuss: Click through the animated slide to display the prompts.

- *What did you notice in the good and bad programming style examples?*
- *Why was one programming style better than the other?*
- *Why would we want to commit code?*
- *What is the purpose of the Backpack?*

Discussion Goal: Students identify aspects of both example programs that made it easy or difficult to read and understand. Students note that the good program example indented blocks of code and included comments to explain the purpose of code segments. Students share that committing code helps create versions of their programs, and the Backpack allows them to store and reuse classes in other programs.

Do This: Click through the animated slide to explain guidelines for good programming style and comments.

Teaching Tip

Have students share a hobby or interest. Ask students if there are guidelines for doing these hobbies appropriately and why these guidelines exist. Connect with students by sharing an example of your own.

Display: Show the video – *Commits and the Backpack*.

The Code Review Process (15 minutes)

Remarks

> Software engineers spend a lot of time reading other people's code and providing feedback. Throughout this course, you will use code reviews as a tool for improving your coding and collaboration skills.

Display: Show the video – *Software Engineering: Code Reviews*.

Distribute: Give each student a copy of the Code Reviews resource.

Do This: Explain examples and guidelines for good feedback.

Discuss: Use the Hold That Thought strategy to discuss the prompt.

- *What would be some examples of bad feedback?*

Discussion Goal: Students share examples of bad feedback and connect to scenarios where they received bad feedback. Students identify things that make feedback bad, such as not being specific or helpful to resolve a problem.

Teaching Tip

You can also have students share examples of times when they received bad feedback. Ask students why they considered it bad feedback or what they wish the person giving the feedback had said. Connect with students by sharing an example of your own.

Do This: Introduce TAG for providing feedback.

Do This: Click through the animated slide to introduce the Code Review Call and Response.

Teaching Tip

The Call and Response strategy helps build community and create routines in the classroom.

Remarks

> You have been placed in code review groups to review each other's code and make revisions to your program based on the feedback you receive.

Do This: Click through the animated slide to have students participate in the Code Review Call and Response.

Do This: Direct students to Level 5 on Code Studio. Students use Level 5 to explore the code review process and practice giving and receiving feedback.

5 **Conducting a Code Review**

Wrap Up (5 minutes)

Impacts of Bad Code

Remarks

> Good programming style and code reviews are an important part of a software engineer's work.

Discuss: Click through the animated slide to display the prompts.

- *What surprises you about this visual? What do you want to know more about?*
- *Approximately $85 billion is lost from developer time spent on bad code each year. How can software engineers minimize this time?*

Discussion Goal: Students consider the broader role of a software engineer and the various tasks they complete as part of their work. Students identify strategies to minimize time spent on bad code, such as using good programming style and planning and reviewing code.

Do This: Review the concepts covered in this lesson.

Display: Key Vocabulary

Assessment: Check for Understanding

Check For Understanding Question(s) and solutions can be found in each lesson on Code Studio. You can use these questions as an exit ticket.

6 **Check for Understanding**

AP Classroom Topic Questions

To assign questions from the AP Classroom Question Bank that align with this lesson, create a custom quiz in AP Classroom by searching the Question Bank for the Essential Knowledge statements listed at the top of this lesson plan. You can find instructions and video demonstrations to do this on **AP Central**.