# Software Development and Design Summary

3.7.1

## What Did I Learn in this Module?

**Software Development**

The software development life cycle (SDLC) is the process of developing software, starting from an idea and ending with delivery. This process consists of six phases. Each phase takes input from the results of the previous phase: 1. Requirements & Analysis, 2. Design, 3. Implementation, 4. Testing, 5. Deployment, and 6. Maintenance. Three popular software development models are waterfall, Agile, and Lean:

- **Waterfall** - This is the traditional software development model. Each phase cannot overlap and must be completed before moving on to the next phase.
- **Agile Scrum -** In rugby, the term scrum describes a point in gameplay where players crowd together and try to gain possession of the ball. The Scrum methodology focuses on small, self-organizing teams that meet daily for short periods and work in iterative sprints , constantly adapting deliverables to meet changing requirements.
- **Lean** - Based on Lean Manufacturing, the Lean method emphasizes elimination of wasted effort in planning and execution, and reduction of programmer cognitive load.

**Software Design Pattern**

Software design patterns are best practice solutions for solving common problems in software development. Design patterns are language-independent. In their Design Patterns book, the Gang of Four divided patterns into three main categories:

· **Creational** - Patterns used to guide, simplify, and abstract software object creation at scale.

· **Structural** - Patterns describing reliable ways of using objects and classes for different kinds of software projects.

· **Behavioral** - Patterns detailing how objects can communicate and work together to meet familiar challenges in software engineering.

The observer design pattern is a subscription notification design that lets objects (observers or subscribers) receive events when there are changes to an object (subject or publisher) they are observing.

The Model-View-Controller (MVC) design pattern is sometimes considered an architectural design pattern. Its goal is to simplify development of applications that depend on graphic user interfaces.

**Version Control Systems**

Version control is a way to manage changes to a set of files to keep a history of those changes. There are three types of version control systems: Local, Centralized, and Distributed.

Git is an open source implementation of a distributed version control system. Git has two types of repositories, local and remote. Branching enables users to work on code independently without affecting the main code in the repository. In addition to providing the distributed version control and source code management functionality of Git, GitHub also provides additional features such as: code review, documentation, project management, bug tracking, and feature requests. After installing Git to the client machine, you must configure it. Git provides a git config command to get and set Git's global settings, or a repository's options. Git has many other commands that you can use, including a host of branching options. Developers use a .diff file to show how two different versions of a file have changed.

**Coding Basics**

Clean code is the result of developers trying to make their code easy to read and understand for other developers. Methods and functions share the same concept; they are blocks of code that perform tasks when executed. If the method or function is not executed, those tasks will not be performed. Modules are a way to build independent and self-contained chunks of code that can be reused. In most OOP languages, and in Python, classes are a means of bundling data and functionality. Each class declaration defines a new object type.

**Code Review and Testing**

A code review is when developers look over the codebase, a subset of code, or specific code changes and provide feedback. The most common types of code review processes include: Formal code review, Change-based code review, Over-the-shoulder code review, and Email pass-around.

Software testing is subdivided into two general categories: functional, and non-functional. Detailed functional testing of small pieces of code (lines, blocks, functions, classes, and other components in isolation) is usually called Unit Testing. After unit testing comes Integration Testing, which makes sure that all of those individual units fit together properly to make a complete application. Test-Driven Development (sometimes called Test-First Development) is testing to validate the intent of the design in light of requirements. This means writing testing code before writing application code. Having expressed requirements in testing code, developers can then write application code until it passes the tests.

**Understanding Data Formats**

Today, the three most popular standard formats for exchanging information with remote APIs are XML, JSON, and YAML.

Extensible Markup Language (XML) is a derivative of Structured, Generalized Markup Language (SGML), and also the parent of HyperText Markup Language (HTML). XML is a generic methodology for wrapping textual data in symmetrical tags to indicate semantics. XML filenames typically end in ".xml".

JavaScript Object Notation JSON), is a data format derived from the way complex object literals are written in JavaScript. JSON filenames typically end in ".json".

YAML Ain't Markup Language (YAML) is a superset of JSON designed for even easier human readability.

Parsing means analyzing a message, breaking it into its component parts, and understanding their purposes in context. Serializing is roughly the opposite of parsing.

3.7.2

## Module 3: Software Development and Design Quiz

1. Which software development methodology prescribes that developers follow a strict process order by completing one step in the SDLC process before proceeding to the next step.

   ◯ Lean
   ◯ Agile
   ◯ Scrum
   ◯ Waterfall

2. Which SDLC development methodology employs many quick iterations known as sprints?

   ◯ Waterfall
   ◯ Lean
   ◯ Extreme Programming
   ◯ Agile

3. Which two programming components are defined as blocks of code that perform tasks when executed? (Choose two.)

   ☐ functions
   ☐ arguments
   ☐ methods
   ☐ objects
   ☐ parameters

4. A developer wants to find the location of the Python 3 executable file. Which command should the developer use?

   ◯ **find python3**
   ◯ **which python3**
   ◯ **where python3**
   ◯ **locate python3**

5. Which SDLC phase concludes with functional code that satisfies customer requirements and is ready to be tested?

   - ◯ deployment
   - ◯ implementation
   - ◯ testing
   - ◯ maintenance

6. What are the three states of a Git file? (Choose three.)

   - ☐ committed
   - ☐ modified
   - ☐ secured
   - ☐ staged
   - ☐ locked
   - ☐ deleted

7. Which term is used to describe the first line of an XML document?

   - ◯ prologue
   - ◯ preamble
   - ◯ introduction
   - ◯ opening

8. How does an application use a module in Python?

   - ◯ by calling the module name
   - ◯ through the **include** statement
   - ◯ by using an assignment statement with a variable
   - ◯ through the **import** statement

9. What is the role of the controller component in the Model-View-Controller (MVC) flow?

   - ◯ It takes user input and manipulates it to the proper format for the model.
   - ◯ It provides visual representations and presentations of the data.
   - ◯ It takes in user input and manipulates it to fit the format for the model or view.
   - ◯ It accepts selected data and displays it to the user.

10. Which code review method involves the developer going through the code line-by-line with the reviewer, allowing the developer to make changes on the spot?

   - ◯

- over-the-shoulder
- change-based
- formal
- email pass-around

Check Show Me

Reset

3.7.3

## Project Activity 2: Agile Team Formation

In this activity, you will complete the following tasks:

- Choose a project.
- Build an agile team.
- Assign project roles.
- Select communication channels that will facilitate the team's work.

Refer to the **DEVASC Project Rubric** below to record your process and outcomes.

Working in technology today means working collaboratively and effectively as a cross-functional team. To prepare for the continuous and accelerating pace of change, you need to:

- Think critically about issues, solve problems creatively
- Work collaboratively
- Communicate clearly in multiple media channels
- Quickly learn ever-changing technologies
- Deal with a flood of information

The rapid changes in our world require you to be flexible, take the initiative, lead when necessary, and to produce something new and useful.

### Scenario

You have been hired as a junior automation engineer to help your company automate its IT infrastructure. You are working on a **daily basis with Python-based automation code directly on your computer**. However, your **team is expanding with new members** and now, you all need a way to collaboratively work on the code. Your manager **requires all members to move the code from their individual computers to GitHub** and to collaborate with the team through the GitHub platform.

### 1. Choose Agile Team Roles

Your team will now create a formal agile team. Each member of your team will have a role during this project. The specific tasks that each team member performs during the project will be decided later.

The roles of an agile team may vary depending on:

- The size of the team
- The nature of the task
- Company best-practices and policies

## Scrum Leader

At a minimum, an agile team will have a "scrum leader". The scrum leader will help obtain any resources needed and take the lead in resolving any questions or problems the team may have. This role makes use of the "soft skills" of project management and communication.

**Note:** The original term is "scrum master." In April 2019, the IETF began work on a new Internet-Draft RFC, Terminology, Power and Oppressive Language. This RFC describes alternatives that shift specific language conventions used by RFC Authors and RFC Editors to avoid oppressive terminology in the technical documentation of the RFC series. The RFC refers specifically to the "master-slave" metaphor as offensive. The authors of this course have decided to replace the term "master" with "leader."

At this point, your team will now select a **scrum leader**. The responsibilities of the scrum leader include:

1. Clearing any obstacles or problems that may arise.
2. Establishing an environment in which the team can work and communicate effectively.
3. Addressing team dynamics to ensure that team members are supportive and working together.
4. Protecting the team from outside interruptions and distractions and acting as the intermediary between the team and the project owner (instructor).
5. Ensuring that the information required by the rubric is documented. (The actual documentation may be done by a recorder.)

**Rubric/Deliverable:**

- Record the name of your scrum leader.
- How did your team select the scrum leader?

**Note:** Your instructor may choose to have a rotating scrum leader.

## Recorder

Select someone to be the **recorder**. This person will be responsible for entering and maintaining the **DEVASC Project Rubrics.** Other responsibilities of the recorder include:

- Recording all meeting days and times.
- Working with the scrum leader to ensure the information in the rubrics is accurate and complete.

**Note:** In a smaller team (less than four people) the scrum leader may also assume the role of recorder.

## Other Roles

Search the internet for "agile team roles" and discuss as a team if any other roles at this point are necessary. Your team can always make changes and adjustments throughout the project.

## 2. Team Communications

In the final step of this activity you decide how your team will communicate and collaborate during this process. Discuss some of the various tools which are available to everyone such as:

- WebEx Teams, or other instant messaging platform
- Webex Meetings, or another video communications platform
- Email
- Scheduled meetings

Document the methods your team will be using and make sure everyone on the team has access these tools.

**Rubric/Deliverable:** Document the communication tools your team will be using and the purposes of each.

The **Rubric/Deliverables** for this activity were:

- Scrum leader information
- Other Agile team roles if applicable
- Team communication tools and purpose of each

Project Activity 2 - Agile Team Formation Rubric

3.6 Understanding Data Formats

4.0 Introduction