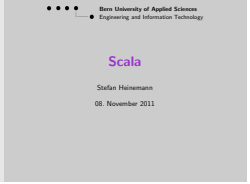


Bern University of Applied Sciences
Engineering and Information Technology

Scala

Stefan Heinemann

08. November 2011

Inhalt
Einleitung
Hallo Welt
Objektorientiert

Inhalt

Einleitung
- objektorientiert und funktional

1. OO geht so weit, dass es keine primitiven gibt
2. Funktional erklären!
3. Elegantere Problemlösung

Einleitung

- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

Einleitung
- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

1. Variablen können nur einen bestimmen Wert annehmen

Einleitung

- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

Einleitung
- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

Einleitung

- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

Einleitung
- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

Einleitung

- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

Einleitung
- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- ...

Einleitung

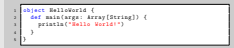
- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

Einleitung
- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

Einleitung

- objektorientiert und funktional
- statisch typisiert
- Bytecode → JVM
- Integration mit Java
- Kompilierbar
- Scriptsprache

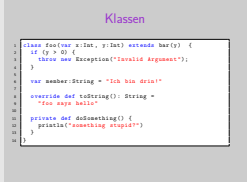
Hallo Welt



1. Obligatorisches Hallo Welt
2. Unvermeidlich, Code zu zeigen
3. object = Singleton-Objekt, keine statischen Sachen
4. Singleton wird beim ersten Aufruf erzeugt
5. Syntax ähnlich wie Java
6. Datentypen werden hinter die Deklaration gestellt
7. Main-Methode wie in Java

Hallo Welt

```
1 object HelloWorld {
2   def main(args: Array[String]) {
3     println("Hello World!")
4   }
5 }
```



1. Da Objektorientierte Programmierung bekannt, nicht interessant das ganze zu erklären, darum ein Beispiel
2. Erklären was wo ist
3. Abstract classes

Klassen

```
1  class foo(var x:Int, y:Int) extends bar(y)  {
2    if (y > 0) {
3      throw new Exception("Invalid Argument");
4    }
5
6    var member:String = "Ich bin drin!"
7
8    override def toString(): String =
9      "foo says hello"
10
11   private def doSomething() {
12     println("something stupid?")
13   }
14 }
```

Traits
- Keine Interfaces
- Keine Mehrfachvererbung

Traits

- Keine Interfaces
- Keine Mehrfachvererbung

Traits
- Keine Interfaces
- Keine Mehrfachvererbung

Traits

- Keine Interfaces
- Keine Mehrfachvererbung

Traits
- Ähnlich wie Interfaces
- Können aber Implementationen enthalten
- Können mehrfach an Klassen weitervererbt werden
- Weniger problematisch als Mehrfachvererbung

Traits

- Ähnlich wie Interfaces
- Können aber Implementationen enthalten
- Können mehrfach an Klassen weitervererbt werden
- Weniger problematisch als Mehrfachvererbung

Traits
- Ähnlich wie Interfaces
- Können aber Implementationen enthalten
- Können mehrfach an Klassen weitervererbt werden
- Weniger problematisch als Mehrfachvererbung

Traits

- Ähnlich wie Interfaces
- Können aber Implementationen enthalten
- Können mehrfach an Klassen weitervererbt werden
- Weniger problematisch als Mehrfachvererbung

Traits
- Ähnlich wie Interfaces
- Können aber Implementationen enthalten
- Können mehrfach an Klassen weitervererbt werden

Traits

- Ähnlich wie Interfaces
- Können aber Implementationen enthalten
- Können mehrfach an Klassen weitervererbt werden
- Weniger problematisch als Mehrfachvererbung

Traits
- Ähnlich wie Interfaces
- Können aber Implementationen enthalten
- Können mehrfach an Klassen weitervererbt werden
- Weniger problematisch als Mehrfachvererbung

Traits

- Ähnlich wie Interfaces
- Können aber Implementationen enthalten
- Können mehrfach an Klassen weitervererbt werden
- Weniger problematisch als Mehrfachvererbung