Nachdenkzettel: Interfaces und Software-Architektur

Lukas, Lara, Jan, Karin

Vorname, Name, Matrikelnummer

1. Spezifizieren Sie das Interface „Stecker“ für diese Implementation.



copyright Aunkrig, [CC-BY-SA-4.0](https://commons.wikimedia.org/wiki/Category:CC-BY-SA-4.0)   
Maße, Vorgaben für die Stifte (Länge, Größe. Abstand, Anzahl), Erdung, Material, Isolierung

1. Ist das a) eine korrekte Ableitung von der obigen Implementation?  
   Nein, keine Ableitung, denn das bestehende Produkt ist inkompatibel

copyright hic et nunc, Cc-by-sa-3.0-migrated

1. Und das? Autor: somnusde, wikimedia-commons, PD  
   Ist auch inkompatibel, es fehlen Teile!   
   Gefahr: passt physisch eigentlich rein

1. Wie sieht es mit 220 V aus? Interface oder Implementation? Und das Material des Schukosteckers?  
   Ein 220V-Stecker entspricht der DIN-Norm 🡪 Implementation  
   Das Material ist verpflichtend Kunststoff und Metall
2. Wieviel Spaß hätten wir ohne die DIN Norm für Schukostecker oder Eurostecker?  
   Gar keinen, sonst hätte jeder seinen eigenen Standard für Stecker 🡪 Herstellerbindung, Inkompatibilität
3. Was gehört alles zum „Interface einer Klasse“ in Java? (Anders formuliert für UX-Leute: wenn ich von jemandem eine Klasse in meinem Code benutze: was ärgert mich, wenn es geändert wird?)

- Methoden (nur Kopf)

- Konstanten

- kein Konstruktor!

- Default-Implementierung möglich!

-

1. „Class B implements X“. Jetzt fügen Sie eine neue Methode in Interface X ein. Was passiert?  
   Compile error 🡪 Klasse B erfüllt nicht mehr die Vorgaben des Interface und muss angepasst werden
2. Zwei Interfaces sind nicht voneinander abgeleitet, haben aber zufällig die gleiche Methode. Können Sie Implementationen dieser Interfaces polymorph behandeln?

|  |  |  |
| --- | --- | --- |
| Interface X { | Interface Y { | class B implements Y { ...} |
| public void foo(); | public void foo(); |  |
| } | } |  |

X x = new B(); //Geht nicht, B ist vom Typ Y und hat trotz der gleich benannte Methode nichts mit X zu tun

x.foo(); ??

1. Ihr code enthält folgendes statement: X xvar = new X();

Was ist daran problematisch, wenn Sie eine Applikation für verschiedene Branchen/Kunden/Fälle bauen?   
Man muss den Core der Applikation pro Kunde ändern, besser: Factory 🡪 kundenspezifische Implementationen können von außen „eingesteckt“ werden

1. Von ArrayList ableiten oder eigene Klasse „Catalog“ oder ähnlich bauen und ArrayList<> verwenden? Sprich: soll man von Java Basisklassen ableiten? Beispiele: Vegetable, VegetableCatalog Task, TaskList, GameObject, GameObjectList etc.  
     
   Niemals Java base class extenden! Lieber Klassen verwenden