Course:	**INFO3114**
Professor:	**Jim Cooper**
Project:	**Project #2 – Device Tree Form – Version 1.0**
Due Date:	**Friday, August 14, 2020**
Submitting:	**Please see the last page for instructions.**

How will my project be marked?

- This project counts for 15% of your final grade and will be evaluated using the following grid:

Marks Available	What are the Marks Awarded For?	Mark Assigned
2	Good coding style including proper indentation and use of variable and object naming conventions and suitable comments	
1	Display page correctly and with good styling	
3	Submission includes code for all class types with required methods and property getters and setters	
5	Ability to create objects of all 3 types with correct inheritance chains	
3	Enable the user to select and edit all properties for the currently selected object	
1	Proper zipped submission	
15	Total	

Project Description

This project is about starting with a class-hierarchy design and then using that design to build a simple web user interface to enable viewing and editing the properties of any objects created.

The main concept here is to create a simulated controller that allows all devices of a specific type to be monitored, updated, enabled and disabled.

Specific Requirements:

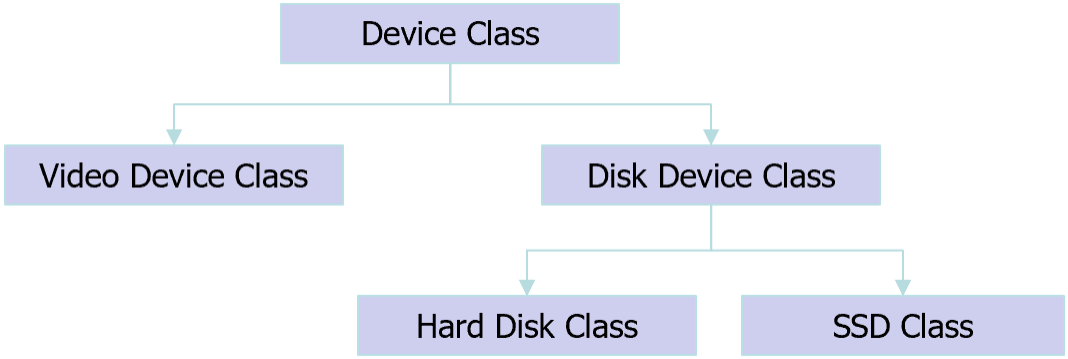
Create the following set of classes.

1. Create a base class called “Device”

2. Create two derived classes “VideoDevice” and “DiskDevice” that both inherit from “Device”

3. Create two more derived classes “HardDisk” and “SSD” that both inherit from “DiskDevice”

For the purpose of this project, “VideoDevice” does not have any inherited classes.



Device Class:

This class needs two basic methods:

enable();
disable();

You’ll also need a built-in boolean property called _status and a getter method called status. The two methods are used to control the status property (false = disabled and true = enabled).

Several other properties that you’ll want to track in the base class are:

Replacement Cost ($)
Supplier Name (string)
Serial Number (string)

Video Device Class:

This class needs the following properties:

Resolution (string) – example: “3840 x 2164”
Type (string) (one of LCD, LED, Plasma)

Disk Device Class:

This class needs the following properties:

Size: (string) – example: “4TB”
Transfer Rate (string) – example: “7GB / second”

Hard Disk Class:

This class needs the following properties:

Platter Size: (string) – example: “2.5 inches”
Number of Platters (integer)

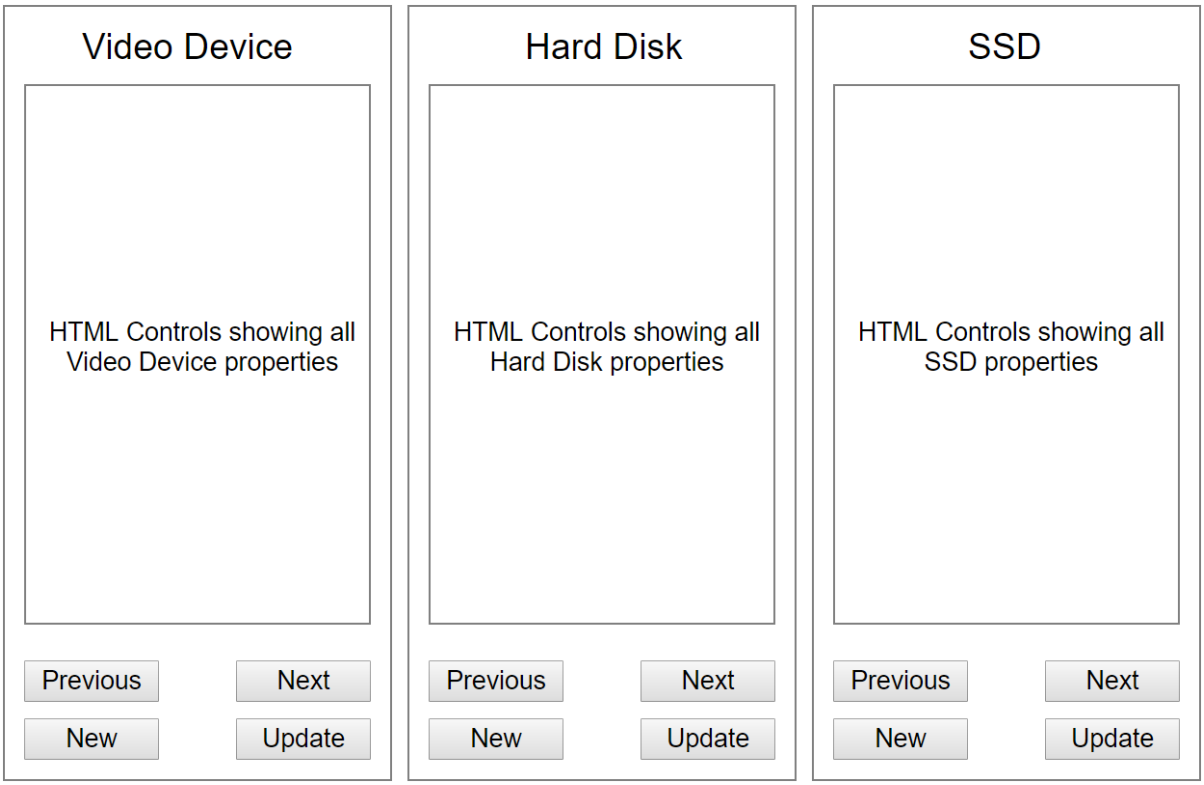
SSD Class:

This class needs the following properties:

Type: (string) (one of Flash, DRAM)
Wear Leveling: (boolean)

Create a user interface to manage the objects you create:

Your UI should consist of 3 panels, each of which provides the tools to manage one of, a video device object, a hard disk object or an SSD object (see the following image).



Each panel should be able to:

1. Create a new object of the specific type (including all inherited properties)
2. Update all properties for the currently visible device object.
3. Switch to the next (or previous) device and edit (or view) its properties.

Your UI should contain controls such as text boxes for editing string values (no character filtering is required), pulldowns (or radio buttons) for selecting a “one of” value, and a check box for each device type to change the status (checked = enabled and unchecked = disabled).

You’ll need to keep an array of each object type to facilitate the ability to switch to the next or previous object.

Also remember that each panel must provide the ability to edit all properties from the base class on down through all derived classes.

We are not including any way to delete a device object although that might reasonably be included in this kind of application.

*** End of Requirements ***

How should I submit my project?

Electronic Submission:

Submit your program file(s) to the *Info3114 "Project 2"* electronic submission folder in *FOL*. These files should be submitted as a single "zip" file containing your web application's complete website.

I strongly recommend that you test your own submission to ensure that nothing has been missed.

Submit your project on time!

Project submissions must be made on time! Late projects will be subject to divisional policy on missed test and late projects. In accordance with this policy, no late projects will be accepted without prior notification being received by the instructor from the student.

Submit your own work!

It is considered cheating to submit work done by another student or from another source. Helping another student cheat by sharing your work with them is also not tolerated. Students are encouraged to share ideas and to work together on practice exercises, but any code or documentation prepared for a project must be done by the individual student. Penalties for cheating or helping another student cheat may include being assigned zero on the project with even more severe penalties if you are caught cheating more than once. Just submit your own work and benefit from having made the effort on your own.