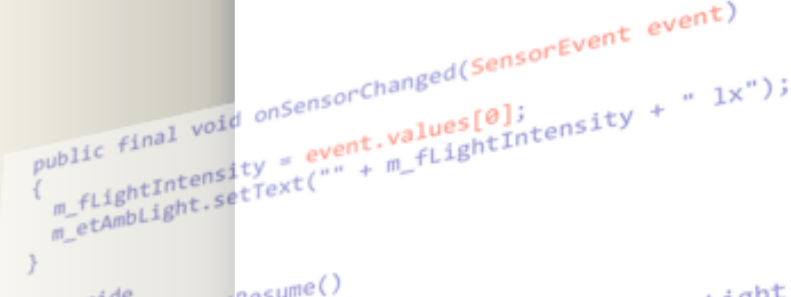
Featherweight Java

Jiří Šebek

b6b36nss



Co je FJ ?

- Featherweight Java (FJ) je podmnožinou Javy.
- Reprezentuje fomalizovaný a minimalizovaný model.
- Programy napsané ve FJ jsou kompilovatelné pomocí JVM.
- Slouží především pro dokazovací účely.
- Lze pomocí ní dokazovat typovou správnost v Javě nebo správnost přepisovacích pravidel (správnost algoritmů apod.)

Formální zápis

Syntax:

$$L ::= \texttt{class C extends C } \{\overline{C}\ \overline{f};\ K\ \overline{M}\}$$

$$K ::= \texttt{C}(\overline{C}\ \overline{f})\{\texttt{super}(\overline{f});\ \texttt{this.}\overline{f}=\overline{f};\}$$

$$M ::= \texttt{C m}(\overline{C}\ \overline{x})\{\ \texttt{return e;}\ \}$$

$$e ::= x \mid e.f \mid e.m(\overline{e}) \mid \texttt{new C}(\overline{e}) \mid (C)e$$

Formální zápis

Subtyping:

$$\texttt{C} <: \texttt{C}$$

$$\frac{\texttt{C} <: \texttt{D} \qquad \texttt{D} <: \texttt{E}}{\texttt{C} <: \texttt{E}}$$

$$\frac{\texttt{class C extends D \{...\}}}{\texttt{C} <: \texttt{D}}$$

Formální zápis

Field lookup:

$$fields(\texttt{Object}) = \bullet$$

$$\frac{\texttt{class C extends D \{}\overline{\texttt{C f}}\texttt{; K \}} \qquad fields(\texttt{D}) = \overline{\texttt{D g}}}{fields(\texttt{C}) = \overline{\texttt{D g}}, \overline{\texttt{C f}}}$$

Method type lookup:

$$\frac{\texttt{class C extends D \{}\overline{\texttt{C f}}\texttt{; K \}} \qquad \texttt{B m(}\overline{\texttt{B x}}\texttt{)\{ return e; \}} \in \overline{\texttt{M}}}{mtype(\texttt{m}, \texttt{C}) = \overline{\texttt{B}} \rightarrow \texttt{B}}$$

$$\frac{\texttt{class C extends D \{}\overline{\texttt{C f}}\texttt{; K \}} \qquad \texttt{m} \notin \overline{\texttt{M}}}{mtype(\texttt{m}, \texttt{C}) = mtype(\texttt{m}, \texttt{D})}$$

Method body lookup:

$$\frac{\texttt{class C extends D \{}\overline{\texttt{C f}}\texttt{; K \}} \qquad \texttt{B m(}\overline{\texttt{B x}}\texttt{)\{ return e; \}} \in \overline{\texttt{M}}}{mbody(\texttt{m}, \texttt{C}) = \overline{\texttt{x}}\texttt{.e}}$$

$$\frac{\texttt{class C extends D \{}\overline{\texttt{C f}}\texttt{; K \}} \qquad \texttt{m} \notin \overline{\texttt{M}}}{mbody(\texttt{m}, \texttt{C}) = mbody(\texttt{m}, \texttt{D})}$$

Formální zápis

Expression typing:

$$\Gamma \vdash \mathtt{x} : \Gamma(\mathtt{x}) \qquad \text{(T-Var)}$$

$$\frac{\Gamma \vdash \mathtt{e}_0 : \mathtt{C}_0 \qquad \mathit{fields}(\mathtt{C}_0) = \overline{\mathtt{C}}\ \overline{\mathtt{f}}}{\Gamma \vdash \mathtt{e}_0.\mathtt{f}_i : \mathtt{C}_i} \qquad \text{(T-Field)}$$

$$\frac{\Gamma \vdash \mathtt{e}_0 : \mathtt{C}_0 \qquad \mathit{mtype}(\mathtt{m}, \mathtt{C}_0) = \overline{\mathtt{D}} \to \mathtt{C} \qquad \Gamma \vdash \overline{\mathtt{e}} : \overline{\mathtt{C}} \qquad \overline{\mathtt{C}} <: \overline{\mathtt{D}}}{\Gamma \vdash \mathtt{e}_0.\mathtt{m}(\overline{\mathtt{e}}) : \mathtt{C}} \qquad \text{(T-Invk)}$$

$$\frac{\mathit{fields}(\mathtt{C}) = \overline{\mathtt{D}}\ \overline{\mathtt{f}} \qquad \Gamma \vdash \overline{\mathtt{e}} : \overline{\mathtt{C}} \qquad \overline{\mathtt{C}} <: \overline{\mathtt{D}}}{\Gamma \vdash \mathtt{new}\ \mathtt{C}(\overline{\mathtt{e}}) : \mathtt{C}} \qquad \text{(T-New)}$$

$$\frac{\Gamma \vdash \mathtt{e}_0 : \mathtt{D} \qquad \mathtt{D} <: \mathtt{C}}{\Gamma \vdash (\mathtt{C})\mathtt{e}_0 : \mathtt{C}} \qquad \text{(T-UCast)}$$

$$\frac{\Gamma \vdash \mathtt{e}_0 : \mathtt{D} \qquad \mathtt{C} <: \mathtt{D} \qquad \mathtt{C} \neq \mathtt{D}}{\Gamma \vdash (\mathtt{C})\mathtt{e}_0 : \mathtt{C}} \qquad \text{(T-DCast)}$$

$$\frac{\Gamma \vdash \mathtt{e}_0 : \mathtt{D} \qquad \mathtt{C} \not<: \mathtt{D} \qquad \mathtt{D} \not<: \mathtt{C} \qquad \mathit{stupid\ warning}}{\Gamma \vdash (\mathtt{C})\mathtt{e}_0 : \mathtt{C}} \qquad \text{(T-SCast)}$$

Formální zápis

Method typing:

$$\frac{\overline{\mathtt{x}} : \overline{\mathtt{C}}, \mathtt{this} : \mathtt{C} \vdash \mathtt{e}_0 : \mathtt{E}_0 \qquad \mathtt{E}_0 <: \mathtt{C}_0 \qquad \mathtt{class\ C\ extends\ D\ \{...\}} \qquad \text{if } \mathit{mtype}(\mathtt{m}, \mathtt{D}) = \overline{\mathtt{D}} \to \mathtt{D}_0, \text{ then } \overline{\mathtt{C}} = \overline{\mathtt{D}} \text{ and } \mathtt{C}_0 = \mathtt{D}_0}{\mathtt{C}_0\ \mathtt{m}(\overline{\mathtt{C}}\ \overline{\mathtt{x}})\{\ \mathtt{return\ e}_0\mathtt{;}\ \}\ \mathtt{OK\ IN\ C}} \quad \text{(T-Method)}$$

Class typing:

$$\frac{\mathtt{K} = \mathtt{C}(\overline{\mathtt{D}}\ \overline{\mathtt{g}},\ \overline{\mathtt{C}}\ \overline{\mathtt{f}})\{\mathtt{super}(\overline{\mathtt{g}})\mathtt{;}\ \mathtt{this}.\overline{\mathtt{f}}=\overline{\mathtt{f}}\mathtt{;}\} \qquad \mathit{fields}(\mathtt{D}) = \overline{\mathtt{D}}\ \overline{\mathtt{g}} \qquad \overline{\mathtt{M}}\ \mathtt{OK\ IN\ C}}{\mathtt{class\ C\ extends\ D}\ \{\overline{\mathtt{C}}\ \overline{\mathtt{f}}\mathtt{;}\ \mathtt{K}\ \overline{\mathtt{M}}\}\ \mathtt{OK}} \quad \text{(T-Class)}$$

FJ redukce

- relace, kde e -> e’ (e se redukuje na e’ v jednom kroku)
- e ->* e’, je reflexivní a tranzitivní uzávěr relace ->
- redukční pravidla mohou být aplikována na jakoukoli část výrazu (není určeno pořadí ve kterém se redukce provádí), takováto redukce je nedeterministická => potřebujeme i pravidla typu: když platí e->e’, pak platí i e.f->e’.f.

FJ příklad

```
class A extends Object {
  A() { super(); }
}
class B extends Object {
  B() { super(); }
}
class Pair extends Object {
  Object fst;
  Object snd;
  Pair(Object fst, Object snd) {
    super(); this.fst=fst; this.snd=snd;
  }
  Pair setfst(Object newfst) {
    return new Pair(newfst, this.snd);
  }
}
```

FJ příklad

```
new Pair(new A(), new B()).setfst(new B())
```

```
new Pair(new B(), new B())
```

FJ příklad

```
((Pair)new Pair(new Pair(new A(), new B()), new A()).fst).snd
```

```
   ((Pair)new Pair(new Pair(new A(), new B()), new A()).fst).snd
→  ((Pair)new Pair(new A(),new B())).snd
→  new Pair(new A(), new B()).snd
→  new B()
```

```
new B()
```

FJ příklad

```
new Pair(new A(), new B()).setfst(new B())
→ [ new B()/newfst,
    new Pair(new A(),new B())/this ] new Pair(newfst, this.snd)
i.e., new Pair(new B(), new Pair(new A(), new B()).snd)
```

- / označuje substituci