



Resources: Operating Platforms and UML Diagrams



TidBIT

Unified Modeling Language (UML) is intended to provide a standard way of visualizing a design. On a single drawing or sheet of paper, you can grasp the organization and essence of an application design. The alternative would be to sit down and read hundreds or thousands of lines of code while keeping what you are reading straight in your head, to form a mental picture of how all the classes relate to one another and where pieces of data “live.” UML has a standard and strict set of rules for shapes, the various types of diagrams, and what they mean and are used for.

Software is more about design and organization than about the language and syntax rules of curly braces and semicolons. You should be able to look at a UML class diagram and picture the code that is built from it. Conversely, given a set of classes, you should be able to create a UML class diagram from them. There are even commercial tools that can do that translation back and forth for you: from diagram to code and back. However, you need to understand the concepts and learn that ability for yourself before using tools to do it for you.




Required Resources

Reading: *Hands-On Design Patterns With Java*, Chapters 1 and 2

Chapter 1 provides an overview of Unified Modeling Language (UML), with a specific focus on diagrams applicable to the hands-on activities in this book. Review this information, which includes the basic components and a simple example of a UML class diagram, as a refresher on this type of diagram.

In Chapter 2, you will learn about object-oriented programming principles and design patterns. Chapter 2 is a GitHub repository of sample code. Given this code, practice creating UML diagrams that represent the code.


Reading: [Operating System Concepts](https://go.oreilly.com/SNHU/library/view/operating-system-concepts/9780470128725/silb_9780470128725_oeb_c01_r1.html) 

(https://go.oreilly.com/SNHU/library/view/operating-system-concepts/9780470128725/silb_9780470128725_oeb_c01_r1.html), Chapter 1 (1.1–1.3 and 1.5–1.10) and Chapter 2 (2.1–2.8 and 2.10–2.12)

Chapter 1 provides foundational information on operating systems. The chapter also defines and further explores topics including computer-system architecture, operating-system architecture, process, memory, and storage management. Chapter Two offers an overview of operating system structures. The chapter outlines specific topics such as types of system calls, systems programs, operating-system design and implementation, and operating-system generation.




Additional Support (Optional)

Video: [UML Class Diagram Tutorial](https://www.youtube.com/watch?v=UI6lqHOVHic)  (<https://www.youtube.com/watch?v=UI6lqHOVHic>) (10:16)

Watch this video from Lucidchart to review classes, attributes, and methods for creating UML class diagrams. Examples of inheritance, aggregation, and composition

relationships are also covered for you to review.

Reading: [What's the Difference Between a Software Product and a Platform?](https://www.forbes.com/sites/adrianbridgwater/2015/03/17/whats-the-difference-between-a-software-product-and-a-platform/) 

(<https://www.forbes.com/sites/adrianbridgwater/2015/03/17/whats-the-difference-between-a-software-product-and-a-platform/>)

This article gives an overview of the differences between software products and platforms.



Reflect in ePortfolio



Download



Print



Open with docReader



Activity Details

You have viewed this topic



Explore these resources, which will help you learn how to identify both the structures of various operating systems and the essential functions and characteristics of operating platforms. You will also learn to create UML diagrams that depict properties of and relationships between classes from code.

Last Visited Jan 6, 2025 7:23 PM