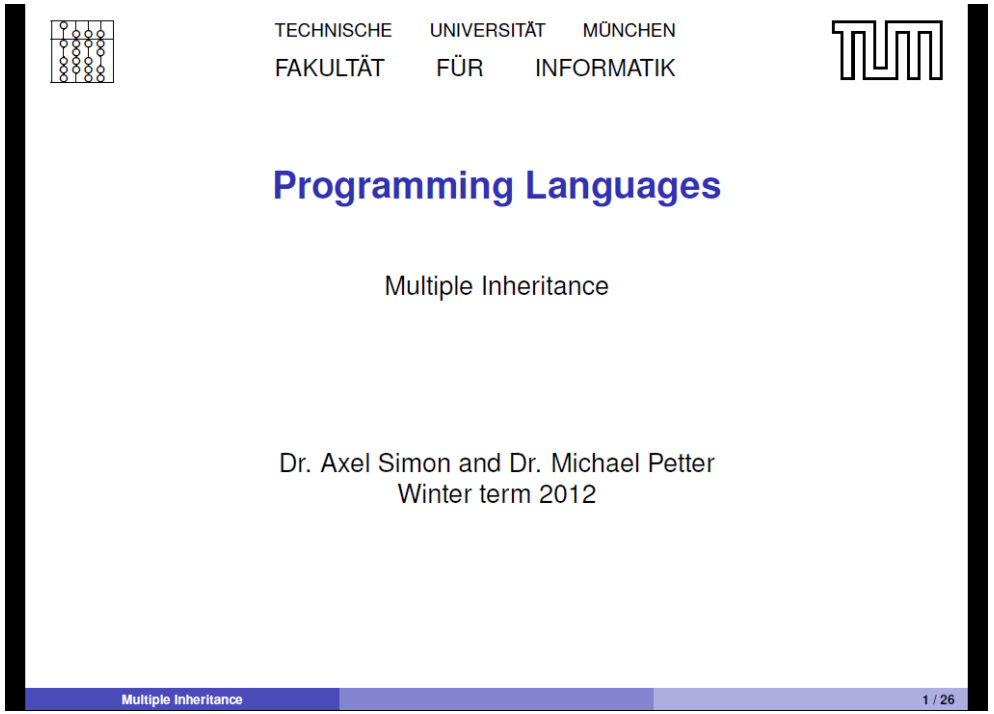
Script generated by TTT

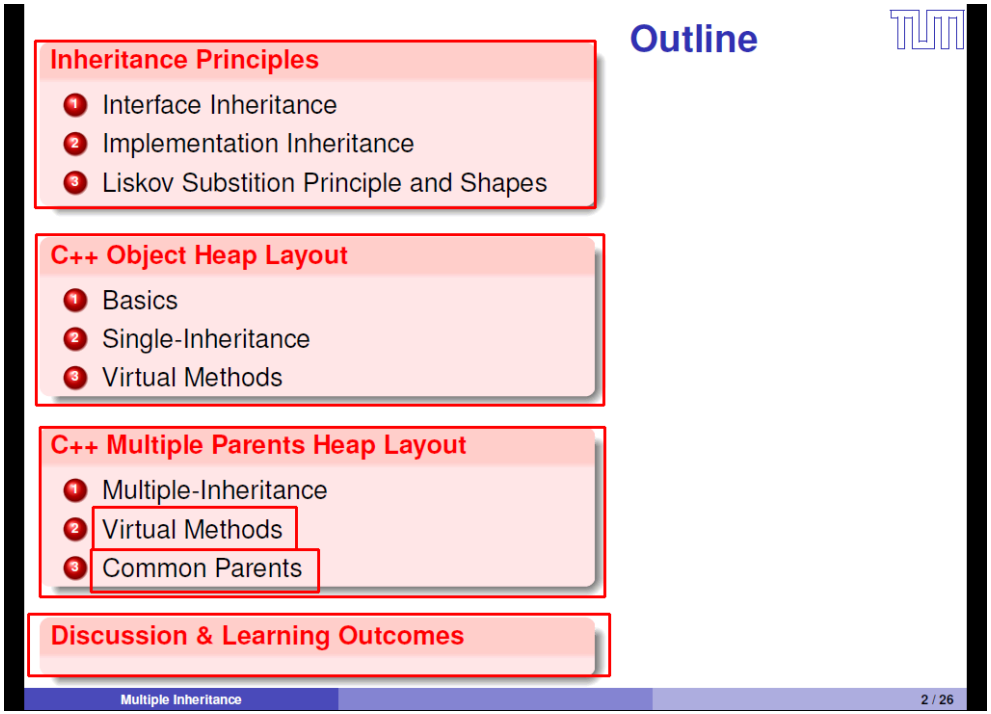
Title: Simon: Programmiersprachen (07.12.2012)

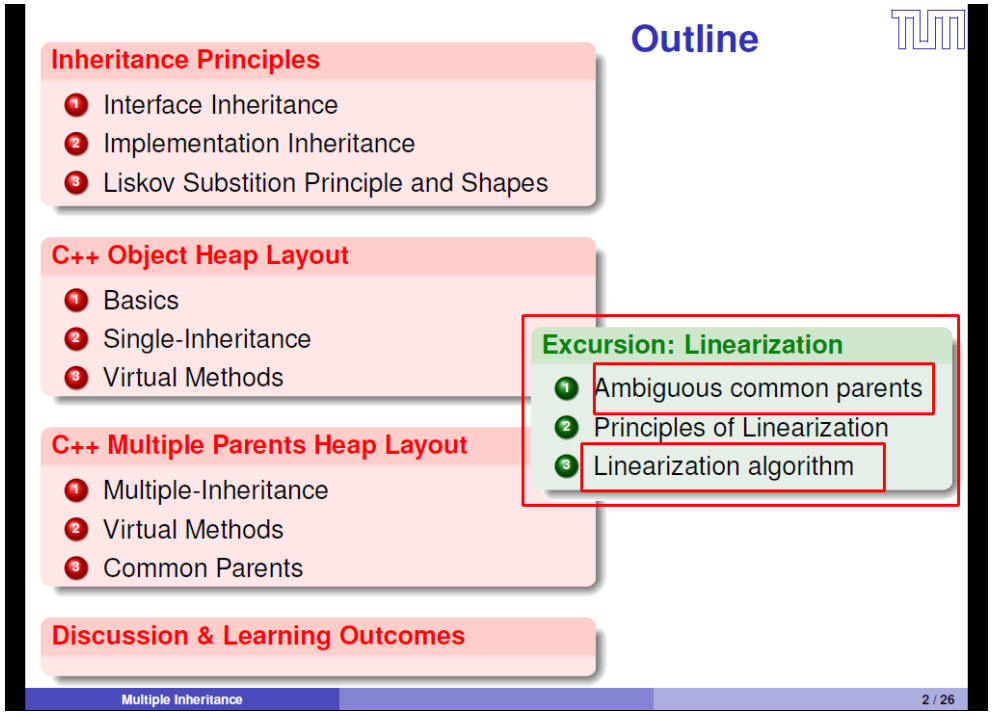
Date: Fri Dec 07 11:05:39 CET 2012

Duration: 20:55 min

Pages: 9







“Wouldn’t it be nice to inherit from several parents?”

Interface vs. Implementation inheritance

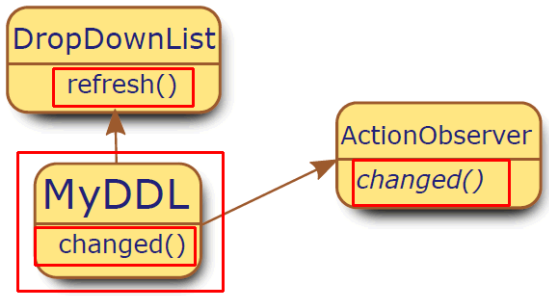
The classic motivation for inheritance is implementation inheritance

- *Code reusage*
- Child specializes parents, replacing particular methods with custom ones
- Parent acts as library of common behaviours
- Implemented in languages like C++ or Lisp

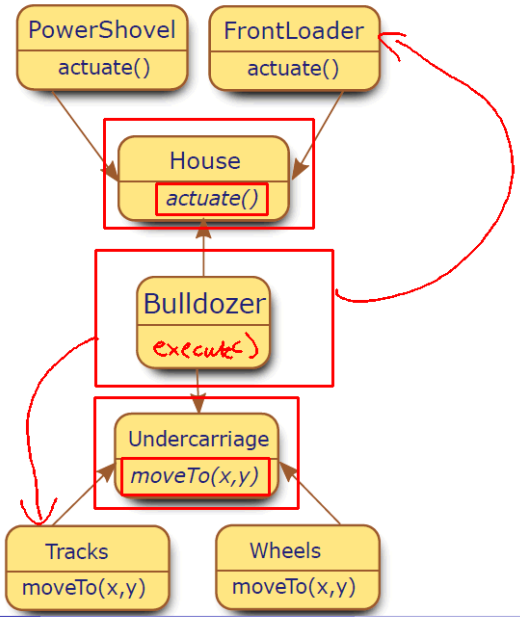
Code sharing in interface inheritance inverts this relation

- *Behaviour contract*
- Child provides methods, with signatures predetermined by the parent
- Parent acts as generic code frame with room for customization
- Implemented in languages like Java or C#

Interface Inheritance



Implementation inheritance



Exkursion: LSP and Square-Rect-Problem

The Liskov Substitution Principle

Functions that use pointers or references to base classes must be able to use objects of derived classes without knowing it.

```
class Rectangle {
  void setWidth (int w){ this.w=w; }
  void setHeight(int h){ this.h=h; }
  void getWidth ()     { return w; }
  void getHeight()     { return h; }
}
class Square extends Rectangle {
  void setWidth (int w){ this.w=w;h=w; }
  void setHeight(int h){ this.h=h;w=h; }
}
```

```
Rectangle r =
        new Square(2);
r.setWidth(3);
r.setHeight(4);
assert r.getHeight()*
       r.getWidth()==12;
```