LAB: EXPERIMENTING WITH SPECIALIZATION

Getting Ready: Before going any further you should:

1. Make a directory on your `N:` drive for this lab.
2. Setup your development environment.
3. Download Clock.class and icon.gif, and put them in your working directory.

The documentation for the `Clock` class is available at javadocs/Clock.html.

Part I: In this part of the lab you will use an existing class.

1. Write a driver that constructs two clocks, one for Harrisonburg and one for Paris.
2. Compile and execute your driver.

Part II: In this part of the lab you will specialize an existing class.

1. Create a file named `AlarmClock.java` that contains the following:

```
/**
 * A GUI window that contains an alarm clock
 * with a digital display
 *
 * @author
 * @version 1.0
 */
public class AlarmClock extends Clock
{
    private boolean    on;
    private int        hour, minute, second;
    private String     ampm;



    /**
     * Default Constructor
     */
    public AlarmClock()
    {

    }


    /**
     * Explicit Value Constructor
     *
     * Constructs a clock for a particular city, setting
     * the time zone appropriately
     *
```

```
     * @param city   The city
     */
    public AlarmClock(String city)
    {

    }


}
```

2. Modify the two constructors in the `AlarmClock` class so that they call the appropriate constructors in the `Clock` class.

3. Modify your driver so that the clock for Harrisonburg is actually an `AlarmClock` object.

4. Compile and execute your driver.

5. Add the following to your `AlarmClock.java` class:

```
    /**
     * Set the alarm
     *
     * @param hour     The hour of the alarm
     * @param minute   The minute of the alarm
     * @param second   The second of the alarm
     * @param ampm     "AM" for before noon, "PM" for noon and after
     */
    public void setAlarm(int hour, int minute, int second, String ampm)
    {

    }


    /**
     * Turn the alarm off
     */
    public void turnAlarmOff()
    {
         on = false;
    }



    /**
     * Turn the alarm on
     */
    public void turnAlarmOn()
    {
         on = true;
    }
```

and complete the `setAlarm()` method (i.e., set the attributes appropriately).

6. Add the following to your `AlarmClock.java` class:

```
    /**
     * Update the time displayed on the clock
```

```
     */
    public void updateTime()
    {
         int         hourNow, minuteNow, secondNow;
         String      ampmNow;

         // Call the parent's version of updateTime()


         // If the alarm is on, get the current hour, minute
         // second, and ampm and check to see if the alarm
         // should sound now
    }
```

and complete the `udpateTime()` method.

7. Modify your driver so that it sets the alarm (for a time about a minute in the future) and turns the alarm on.

8. Compile and execute your driver.

9. What happens (or doesn't happen) if you comment out the call to the parent's `updateTime()` method?

Part III: In this part of the lab you will think about specialization in Java, and what kinds of errors you might encounter when you compile classes that you develop.

1. Create a file named `Tester1.java` that contains the following:

```
/**
 * A Driver for testing the Clock and AlarmClock classes
 */
public class Tester1
{
    /**
     * The enty point of the application
     *
     * @param args  The command line arguments
     */
    public static void main(String[] args)
    {
         AlarmClock        home;
         Clock             paris;

         home = new AlarmClock();
         paris = new Clock("Paris");

         setup(home);
         setup(paris);
    }


    /**
     * Setup a Clock
     *
     * @param clock   The clock to setup
     */
    private static void setup(Clock clock)
    {
         clock.reverseColors();
```

```
    }

}
```

2. The `setup()` method has a `Clock` as a formal parameter but is actually passed an `AlarmClock`. Will this class compile? If so, why? If not, why not?

3. Create a file named `Tester2.java` that contains the following:

```
/**
 * A Driver for testing the Clock and AlarmClock classes
 */
public class Tester2
{
    /**
     * The enty point of the application
     *
     * @param args  The command line arguments
     */
    public static void main(String[] args)
    {
         AlarmClock        home;
         Clock             paris;

         home = new AlarmClock();
         paris = new Clock("Paris");

         setup(home);
         setup(paris);
    }


    /**
     * Setup a clock
     *
     * @param clock   The clock to setup
     */
    private static void setup(Clock clock)
    {
         clock.reverseColors();
         clock.turnAlarmOn();
    }

}
```

4. Will this class compile? If so, why? If not, why not?

5. Create a file named `Tester3.java` that contains the following:

```
/**
 * A Driver for testing the Clock and AlarmClock classes
 */
public class Tester3
{
    /**
     * The enty point of the application
     *
     * @param args  The command line arguments
     */
    public static void main(String[] args)
    {
         AlarmClock        home;
```

```
        Clock            paris;

        home = new AlarmClock();
        paris = new Clock("Paris");

        setup(home);
        setup(paris);
    }


    /**
     * Setup a clock
     *
     * @param clock   The clock to setup
     */
    private static void setup(AlarmClock clock)
    {
        clock.reverseColors();
        clock.turnAlarmOn();
    }

}
```

6. Will this class compile? If so, why? If not, why not?

7. Create a file named `Tester4.java` that contains the following:

```
/**
 * A Driver for testing the Clock and AlarmClock classes
 */
public class Tester4
{
    /**
     * The enty point of the application
     *
     * @param args  The command line arguments
     */
    public static void main(String[] args)
    {
        Clock        home, paris;

        home  = createClock("Harrisonburg");
        paris = createClock("Paris");

    }


    /**
     * Create and setup a Clock
     *
     * @param clock   The clock to setup
     */
    private static AlarmClock createClock(String city)
    {
        AlarmClock      temp;

        temp = new AlarmClock(city);
        temp.reverseColors();

        return temp;
    }

}
```

8. The `createClock()` method creates and returns `AlarmClock` objects that `main` assigns to `Clock` objects. Will this class compile? If so, why? If not, why not?

9. Create a file named `Tester5.java` that contains the following:

```
/**
 * A Driver for testing the Clock and AlarmClock classes
 */
public class Tester5
{
    /**
     * The enty point of the application
     *
     * @param args  The command line arguments
     */
    public static void main(String[] args)
    {
         Clock        home, paris;

         home  = createClock("Harrisonburg");
         paris = createClock("Paris");

         home.setAlarm(1, 39, 45, "PM");
         home.turnAlarmOn();
    }



    /**
     * Create and setup a Clock
     *
     * @param clock   The clock to setup
     */
    private static AlarmClock createClock(String city)
    {
         AlarmClock      temp;

         temp = new AlarmClock(city);
         temp.reverseColors();

         return temp;
    }

}
```

10. Will this class compile? If so, why? If not, why not?