# Konzepte objektorientierter Programmierung

**Prof. Dr. Peter Müller**

**Werner Dietl**

Software Component Technology

Exercises 9: Ownership

Wintersemester 03/04

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

# Questions?

# Type Scheme Combinator

$$* : TypeScheme \times TypeScheme \rightarrow TypeScheme \cup \{undef\}$$

| | grndS(C) | repS(C) | roS(C) | boolS | intS | nullS |
|---|---|---|---|---|---|---|
| grndS(D) | grndS(C) | repS(C) | roS(C) | boolS | intS | nullS |
| repS(D) | repS(C) | *undef* | roS(C) | boolS | intS | nullS |
| roS(D) | roS(C) | roS(C) | roS(C) | boolS | intS | nullS |

# Object Creation

Compilation Error!

```
readonly T v = new readonly T();
```

Objects have to be created in a specific Universe – this is needed e.g. for dynamic casts from readonly to readwrite types.

$$\frac{\mathrm{TS} \preceq_s [\mathrm{v}], \mathrm{TS} \quad \textit{is grndS or repS}}{\vdash \texttt{v=new TS();}}$$

# Attribute Access on `this`

```
rep T f;
T v = this.f;
this.f = new rep T();
```

Compilation Error!

$$\frac{[\text{this}] * [\text{f}] \preceq_S [\text{v}]}{\vdash \texttt{v=this.f;}}$$

$$\frac{[\text{this}] \; is \; no \; roS, [\text{e}] \preceq_S [\text{this}] * [\text{f}]}{\vdash \texttt{this.f=e;}}$$

# Attribute Access

```
class C { rep T f; … }
C w = …;
rep T v;
v = w.f;
```

Compilation Error!

$$\frac{[\mathrm{f}]\ is\ repS \Rightarrow [\mathrm{w}]\ is\ roS, [\mathrm{w}] * [\mathrm{f}] \preceq_S [\mathrm{v}]}{\vdash \texttt{v=w.f;}}$$

$$\frac{[\mathrm{v}]\ is\ no\ roS, [\mathrm{f}]\ is\ no\ repS, [\mathrm{e}] \preceq_S [\mathrm{v}] * [\mathrm{f}]}{\vdash \texttt{v.f=e;}}$$

# Method Calls

```
class C { T m( T par ); … }
rep C w = …;
rep T v;
v = w.m( new rep T() );
```

$$\frac{\mathrm{m}\ \textit{is not functional},\quad \textit{par}(\mathrm{m})\ \textit{is no repS}, \textit{res}(\mathrm{m})\ \textit{is no repS},\quad [\mathrm{w}]\ \textit{is no roS},\quad [\mathrm{e}] \preceq_s [\mathrm{w}] * \textit{par}(\mathrm{m}), [\mathrm{w}] * \textit{res}(\mathrm{m}) \preceq_s [\mathrm{v}]}{\vdash \texttt{v=w.m(e);}}$$

# Functional Method Calls

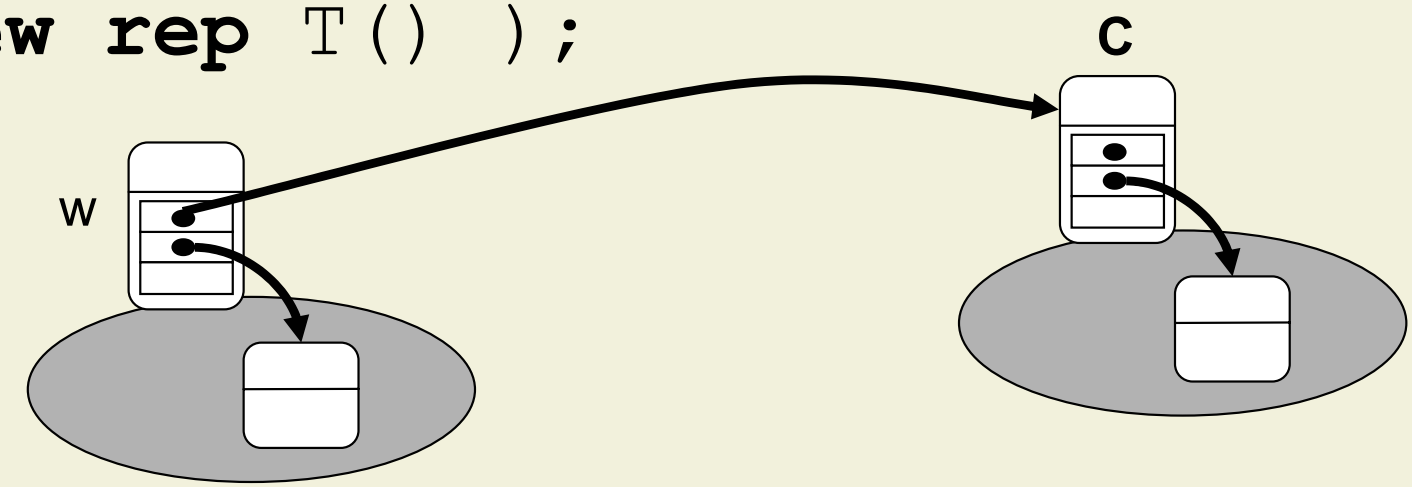
```
class C {
    functional rep T m( readonly T par ); … }
rep C w = …;
rep T v;
v = w.m( new rep T() );
```

Compilation Error!

$$\frac{\mathrm{m}\ \mathit{is}\ \mathit{functional}^5,\quad \mathit{res}(\mathrm{m})\ \mathit{is}\ \mathit{repS} \Rightarrow [\mathrm{w}]\ \mathit{is}\ \mathit{roS},\quad [\mathrm{e}] \preceq_S [\mathrm{w}] * \mathit{par}(\mathrm{m}), [\mathrm{w}] * \mathit{res}(\mathrm{m}) \preceq_S [\mathrm{v}]}{\vdash \texttt{v=w.m(e);}}$$

# Parameters of Functional Methods

```
class C {
    functional void m( rep T par ); … }
readonly C w = …;
w.m( new rep T() );
```




➔ all parameters of functional methods have to be `readonly`

# Method Calls on **this**

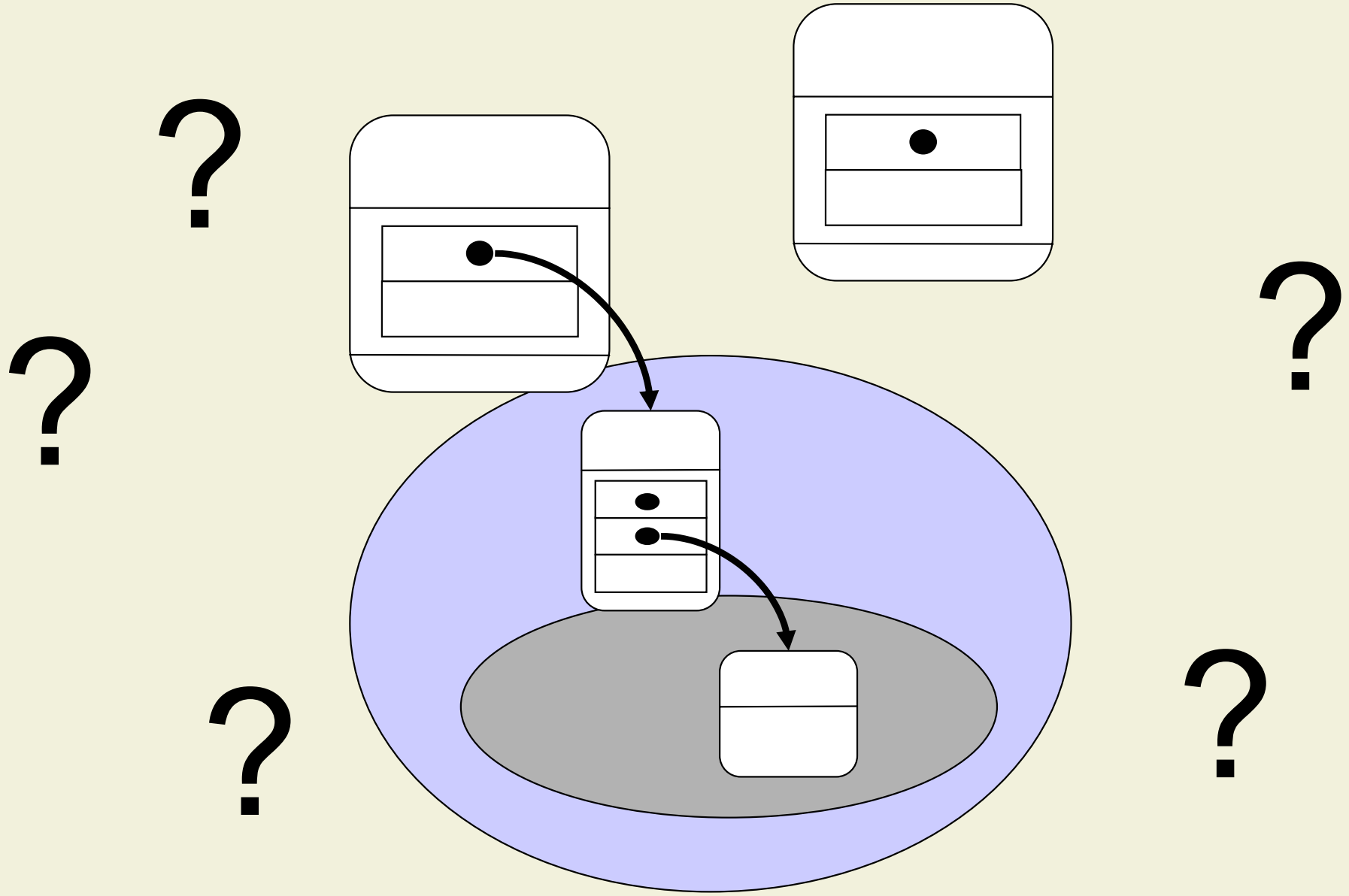
```
class C { rep T m( rep T par ); … }
rep T v;
v = this.m( new rep T() );
```

$$\frac{\mathrm{m}\ \textit{is not functional},\quad [\mathrm{e}] \preceq_s [\mathrm{this}] * \textit{par}(\mathrm{m}), [\mathrm{this}] * \textit{res}(\mathrm{m}) \preceq_s [\mathrm{v}]}{\vdash \texttt{v=this.m(e);}}$$

# Functional Method Calls on **this**

```
class C {
   functional rep T m( readonly T par );
  … }
rep T v;
v = this.m( new rep T() );
```

$$\frac{\mathrm{m}\ \mathit{is}\ \mathit{functional},\quad [\mathrm{e}] \preceq_s [\mathrm{this}] * \mathit{par}(\mathrm{m}), [\mathrm{this}] * \mathit{res}(\mathrm{m}) \preceq_s [\mathrm{v}]}{\vdash \texttt{v=this.m(e);}}$$



# Questions?