Lesson 1: Project Planning

45 minutes

Overview

How can I identify the requirements and things I need to know to complete a project?

Students are introduced to the Natural Language Processing Project and evaluate requirements and examples to identify questions and key features. Before beginning work on the project, students expand their understanding of program structure and comments by identifying best practices and exploring the use of Javadocs to generate program documentation.

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)
- **Natural Language Processing**

Activity (30 minutes)
- **Project Introduction**
- **Documentation**

Wrap Up (5 minutes)
- **Three W's**
- **Assessment: Check for Understanding**
- **AP Classroom Topic Questions**

Objectives

Students will be able to:

- Identify project requirements
- Write single-line, multi-line, and Javadoc comments

Preparation

- Print copies of the Unit 6 Guide (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

- **Project Characteristics** - Handout
- **U6L1 Extra Practice** - Handout
- **Unit 6 Guide** - Handout

Vocabulary

- **HTML** - Stands for Hypertext Markup Language; the standard system for tagging text files to be displayed on the World Wide Web
- **Javadocs** - The documentation tool for Java that generates an HTML document from comments written inside `/** */` and formatted using `@` tags
- **Natural Language Processing (NLP)** - The ability of a computer

program to understand human language
- **documentation** - written descriptions of the purpose and functionality of code

Teaching Guide

Warm Up (10 minutes)

Natural Language Processing

Remarks

> Throughout this unit, we will explore how natural language processing is used to understand the structure and meaning of text. We will learn common strategies and techniques used to develop programs that can understand, manipulate, and respond to human language.

Do This: Define *natural language processing (NLP)*.

Display: Show the video – *Software Engineering: Natural Language Processing*.

Discuss: Click through the animated slide to display the prompts. Use the Retrieve-Pair-Share strategy to discuss the prompts.

- *What are some other examples of where you have seen or experienced NLP?*
- *What are some examples of where you think NLP could be useful?*

Discussion Goal: Students share examples that they have seen or experienced, such as chatbots on websites or virtual assistants. Students suggest applications of NLP that they feel could be useful.

Activity (30 minutes)

Project Introduction (15 minutes)

Do This: Review the lesson objectives.

Remarks

> For the unit project, you will create a program that uses natural language processing techniques to analyze, manipulate, or generate text. Let's start by reviewing the project requirements to identify what we know and what we need to know, then we will look at some examples to identify specific characteristics and features that we need to develop.

Do This: Introduce the project requirements and rubric.

Discuss: Click through the animated slide to display the prompts. Use the Retrieve-Pair-Share strategy to discuss the prompts.

- *What are things in the project requirements and rubric that we already know?*
- *What are some things we need to know?*

Discussion Goal: Students identify concepts they already know in the project requirements and the rubric, such as file and user input. Students suggest things they need to know, such as how to write algorithms using natural language processing techniques, `ArrayList`s, and `String` manipulations.

Remarks

> Now that we have an idea of what we know and what we need to know, let's take a look at some examples of this project to understand the type of work you will need to complete.

Group: Place students in pairs.

Distribute: Give each pair a copy of the Project Characteristics handout.

Do This: Direct students to Level 1 on Code Studio. Students analyze the example projects and generate a list of common characteristics and key features on the Project Characteristics handout.

1 **Natural Language Processing Project Examples**

Teaching Tip

Emphasize to students that while these are examples of the unit project, they are not representative of the only ways they can approach or develop the project.

It may be helpful to model the process for students first, then facilitate a discussion about the key features of the example projects they reviewed.

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

- *What words and phrases are in the rubric that aligns with your list of criteria?*

Discussion Goal: Students make connections between the features of the example projects that they noticed and the project's requirements. Students also notice that there are multiple ways to achieve each requirement.

Documentation (15 minutes)

Remarks

> Now that we have a solid understanding of the requirements of this project, we can start working through the things we identified that we need to know. We will be learning a lot throughout this unit that will be useful for completing this project. First, let's learn about documenting our code.

Distribute: Give each student a copy of the Unit 6 Guide.

Discuss: Use the Retrieve-Pair-Share strategy to discuss the prompt.

- *Based on our experiences with writing and reading each other's code, what are some best practices we should follow when writing and documenting code?*

Discussion Goal: Students share strategies to write and document code, such as naming conventions for variables and methods, using white space and indentation to make code easily readable, and writing comments to explain the purpose of individual lines or blocks of code.

Teaching Tip

You could choose to capture the strategies shared on poster paper, a presentation slide, or a digital whiteboard.

Do This: Define *documentation*.

Remarks

> Good documentation makes it easier for other software engineers to understand what is going on in your code. There are multiple ways to document your code, such as the single-line and multi-line comments we use in our programs already.
>
> Today we're going to learn about one more commenting technique that can create documentation that looks like the documentation we've been looking at and using throughout the year.

Do This: Click through the animated slide to identify the types of documentation in the open-source code example and introduce Javadocs.

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

- *What do you notice? What do you wonder?*

Discussion Goal: Students connect the comments shown in the open-source code example to comments they have seen or written in previous programs. Students identify the `@` tags and might wonder if there are other tags, the purpose of the tags, or if the tags are used somewhere else in the program.

Teaching Tip

Guide students to notice the differences between multi-line comments and Javadocs. Ask students how multi-line comments are written and have them compare those to Javadocs comments.

Do This: Click through the animated slide to explain that Javadocs generates an HTML document from comments written inside `/** */` and formatted using `@` tags and to define HTML.

Do This: Click through the animated slide to introduce the Javadocs tags for specifying parameters and returns.

Do This: Direct students to Level 2 on Code Studio to add Javadocs comments to their `Dessert` class. Have students share their documentation with a neighbor for feedback regarding the clarity and structure of their comments.

2 **Documenting Program Code**

Teaching Tip

Students will likely not have time to document the entire `Dessert` class. Students should focus on adding documentation where indicated in the level instructions and then sharing their work with a neighbor for feedback.

Wrap Up (5 minutes)

Three W's

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

- *What did we learn today?*
- *So what?*
- *Now what?*

Discussion Goal: Students share the concepts they learned from the lesson, including how to identify project requirements and characteristics from examples and create Javadocs documentation. Students suggest problems where they might use these natural language processing strategies and Javadocs documentation and make predictions about upcoming lessons or problems.

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. These questions can be used for an exit ticket.

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**.