Objektorientierte Programmierung WS 2013/2014 - Lösung 7

Prof. Dr. U. Kastens
Institut für Informatik, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn

Lösung zu Aufgabe 1

Einführen und Verwenden eines Prototyp-Objektes für Passwort-Objekte:

```
function Passwort (chars, len) {
  this.chars = chars;
  this.len = len;
}

function Gen() {
  this.show = function() {
  var zaehler = 0;
  var pw = "";
  var pos;
  while (zaehler < this.len) {
    pos = Math.random() * this.chars.length;
    pw = pw + this.chars.charAt(pos);
    zaehler = zaehler + 1;
  }
  document.getElementById("result").innerHTML = pw;
  }
}

Passwort.prototype = new Gen();
```

Lösung zu Aufgabe 2

Verwenden Sie die Scala-Sprachbeschreibung, um folgende Aufgaben zu lösen:

a) Übersetzen und Ausführen:

```
scalac first.scala
scala Willy
```

b) Ein parameterloser Konstruktor, der einen unverheirateten "Max Mustermann" erzeugt:

```
def this() = {
               this("Mustermann", "Max", null)
             }
```

c) Zugriff auf das Mixin durch `with`-Klausel und Verwendung der so geerbten Methoden (siehe `blatt7/first-sol`).

d) `AnyRef` ist die höchste Klasse der Referenztypen und damit Oberklasse alle benutzerdefinierten Klassen. Die Angabe `extends AnyRef` ist daher redundant und kann gestrichen werden. Syntaktisch ist gefordert, dass das `with` einer Oberklassenangabe folgt. Falls die nicht vorhanden ist, kann `extends` auch direkt von einer Trait-Angabe gefolgt werden, d.h.

```
class Person(ln : String, fn : String, s : Person) extends AnyRef with Log
```

wird ersetzt durch

```
class Person(ln : String, fn : String, s : Person) extends Log
```

e) Beim Verwenden der Methode `logConstructor()` des Mixins `Log` um Log-Ausgabe für alle drei Konstruktoren der Klasse `Person` zu erzeugen, ist zu beachten, dass die Delegation des Konstruktors per `this(...)` wie in Java als erstes im Konstruktor-Rumpf geschehen muss. Der im Klassenrumpf integrierte dreistellige Konstruktor kann ebenso um den `logConstructor()`-Aufruf erweitert werden.

Lösung zu Aufgabe 3

a) Case Classes in Scala dienen zur Fallunterscheidung auf der Basis von Typen. Sie erlauben eine Musterschreibweise, mit der bequem auf die Objektvariablen-Werte zugegriffen werden kann.
b) Die vollständige Lösung ist in `blatt7/caseclass-sol`.