DSD Lab 7: Spring automatic wiring (max: 12p)

Use Spring Tool Suite: https://spring.io/tools/sts/all

Use the Simple Spring Maven project template in these tasks: **New Project > Spring Legacy Project > Simple Spring Maven**

From this lab onward, use Java-based configuration.

TIP: See *stereo-autoconfig* project in Spring Essentials lecture notes in e-class.

Youtube tutorial video:
https://www.youtube.com/watch?v=IWc6Qo-qUZ0&list=PLyCD4xKABPa4V4KLtBoZXtDDZ4WRJg68z

TASK 1: Autowired cars and weather station (4p)

Modify your Lab 6, Task 1 and Task 2 to use autowiring instead of manual wiring. You can also use the model solutions in Car_model.zip and WeatherApp_model.zip.

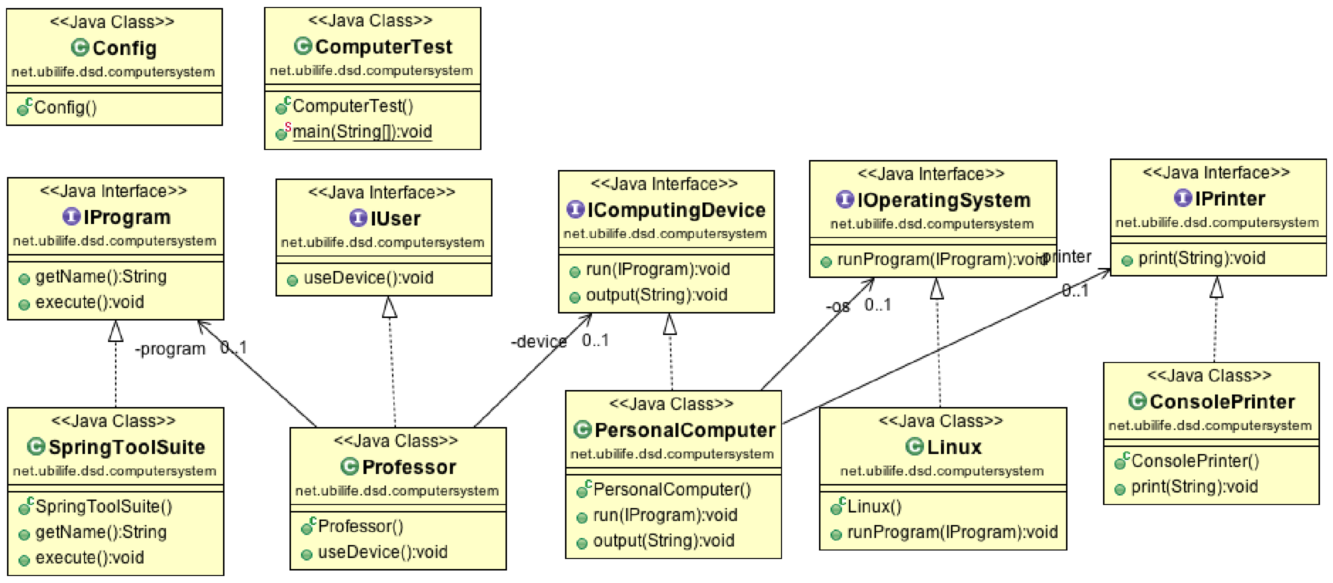
Use Java-based configuration!

TIPS:

- Each bean class must have a default constructor (without parameters)
- Autowiring steps
  - Enable component scan in configuration
  - Annotate beans with @Component
  - Apply @Autowired on dependencies

TASK 2: Autowired computer system (6p)

Create new project for this task, and then make the these interfaces and classes:



<<Java Class>> Config net.ubilife.dsd.computersystem Config()
<<Java Class>> ComputerTest net.ubilife.dsd.computersystem ComputerTest() main(String[]):void
<<Java Interface>> IProgram net.ubilife.dsd.computersystem getName():String execute():void
<<Java Interface>> IUser net.ubilife.dsd.computersystem useDevice():void
<<Java Interface>> IComputingDevice net.ubilife.dsd.computersystem run(IProgram):void output(String):void
<<Java Interface>> IOperatingSystem net.ubilife.dsd.computersystem runProgram(IProgram):void
<<Java Interface>> IPrinter net.ubilife.dsd.computersystem print(String):void
-program 0..1
-device 0..1
-os 0..1
-printer 0..1
<<Java Class>> SpringToolSuite net.ubilife.dsd.computersystem SpringToolSuite() getName():String execute():void
<<Java Class>> Professor net.ubilife.dsd.computersystem Professor() useDevice():void
<<Java Class>> PersonalComputer net.ubilife.dsd.computersystem PersonalComputer() run(IProgram):void output(String):void
<<Java Class>> Linux net.ubilife.dsd.computersystem Linux() runProgram(IProgram):void
<<Java Class>> ConsolePrinter net.ubilife.dsd.computersystem ConsolePrinter() print(String):void

(continues on next page)

Use autowiring to handle dependencies:

- Professor depends on IProgram (SpringToolSuite)
- Professor depends on IComputingDevice (PersonalComputer)
- PersonalComputer depends on IOperatingSystem (Linux)
- PersonalComputer depends on IPrinter (ConsolePrinter)

IMPORTANT: use interfaces in dependencies! (*loosely coupled*)

Method call order:

- Professor's useDevice() calls IComputingDevice's run() and output().
- run() calls IOperatingSystem's runProgram()
- runProgram() calls IProgram's getName() and execute()
- output() calls IPrinter's print()

In ComputerTest, get Professor bean and call its useDevice() method.

In all methods, print out class name, method name and relevant string.

Example:

```
Professor.useDevice()> Running a program...
PersonalComputer.run()> Running SpringToolSuite
Linux.runProgram()> Executing SpringToolSuite
SpringToolSuite.execute()> Wiring beans with STS!
Professor.useDevice()> Testing output...
PersonalComputer.output()> Outputting...
ConsolePrinter.print()> Hello, Spring!
```

TASK 3: Windows OS (2p)

Add a Windows component that implements IOperatingSystem to your Task 2 project. Try to run the project. You should see something like (excerpt):

```
Exception in thread "main" org.springframework.beans.factory.BeanCreationException: Error creating bean
with name 'personalComputer': Injection of autowired dependencies failed; nested exception is
org.springframework.beans.factory.BeanCreationException: Could not autowire field: private
net.ubilife.dsd.computersystem.IOperatingSystem net.ubilife.dsd.computersystem.PersonalComputer.os;
nested exception is org.springframework.beans.factory.NoUniqueBeanDefinitionException: No qualifying
bean of type [net.ubilife.dsd.computersystem.IOperatingSystem] is defined: expected single matching
bean but found 2: linux,windows
...
```

NoUniqueBeanDefinitionException means that you have more than one suitable bean for autowiring. The problem is in PersonalComputer:

```
        @Autowired
        private IOperatingSystem os;
```

Spring doesn't know which one to autowire: Linux or Windows!

Here's an example of how to fix this problem with the @Qualifier annotation:

```
public interface IAnimal { }

@Component
public class Dog implements IAnimal { ... }

@Component
public class Cat implements IAnimal { ... }
```

```
@Component
public class Veterinarian {
        @Autowired
        @Qualifier("dog")              // inside Qualifier, use a class' name with first letter lowercase
        private IAnimal animal;
        ...
}
```

Now fix your project to make sure that Professor can also use Windows OS!