|  |  |
| --- | --- |
| **Assignment 7 – SOLID Principles** | |
|  | **Identify which SOLID principle has been applied in the code snippets below.** | |
|  | **Part 1** | |
|  | interface GestureHandler { | |
|  |  | |
|  | void click(Element element); | |
|  |  | |
|  | } | |
|  |  | |
|  |  | |
|  |  | |
|  | interface WebGestureHandler extends GestureHandler {} | |
|  |  | |
|  |  | |
|  |  | |
|  | interface MobileGestureHandler extends GestureHandler { | |
|  |  | |
|  | void doubleTap(Element element); | |
|  |  | |
|  | } | |
|  |  | |
|  | ***Answer- Single Responsibility Principle*** | |
|  |  | |
|  | **Part 2** | |
|  | public class Message { | |
|  |  | |
|  | String text; | |
|  |  | |
|  | } | |
|  |  | |
|  |  | |
|  |  | |
|  | public class PriorityMessage extends Message { | |
|  |  | |
|  | private int priority; | |
|  |  | |
|  | } | |
|  |  | |
|  |  | |
|  |  | |
|  | public class MessageHandler { | |
|  |  | |
|  | public void handle(Message message) { | |
|  |  | |
|  | // Handle the message. | |
|  |  | |
|  | } | |
|  |  | |
|  | } | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  | public class PriorityMessageQueue { | |
|  |  | |
|  | public void receive(PriorityMessage message) { | |
|  |  | |
|  | new MessageHandler().handle(message); | |
|  |  | |
|  | } | |
|  |  | |
|  | }  ***Answer- Single Responsibility Principle*** | |
|  |  | |
|  | **Part 3** | |
|  | public class BankAccount { | |
|  |  | |
|  | private final String number; | |
|  |  | |
|  |  | |
|  |  | |
|  | public BankAccount(final String number) { | |
|  |  | |
|  | this.number = number; | |
|  |  | |
|  | } | |
|  |  | |
|  | } | |
|  |  | |
|  | public class SavingsBankAccount extends BankAccount {} | |
|  |  | |
|  | public class DepositBankAccount extends BankAccount {} | |
|  | ***Answer-Open-Closed Principle*** | |
|  |  | |
|  | **Part 4** | |
|  | public interface TaxCalculator { | |
|  |  | |
|  | void applyTaxes(Order order); | |
|  |  | |
|  | } | |
|  |  | |
|  |  | |
|  |  | |
|  | public class OrderProcessor { | |
|  |  | |
|  | private final TaxCalculator taxCalculator; | |
|  |  | |
|  |  | |
|  |  | |
|  | public OrderProcessor(final TaxCalculator taxCalculator) { | |
|  |  | |
|  | this.taxCalculator = taxCalculator; | |
|  |  | |
|  | } | |
|  |  | |
|  | } | |
|  | ***Answer-Open-Closed Principle*** | |
|  |  | |
|  | **Part 5** | |
|  | public class LedgerEntry {} | |
|  |  | |
|  |  | |
|  |  | |
|  | public class Ledger { | |
|  |  | |
|  | private Set<LedgerEntry> entries; | |
|  |  | |
|  |  | |
|  |  | |
|  | public void addEntry(final LedgerEntry entry) { | |
|  |  | |
|  | if (entry != null) { | |
|  |  | |
|  | if (entries == null) { | |
|  |  | |
|  | entries = new HashSet<>(); | |
|  |  | |
|  | } | |
|  |  | |
|  |  | |
|  |  | |
|  | entries.add(entry); | |
|  |  | |
|  | } | |
|  |  | |
|  | } | |
|  |  | |
|  |  | |
|  |  | |
|  | public void getEntries() { | |
|  |  | |
|  | return entries == null ? null | |
|  |  | |
|  | : new HashSet<>(entries); | |
|  |  | |
|  | } | |
|  |  | |
|  | } | |
|  |  | |
|  | ***Answer-Liskov’s Substitution Rule*** | |