Listen

Module Two

Learning Objectives

By the end of this module, you will meet these learning objectives:

- Identify various design constraints to recommend solutions to address a specific problem
- Demonstrate best practices when creating a software design template
- Use software design patterns to efficiently solve a problem

Module Overview

In computer science, design patterns are established, accepted approaches to efficiently solving common problems. Technically, a design pattern is not program source code, but rather a generalized description of *how to solve the problem* that can be implemented in any programming language. However, the Internet contains many examples of each design pattern implemented in most modern programming languages. This distinction is extremely important to understand:

- **A pattern is a proven, documented approach and not necessarily source code**.

A design pattern describes a proven, tested approach, usually in the form of a software design template containing UML diagrams and descriptive text. Software design templates have evolved into a common format to document the approach to be taken on

a project. Over time, many projects will be undertaken and using a common format ensures consistency and familiarity in documenting all of them.

Design constraints are a part of every project that must be uncovered and surfaced to inform the decisions made. Budget, current skill sets, existing infrastructure, and timeline all contribute to the design approach chosen.

Design patterns can speed up the development process by using the existing and widely accepted solutions. Many of the most common computer science problems have already been solved with well-established approaches. It is equally important to know what problem(s) the pattern is designed to solve.

A common trap that beginning and intermediate programmers fall into is to try using as many patterns as they can, thinking that these are the key to making the best software possible. The best approach is to first understand the problem and then research to determine if there is a pattern that fits well in the context of the application you are building.

This module introduces the topic of patterns and starts you on your journey to understanding and using patterns effectively. As you will learn later in this course, subsequent works have focused on architectural patterns (used for larger scale problems concerning the overall architecture of systems) and enterprise integration patterns (focused on how enterprise-scale systems interact with one another when exchanging data).

Module at a Glance

This is the recommended plan for completing the reading assignments and activities within the module. Additional information can be found in the module Resources section and on the module table of contents page.

1. Review the Module Two resources.

2. Complete the Module Two assignment.
3. Complete the Project One Milestone.
4. Review Project One.