

Assignment 4 – Spring 2020

In this assignment we utilize Java's inheritance capabilities and build a hierarchy of classes using Interfaces and Abstract classes. We will, first, rely on the Java SDK for available and reusable tasks before designing new data types. The aim is to insure maximum efficiency and reusability with fewer lines of code.

- ✓ *Please do your own work, sharing and/or copying code and/or solution ideas with/from others will result in a grade of 0 and disciplinary actions for all involved parties. If you run into any problems and have done your best to solve them, please see me before/after class or e-mail me.*
- ✓ *If your submission is late, you will incur a 20% deduction of your score for each late day.*

Problem Description:

Implement the classes shown in the class diagram below (Figure 1). The notation used in this figure is detailed in Table 1. Please adhere to the names shown in the diagram. Some of the methods of the classes have been omitted as it is part of the assignment to decide on the best place to implement and/override certain methods. Use the provided test code and the sample output shown in Figure 2 to guide you through the class' implementations.

Class *Main*:

- ✓ Provided test code. Please do not modify or submit this class file.

Class *Screen*

- ✓ An abstract class with two abstract method. Subclasses must implement these two methods.
- ✓ Abstract *getType()*, each subclass returns its name (i.e. *ComputerMonitor*, *CRT*, *LED*, and *Smart*)
- ✓ Abstract *equals()*, each subclass returns a true or false as described next.
- ✓ The non-default constructor initializes the class' private fields.

Class *ComputerMonitor*:

- ✓ Abstract classes which inherits from class *Screen*.
- ✓ The class does NOT define *getType()* or *equals()*.

Class *CRT*:

- ✓ A class which inherits from class *ComputerMonitor*
- ✓ The non-default constructor initializes the class' (and base class') private fields
- ✓ *getType()* return "CRT"
- ✓ *equals()* returns *true* if the two objects have the same *ID*. If the *ID* values are not equal, return *false*.

Class *LED*:

- ✓ A class which inherits from class *ComputerMonitor*
- ✓ The non-default constructor initializes the class' (and base class') private fields
- ✓ *getType()* return "LED"
- ✓ *equals()* returns true if the two objects have the same *salePrice*.

Class *SmartTV*:

- ✓ A class which inherits from class *Screen*
- ✓ The non-default constructor initializes the class' (and base class') private fields
- ✓ *getType()* return "Smart"
- ✓ *equals()* returns the result of comparing the value of *model*. Use the *equals()* of class String. For example, "Hello".*equals*("Hello");

Class *Inventory*:

- ✓ *listInventory* points to an instance of type *ArrayList*. The list will hold objects of type *Screen*
- ✓ *addToInventory()* checks if a similar instance is already stored in the list *listInventory*. If there is one, throw an exception of type *Exception* and message "The inventory contains a similar *Screen* object". If there is no duplicate object, add the object to the list and return **true**. **YOU MUST USE the *contains()* method of *listInventory* which requires that each subclass to implement the method *equals()*. Do not write your own search code.**
- ✓ *listComputerMonitor()*, returns a new list containing instances of supertype *ComputerMonitor*. Hint: the following statement checks if the reference variable *someRefVar* points to an instance of type *MyClass*









```
if (someRefVar instanceof MyClass)
    System.out.println("Reference to type MyClass");
```

- ✓ *listSmartTV()*, returns a new list containing instances of supertype *SmartTV*.
- ✓ *printInventory()*, prints a table containing the information from the list. See Figure 2 for the output's format. Note that the printed table's header is simply a fixed string. However, for the table's rows cannot be fixed and you should use Java's *printf* method.

Grades:

Item	Points
Class <i>Screen</i>	10
Class <i>ComputerClass</i>	10
Class <i>CRT</i>	10
Class <i>LED</i>	10
Class <i>SmartTV</i>	10
Class <i>InventoryList</i>	
<i>add</i>	10
<i>listComputerClass</i> and <i>listSmartTV</i>	10
<i>printInventory</i>	10
<i>equals</i>	10
Correct output	10
	100

Class diagram legend:

Symbol	Description
	A private member (i.e. variable or method)
	A private final member (i.e. variable)
	A public field (i.e. variable or method)
	A public abstract member (i.e. variable or method)
	A public constructor
	A static public member
	An interface
	A public class

Symbol	Description
	A public abstract class
	A hollowed arrow indicates inheritance
	An open-ended arrow indicates composition
	A dotted line and hollowed arrow indicate class implementation

Table 1: UML Diagram Legend

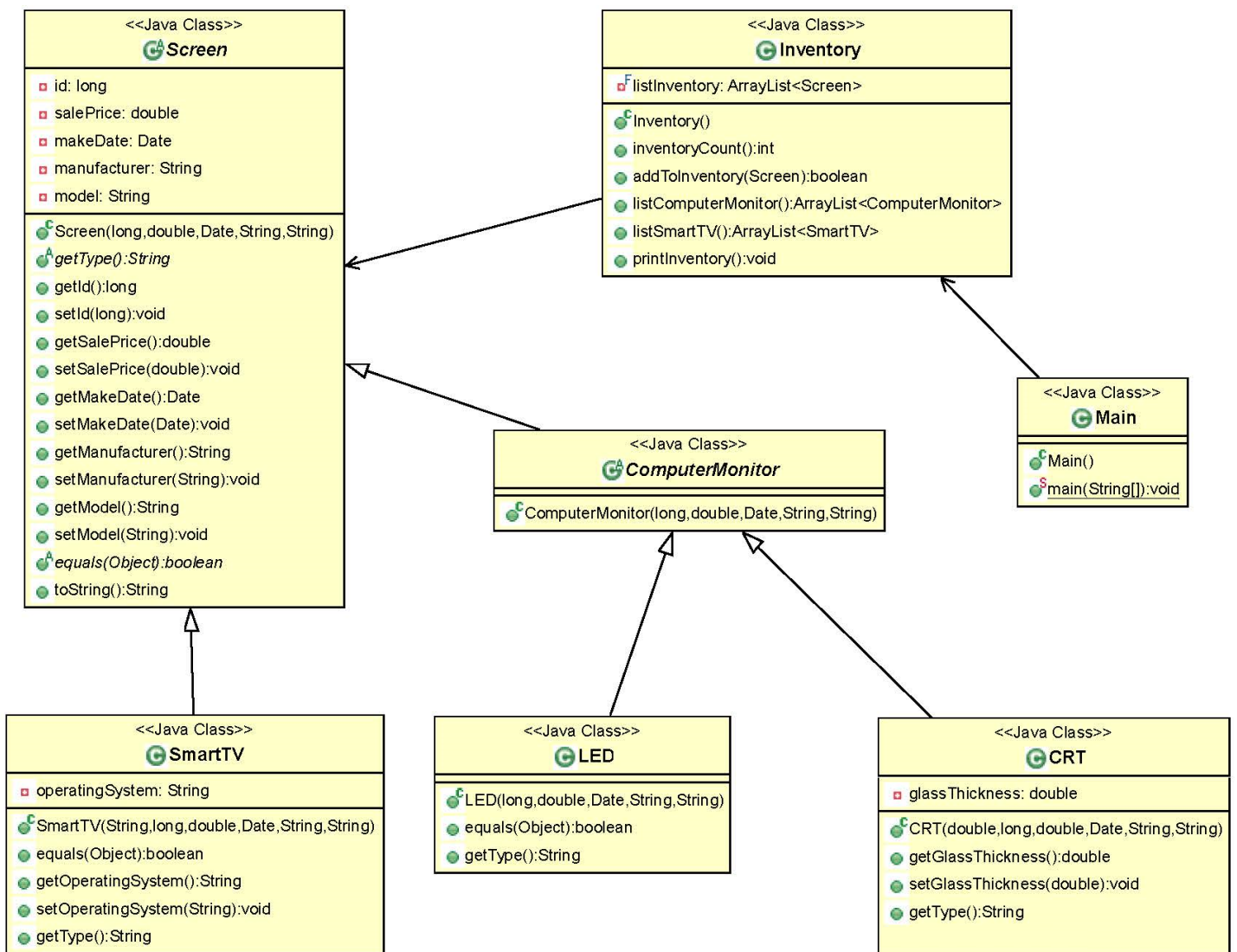


Figure 1: Class diagram

Duplicate inventory, skipping: 1838900843,212.30,08/25/2012,Alienware,VN3854B,,Android TV
 Duplicate inventory, skipping: 1501815258,968.22,02/01/2016,LG,IS5100TV,,
 Duplicate inventory, skipping: 198395827,2453.27,04/01/2001,BenQ,CV2584IK,,Android TV
 There are 35 Screens in the inventory
 There are 24 Computer Monitors
 There are 11 Smart TVs

	Type	ID	Price	Make Date	Manufacturer	Model	Glass	OS
1	Smart	1994711256	\$212.30	08/25/2012	Alienware	VN3854B		Android TV
2	CRT	1085793438	\$1,668.38	12/25/2016	LG	VZ2098TV	10.50	
3	LEF	1085793438	\$968.22	02/01/2016	LG	IS5100TV		
4	Smart	1908038773	\$2,453.27	04/01/2001	BenQ	CV2584IK		Android TV
5	CRT	1238567597	\$1,963.18	11/11/2003	AOC	MF2873VZ	10.01	
6	LEF	1891088168	\$186.23	04/25/2007	HP	VX0470TQ		
7	LEF	1962289898	\$2,107.77	03/07/2003	BenQ	DF3765HF		
8	CRT	109690874	\$648.75	12/06/2010	Acer	WQ2398HK	11.44	
9	Smart	1937180916	\$211.45	09/25/2016	Apple	NC5583CT		webOS
10	CRT	1217356715	\$3,648.56	01/01/2003	Acer	ES0642RH	3.22	
11	Smart	1548026999	\$1,466.03	01/01/2002	Planar	AW6360		Tizen
12	LEF	112559202	\$1,493.22	10/02/2017	LG	WZ6895BF		
13	Smart	986703771	\$476.81	06/15/2004	Samsung	IV1136UX		SmartCast
14	LEF	1912458314	\$1,260.86	12/23/2020	Lenovo	UO6143L		
15	Smart	829514163	\$2,447.60	01/21/2006	NEC	XF5230JG		webOS
16	LEF	997935914	\$628.56	02/05/2004	HP	FD1381G		
17	LEF	776199782	\$2,800.41	07/26/2013	Alienware	NW7791CP		
18	Smart	392355169	\$614.63	04/08/2016	Alienware	VG7294RN		Tizen
19	LEF	250750113	\$1,021.36	09/01/2015	Lenovo	HQ6357FV		
20	CRT	873893446	\$2,010.66	02/07/2011	Acer	LY0648S	11.80	
21	LEF	759064919	\$607.18	01/28/2017	Dell	SM1976LE		
22	Smart	396879383	\$187.21	04/05/2012	ViewSonic	LZ6232FH		SmartCast
23	LEF	2058224287	\$2,119.05	02/20/2012	BenQ	YT2305RB		
24	LEF	1330930391	\$928.66	05/16/2002	NEC	PZ6549MK		
25	Smart	1400649635	\$4,605.87	04/19/2005	BenQ	WZ2150IY		Tizen
26	LEF	757248942	\$2,012.19	09/23/2006	NEC	YE6910FO		
27	LEF	1084319473	\$74.15	01/02/2010	Samsung	Y716SE		
28	Smart	491135186	\$2,182.19	05/07/2015	Samsung	YO2277AJ		SmartCast
29	LEF	343765014	\$1,340.68	05/08/2000	Planar	XT7182VE		
30	CRT	1071044513	\$174.65	05/09/2002	Acer	Y1XJY	1.07	
31	LEF	630781837	\$988.07	03/02/2002	HP	DR0989TO		
32	LEF	232209447	\$405.67	06/14/2014	Asus	JJ4083WI		
33	Smart	1604611287	\$1,142.23	11/10/2003	Alienware	LC850G		Android TV
34	LEF	1030750276	\$1,473.19	01/06/2015	NEC	GJ6487W		
35	LEF	350341850	\$153.33	07/04/2003	Lenovo	E9357CW		

Figure 2: Test Code's Output