

# Module Four

---


## Learning Objectives



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

- ☑ Utilize the various services provided by an operating platform to satisfy requirements for basic system functionality
- ☑ Create an application illustrating the RESTful API architectural approach

## Module Overview

Unless you are developing a single-user desktop application or a simplistic, standalone mobile application, you will be using a distributed architecture. Distributed applications need to communicate between their various parts, and REST-style APIs built to communicate over HTTP are the most common and preferred method in use today.

There are many lower-level details to managing communication between applications, such as serializing object instances from memory in order to safely transmit them to another machine or location. The HTTP protocol requires standard or commonly used and accepted code to set up, manage, and examine requests sent and responses received from another application running elsewhere. Coding all of this lower-level logic would take a significant amount of time to create, debug, and polish the code. This is where application libraries come into play, including Jetty  (<https://www.eclipse.org/jetty/>) for

HTTP communication, Jersey  (<https://eclipse-ee4j.github.io/jersey/>) for creating RESTful APIs, and Jackson  (<https://github.com/FasterXML/jackson>) for JSON object creation.

Pre-built libraries can dramatically accelerate development productivity by allowing you to focus on the business functionality you are trying to create; yet it still takes time to research and learn how to integrate each one of the libraries together. This includes determining which versions to use and which are compatible with each other. Because this part of the process is tedious, many developers do not want to waste their time and effort on completing the work.

The open source project Dropwizard was created for this very reason. Dropwizard is like a starter kit with exceptional strength. It takes care of setting up your application project with references to the correct versions of all required libraries as well as standard or commonly used and accepted starter code. With Dropwizard, you can be up and running with the shell of a REST-based application ready for distributed communication in one sitting!

## 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 Four resources.
- 2** Practice using Maven.
- 3** Complete the Module Four assignment.
- 4** Ask questions and provide advice to your peers in this week's discussion.

**5** Submit a journal entry to your instructor.

**6** Review Project Two.