Lesson 4.2: Software Development Life Cycle - The Design Phase

Objectives

In this lesson, students will:

- Be introduced to the Software Development Life Cycle
- Learn the importance of the design phase
- Experience and practice creating a program design and design documentation

Agenda

1. Software Development Life Cycle	10 mins
2. The Design Phase	5 mins
3. Student activity: Design Your Game	25 mins
4. Wrap Up and Reflections	10 mins

Preparation

- ❑ Print student activity worksheet (one per student pair)

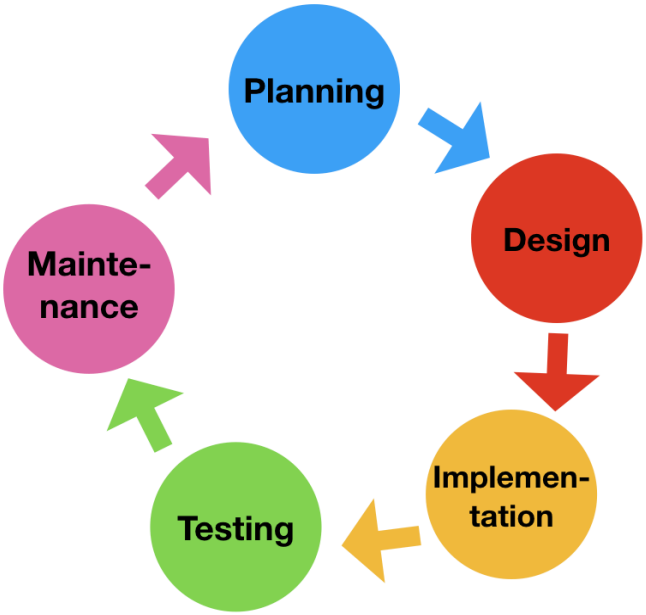
Resources & Links

- ❑ None

1. Software Development Life Cycle (SDLC)

Engage students in an interactive discussion and instruction (the lifecycle image is available in exhibit A).

You may have studied the life cycle of a butterfly or some other animal. Well, software also has a life cycle. Software is programs and code. The software development life cycle is a process used by the software industry to design, develop and test software.



There are different types of software development life cycles, but typically they will include these 5 phases:

1. **Planning**: a plan is put together, breaking down the project into smaller tasks, usually with dates when each task needs to be completed.
2. **Design**: designers or software engineers put a design together to be reviewed to make sure it is what the user wants and that it will work.
3. **Implementation**: this is the code writing phase.
4. **Testing**: Testing is done by the QA team (remember last lesson). Testing involves creating a test plan with test cases, running the test cases, reporting any bugs found while testing and giving feedback about the program.

   Finally, after the bugs have been fixed or changes made as a result of feedback, QA typically re-runs the test case to make sure the bug was fixed and no new ones showed up.
5. **Maintenance**: the software product will need updates from time to time because a user or customer found a bug or something changed (like the operating system) so the program needs to be updated.

These phases are not done as just a linear exercise, meaning you do one, then the next until you are done. The design is typically reviewed to get feedback to make sure it accomplished what was intended. If it does not, designers have to go back and change the design.

Likewise, the plan, test plan and code are also reviewed to get feedback. At every phase, feedback is typically gathered so that the software is completed with high quality. Sometimes during testing or coding, the team realizes the design or the code also needs to change. Often the team is running late and the plan needs to be changed or some tasks need to be skipped. The software development life cycle, as you can see, is a very circular exercise.

You, as a software engineer, are going to design, code and test a game and gather feedback. This will take place over the next several lessons.

2. The Design Phase

During the design phase you document the ideas with words and sketches of how a program is supposed to work, what are the parts, what are the inputs and outputs, and how it flows.

Imagine building a house without a plan first? How would you know how big it is supposed to be, how many rooms, what kind of rooms? The design phase of developing a program is very important for a couple of reasons.

First, it is much easier (and cheaper if somebody is paying for it) to change sketches and words than written code. Once your design is finished it is reviewed by others to make sure the intention is captured and nothing was forgotten. If changes need to be made, you simply change the sketches and documentation. Often a design is reviewed many times. The documentation is also useful when another engineer needs to test the program.

The other reason designing is important is because what you create will be better. You had a chance to think about everything and get feedback from others. You can then apply feedback to your design to make improvements.

Programs will be better when they go through iterations with a chance to apply improvements.

3. Student Activity: Design Your Game

Students will work in pairs to design, code and test a game. In this activity, students will design and document the design of their game.

The design will also need to be realistic. They will have 2 lessons to code their game. Let students know their game has to be able to be coded in that amount of time. Time restrictions exist for any software development project.

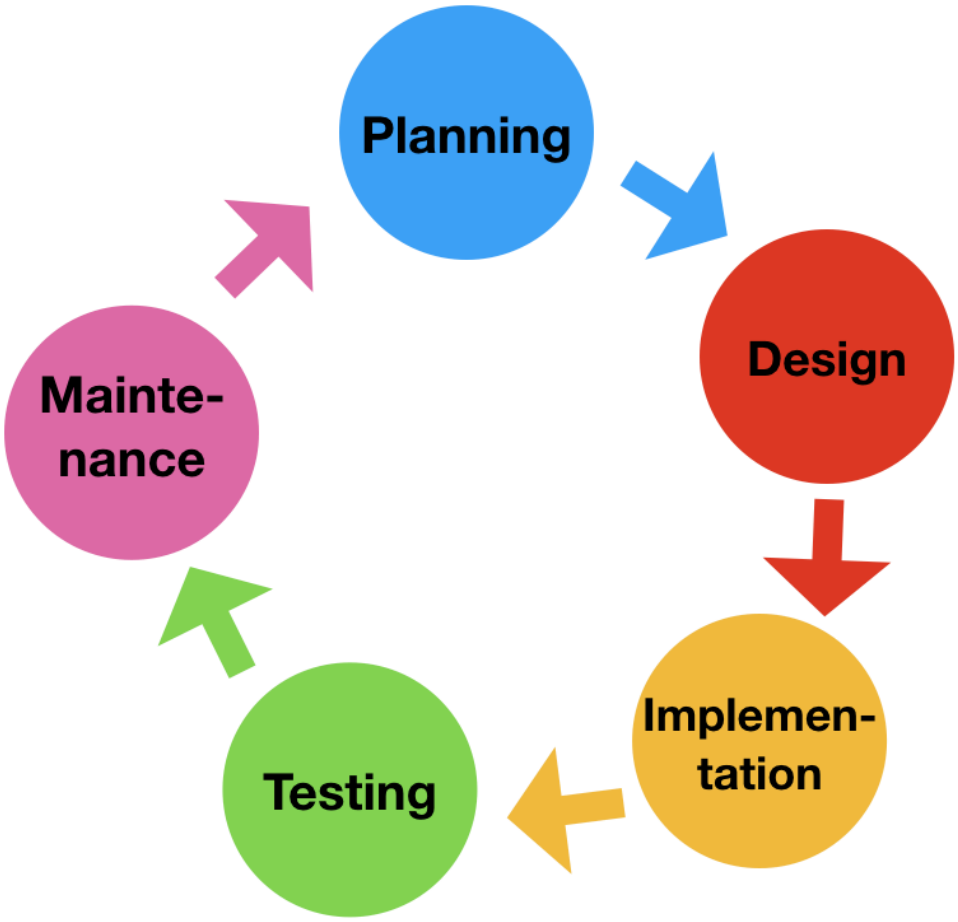
Distribute the student activity worksheet which helps guide students through the design process. Encourage them to fill out as many sections and details as possible.

4. Wrap Up and Reflections

Reflection Points:

- What did you like about today's activity?
- What was challenging during the design phase?
- Why is it important to design a program before coding it?
- What happens after the design is done? (Answer: Review and revise if necessary)

Exhibit A:



Student Activity Worksheet: Design your Game

1. Describe your game (remember somebody that may need to change or test your game needs to understand what the game is about)

2. Will the game keep score? If so, what does the score represent? How do you win the game?

3. What is the game flow, what does the stage look like from one step to the next?

What's happening?

What's happening?

What's happening?

What's happening?

4. List any sprites in your game

Sprite name	What does the sprite do or represent?	What is the first position when the game starts?

5. The game may need variables to store things like the score or a timer or to track things.

Variable Name	What is it used for?	What is the initial value?	What causes it to change?