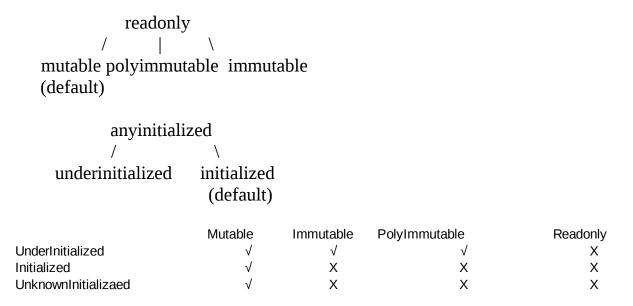
```
cd ::= class C extends D \{ \overline{fd}; kd \overline{md} \}
                                                                                        class
fd := t C f
                                                                                        field
kd := q C (t C g, t C f) \{ super(g); this.f = f; \};
                                                                                    constructor
md := t C m (t C this, t C x) {\overline{t C y}} s; return z 
                                                                                  instance method
                                                                                     expression
e := x | x.f
s := x = e \mid x.f = y \mid x = y.m(z) \mid super(g) \mid x = new t() \mid s;s
                                                                                     statement
t := k q
                                                                                   qualifier type
k ::= initilized | underinitialized | anyinitialized
                                                                             initializatioin qualifier
q ::= readonly | polyimmutable | mutable | immutable
                                                                              immutability qualifier
```

Each class has only one constructor. But it doesn't affect the generality.

## **Type Hierarchy**



√ means allowing assigning fields

**Figure 1 Combination of qulifiers.** Two qualifier hierarchies are orthogonal. If an object is under initialization, its immutability guarantee is not satisfied. So even immutable and polyimmutable objects can also be modifed when under initialization. We don't have readonly objects, so there is no need to initialize readonly objects. Therefore, readonly doesn't have such exception when under initialization.

Subtype relations:

$$k_1 q_1 <: k_2 q_2 <=> k_1 <: k_2 \Lambda q_1 <: q_2$$

#### **Helper Functions**

$$q C f$$
No initialization modifier on field declarations  $fType(f) = q$ 

cBody(c) returns constructor body of c. mBody(m) returns method body of m.

### **Viewpoint Adaptation Rules**

```
_ > mutable = mutable
_ > readonly = readonly
_ > immutable = immutable
q > polyimmutable = q
```

#### **Special Rules**

- Forbid mutable and readonly on fields
- Forbid readonly constructor return type
- In constructor,  $q_{this} = q_{ret}$
- Does not allow initialization modifier on fields and constructor return types

# **Typing Rules**

$$\begin{array}{c}
x \in \Gamma \\
\hline
\Gamma \vdash x : \Gamma(x)
\end{array} \quad (\text{T-VAR}) \\
\Gamma(x) = k_x q_x \quad \text{fType}(f) = q_f \quad q = q_x \triangleright q_f \\
k = \begin{cases}
\text{initialized} & \text{if } k_x = \text{initialized} \\
\text{anyinitialized} & \text{otherwise}
\end{cases} \quad (\text{T-FLD})$$

**Figure 2 Expression typing** 

$$\Gamma \vdash e = t_e \quad t_e <: \Gamma(x)$$

$$\Gamma \vdash x = e$$
(T-VARASS)

$$\begin{split} \Gamma(x) &= k_x \, q_x \quad \Gamma(y) = k_y \, q_y \quad typeof(f) = q_f \\ q_x &= \text{ mutable} \\ &\quad \textbf{V} \ (k_x = \text{ underinitialized } \ \textbf{\Lambda} \ q_x = \text{ immutable}) \ /\!/ \ Didn't \ restrict \ x = "this" \\ &\quad \textbf{V} \ (k_x = \text{ underinitialized } \ \textbf{\Lambda} \ q_x = \text{ polyimmutable}) \ /\!/ \ Didn't \ restrict \ x = "this" \\ q_y &<: q_x \,\rhd\, q_f \\ k_x = \text{ underinitialized } \ \textbf{V} \ k_y = \text{ initialized} \end{split}$$

(T-FLDASS)

$$\Gamma \vdash x.f = y$$

$$\Gamma \vdash s_1; s_2$$
 (T-SEQ)

Figure 3 Statement typing

#### **Well-formdness Rules**

$$\begin{array}{l} mBody(m) = s; return \ z \quad typeof(m) = k_{this} \ q_{this}, \ \overline{k_p} \ \overline{q_p} \rightarrow t_{ret} \\ \Gamma = (this: k_{this} \ q_{this}, \ \overline{p}: \overline{k_p} \ \overline{q_p}, \ \overline{y}: \overline{k_{local}} \ q_{local}) \quad \Gamma \ \vdash \ s \quad \Gamma(z) <: t_{ret} \\ \vdash m \ is \ OK \\ \end{array}$$

Figure 4 Well-formdness typing