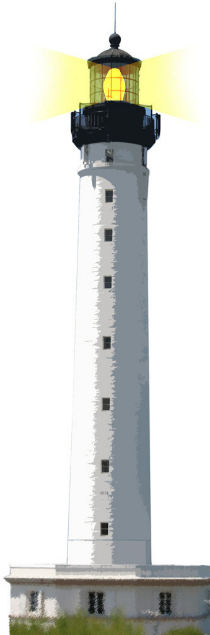
Learning Object-Oriented Programming and Design with TDD

About Types and Lookup

S. Ducasse

http://stephane.ducasse.free.fr



http://www.pharo.org

WXSYY

Remember: Static vs. Dynamic Types

```
A a = new B();
```

- The static type of variable a is A i.e., the statically declared class to which it belongs.
  - The static type never changes.
- The dynamic type of a is B i.e., the class of the object currently bound to a.
  - The dynamic type may change throughout the program.

Setting the stage

```
public interface Acceptable {
    public void accept();
}
```

```
public class Person implements Acceptable {
    public void accept(){
        System.out.println("accept");
    }
    public void agree(){
        System.out.println("agree");
    }
}
```

Normal

```
Person p = new Person();
p.accept();
p.agree();
```

```
accept
agree
```

Normal too

```
Person p = new Person();
Acceptable r = p;
r.accept;
```

```
accept
```

Influence of static type

```
Person p = new Person();
Acceptable r = p;
r.agree(); >>> BREAK!
```

```
java: cannot find symbol
  symbol:   method agree()
  location: variable a of type designCorner.Acceptable
```

- At compile time, the typechecker does not use the dynamic type of the object.
- Within the static type Acceptable there is no method agree().

Same without Interface

```
public class Machine {
   public void accept(){System.out.println("accept");}}
public class Robot extends Machine {
   public void accept(){System.out.println("accept");}
   public void agree(){System.out.println("agree");}}
```

```
Robot r = new Robot();
r.accept();
r.agree();
Machine m = r;
m.accept();
m.agree(); >>> BREAK!
((Machine)r).agree(); >>> BREAK!
```

- A typechecker rejects programs that would execute without problems to make sure that it can find execution that would fail.
- An interface provides a view on the object behavior

Nominal Typing and Typecheking

- Nominal = name
- The type checker look for name of the Type and not inside the API
- The type checker has a static view of the world.

Even if your class implements the exact same interface

- If you do not have a type relationship between your classes, they are incompatible
- This is true for classes and interfaces

What you should know

- Static types are used to identify at compile time which methods to lookup
- Lookup will look for such method at runtime

A course by Stéphane Ducasse
`http://stephane.ducasse.free.fr`

Reusing some parts of the Pharo Mooc by

Damien Cassou, Stéphane Ducasse, Luc Fabresse
`http://mooc.pharo.org`