Modules

In this lab, you will create a Node module that performs simple calculations for various shapes.

# Objectives

In this lab, you will learn to

* create a Node module, and
* use the newly created module within a calling application.

We are going to create a directory module named ‘shapes’, which resides in the lab folder.  We must create the 'package.json' file to indicate the name and the JavaScript program.  Our module will export three additional modules for rectangle, triangle, and circle.

# Set things up

Let's start our setup by creating an appropriate place for the module we’re going to create.

## Create a shapes directory

1. Create a directory called shapes and create a package.json file within it. Remember a package.json needs keys name and main. Go ahead and use index.js for the main key and we will create it next.

Ok, we're now ready to start coding our module!

# Create the module

Since we want our shapes module to be flexible enough to handle multiple types of shapes, let’s create an index file and reference the individual supported shapes there.

## Create an index.js file

2. Create an index.js file in the shapes directory and add require statements for rectangle, circle, and triangle, which we are about to create.

## Export rectangle, circle, triangle

3. After the require statements, export each of them, making them available to users of our module.

A module can require the use of other modules internally but not necessarily export them. For example, a popular node module express requires cookie, a separate cookie parsing module internally.

## Create rectangle.js, circle.js, and triangle.js

4. Now that the index file is complete, let’s create implementation files for each shape. In case your geometry is rusty, here are some formulas you will include as functions in each shape.

Circle

* a = πr2
* c = 2πr

Rectangle

* a = wh
* p = 2(h + w)

Triangle

* a = **½**bh

You can export each function you create with the syntax:

exports.name = function(…) {…}

For objects with multiple functions, an alternative syntax is:

module.exports = {

someFunction: function(…) {…},

someOtherFunction: function(…) {…}

}

# Write the main program logic

It's time to write the application that will use our newly created module.

## Create geometry.js and require the shapes module

5. Create a geometry.js file and require the shapes module you created.

## Compute some interesting shape values!

6. All that is left now is to write some code using the functions that you exported from your module.

When you're seeing the messages that you expect, you have completed the lab!