7. Zusammenfassung (1)

Typisierung in OO-Sprachen

- Subtyping vs. Subclassing
- Untertypen für Typkonstrukte
- Funktionsuntertypen und Überschreiben
- Generik

Einsatz von Vererbung konzeptioneller Entwurf:

- Abstraktion
- Spezialisierung
- Rollen, Eigenschaften
- Schnittstellenabstraktion

Programmentwicklung:

- Inkrementelle Weiterentwicklung
- Komposition statt Vererbung
- Varianten in mehreren Dimensionen
- Eingebettete Agenten

Entwurfsmuster zur Entkopplung von Software

- Abstract Factory
- Factory Method
- Bridge
- Observer
- Strategy

Bibliotheken

- Programmbausteine am Beispiel Ausnahmen
- Kopplung, Kohärenz
- Entkoppeln durch Interfaces, Funktoren
- Programmgerüste am Beispiel AWT

Zusammenfassung (2)

Entwurfsfehler

Missbrauch von Vererbung:

- Verkannter Schauspieler
- Januskopf
- Ungeplanter Anbau

AntiPatterns

- The Blob
- Poltergeister
- Funktionale Zerlegung
- Spaghetti Code

Üble Gerüche im Code

- Symptome für Refaktorisieren, Katalog

Unangenehme OOP-Überraschungen

- Ursache: meist Überschreiben

OO Sprachkonzepte und ihr Einsatz

- Klassen - Objekte vs. Typen - Werte
- Attribute, Methoden, Konstruktoren
- Zugriffsrechte
- Schnittstelle, Implementierung
- Vererbung, Subtyping
- Polymorphie, dyn. Methodenbindung

Jenseits von Java

- Konkatenation - Delegation
- Multiple Inheritance
- Selektives Erben
- Mixin Inheritance
- Varianten der Methodenbindung
- Prototypen statt Klassen

Verständnisfragen zur Objektorientierten Programmierung (1)

1. Erklären Sie, dass Subclassing und Subtyping unterschiedliche Relationen sind.
2. Erklären Sie Subtyping für Records, Funktionen, Variable, Arrays und Objekte.
3. Leiten Sie die Typregel für das Überschreiben aus dem Subtyping ab.
4. Vergleichen Sie Generik in C++ und Java.
5. Wie löst Generik das Typproblem für „binäre Methoden“?
6. Nennen Sie Paradigmen der Vererbung auf Entwurfsebene.
7. Was bedeuten „ Ersetzbarkeit“ und „Subtyping“ beim Einsatz von Vererbung?
8. Nennen Sie einige Kriterien gegen die Anwendung von Vererbung.
9. Erklären Sie Vererbung zum Zwecke der Abstraktion.
10. Grenzen Sie die Vererbungsparadigmen Abstraktion und Spezialisierung gegeneinander ab.
11. Warum muss man beim Paradigma Abstraktion mehr als 2 Klassen zugleich betrachten?
12. Welches Design Pattern basiert auf dem Abstraktions-Paradigma?
13. Erklären Sie Vererbung zum Zwecke der Spezialisierung.
14. Grenzen Sie die Paradigmen Spezialisierung und inkrementelle Weiterentwicklung gegeneinander ab.

Verständnisfragen zur Objektorientierten Programmierung (2)

15. Welche Rolle spielt Spezialisierung bei Programmgerüsten?
16. Erklären Sie die Vererbungsparadigmen Rollen, Eigenschaften.
17. Erklären Sie das Schema der Delegation an implementierten Rollen.
18. Grenzen Sie die Paradigmen Rollen und Abstraktion gegeneinander ab.
19. Erklären Sie das Vererbungsparadigma Spezifikation.
20. Grenzen Sie das Paradigma Spezifikation gegen Spezialisierung, Abstraktion und Rollen ab.
21. Erklären Sie die Technik inkrementelle Weiterentwicklung.
22. Wie kann man Probleme der inkrementellen Weiterentwicklung vermeiden?
23. Unter welchen Umständen ist Komposition der Vererbung vorzuziehen?
24. Erklären Sie die Technik der Varianten in mehreren Dimensionen.
25. Erklären Sie die Technik der eingebetteten Agenten.
26. Nennen Sie Entwurfsmuster, die speziell der Entkopplung von Software dienen.
27. Erläutern Sie das Entwufsmuster Abstract Factory.
28. Erläutern Sie das Entwurfsmuster Factory Method.

Verständnisfragen zur Objektorientierten Programmierung (3)

29. Erläutern Sie das Entwurfsmuster Bridge.
30. Erläutern Sie das Entwurfsmuster Observer.
31. Erläutern Sie das Entwurfsmuster Strategy.
32. Charakterisieren Sie unabhängige, eigenständige und eng kooperierende Bausteine.
33. Zeigen Sie das Zusammenspiel von Sprache und Bibliothek an Ausnahmen in Java.
34. Erklären Sie die Arten der Kopplung von Bausteinen.
35. Erklären Sie Entkopplung durch Interfaces und durch Funktoren.
36. Erklären Sie den Begriff der Koheränz.
37. Was charakterisiert ein Programmgerüst?
38. Geben Sie Beispiele für komplexes Zusammenwirken von Komponenten des AWT Programmgerüstes.
39. Welche Rolle spielt die Software-Architektur eines Programmgerüstes?
40. Geben Sie Schemata für den Missbrauch von Vererbung an.
41. Charakterisieren Sie einige AntiPatterns.
42. Beschreiben Sie Ziele und Methoden des Refactoring.

Verständnisfragen zur Objektorientierten Programmierung (4)

43. Was bedeuten „üble Code-Gerüche“ im Refactoring? geben Sie Beispiele.
44. Geben Sie Beispiele für versehentliches Überschreiben. Wie könnte man es vermeiden?
45. Geben Sie Empfehlungen für den Zugriff auf Attribute.
46. Stellen Sie Klassen und Objekte sowie Typen und Werte gegenüber.
47. Nennen Sie die 3 Situationen, die in Java zur dynamischen Methodenbindung führen.
48. Erläutern Sie die beiden Varianten der mehrfach indirekten Oberklassen in C++.
49. Erläutern Sie das selektive Erben in Eiffel.
50. Was bedeutet Mixin-Inheritance?
51. Erklären Sie das inner-Prinzip aus Simula und Beta.
52. Was bedeutet Mehrfach-Dispatch?
53. Erklären Sie Prototyp-basierte Vererbung an JavaScript.