CS 4110

Programming Languages & Logics

Lecture 29
Featherweight Java

9 November 2016

Announcements

- Homework #8 due tonight at 11:59pm
- No new homework out: Prelim II is next week

Object-Oriented Features

So far we've been mostly studying functional languages...

Object-Oriented Features

So far we've been mostly studying functional languages...

Today we'll study a core calculus developed by Igarashi, Pierce, and Wadler called *Featherweight Java*

Object-Oriented Features

So far we've been mostly studying functional languages...

Today we'll study a core calculus developed by Igarashi, Pierce, and Wadler called *Featherweight Java*

Featherweight Java is small by design: it has essential features including classes, inheritance, constructors, fields, methods, and casts, but omits everything else

Object-Oriented Features

So far we've been mostly studying functional languages...

Today we'll study a core calculus developed by Igarashi, Pierce, and Wadler called *Featherweight Java*

Featherweight Java is small by design: it has essential features including classes, inheritance, constructors, fields, methods, and casts, but omits everything else

Because the language is simple, its proof of type soundness is short and it is easy to extend

Question

What *is* Object-Oriented Programming?

Syntax

$P \quad ::= \quad \overline{CL}\ e$ *programs*

Syntax

P	$::=$	$\overline{CL}\ e$	*programs*
CL	$::=$	$\texttt{class}\ C\ \texttt{extends}\ C\ \{\overline{C\ f};\ K\ \overline{M}\}$	*classes*

Syntax

P	$::=$	$\overline{CL}\ e$	*programs*
CL	$::=$	$\texttt{class}\ C\ \texttt{extends}\ C\ \{\overline{C\ f};\ K\ \overline{M}\}$	*classes*
K	$::=$	$C(\overline{C\ f})\ \{\texttt{super}(\bar{f}); \overline{\texttt{this}.f = \bar{f};}\}$	*constructors*

Syntax

P	$::=$	$\overline{CL}\ e$	*programs*
CL	$::=$	$\texttt{class}\ C\ \texttt{extends}\ C\ \{\overline{C\ f};\ K\ \overline{M}\}$	*classes*
K	$::=$	$C(\overline{C\ f})\ \{\texttt{super}(\bar{f}); \overline{\texttt{this}.f = \bar{f};}\}$	*constructors*
M	$::=$	$C\ m(\overline{C\ x})\{\texttt{return}\ e\}$	*methods*

Syntax

P	$::=$	$\overline{CL}\ e$	*programs*
CL	$::=$	$\texttt{class}\ C\ \texttt{extends}\ C\ \{\overline{C\ f};\ K\ \overline{M}\}$	*classes*
K	$::=$	$C(\overline{C\ f})\ \{\texttt{super}(\bar{f}); \overline{\texttt{this}.f = \bar{f};}\}$	*constructors*
M	$::=$	$C\ m(\overline{C\ x})\{\texttt{return}\ e\}$	*methods*
e	$::=$	x	*expressions*
	$\mid$	$e.f$	
	$\mid$	$e.m(\overline{e})$	
	$\mid$	$\texttt{new}\ C(\overline{e})$	
	$\mid$	$(C)\ e$	

Syntax

P	$::=$	$\overline{CL}\ e$	*programs*
CL	$::=$	$\texttt{class}\ C\ \texttt{extends}\ C\ \{\overline{C\ f};\ K\ \overline{M}\}$	*classes*
K	$::=$	$C(\overline{C\ f})\ \{\texttt{super}(\bar{f});\ \overline{\texttt{this}.f = \bar{f};}\}$	*constructors*
M	$::=$	$C\ m(\overline{C\ x})\{\texttt{return}\ e\}$	*methods*
e	$::=$	x	*expressions*
	$\mid$	$e.f$	
	$\mid$	$e.m(\overline{e})$	
	$\mid$	$\texttt{new}\ C(\overline{e})$	
	$\mid$	$(C)\ e$	
v	$::=$	$\texttt{new}\ C(\overline{v})$	*values*

Syntax

P	$::=$	$\overline{CL}\, e$	*programs*
CL	$::=$	$\texttt{class}\, C\, \texttt{extends}\, C\, \{\overline{C\, f};\ K\, \overline{M}\}$	*classes*
K	$::=$	$C(\overline{C\, f})\, \{\texttt{super}(\bar{f}); \overline{\texttt{this}.f = \bar{f};}\}$	*constructors*
M	$::=$	$C\, m(\overline{C\, x})\{\texttt{return}\, e\}$	*methods*
e	$::=$	x	*expressions*
	$\mid$	$e.f$	
	$\mid$	$e.m(\overline{e})$	
	$\mid$	$\texttt{new}\, C(\overline{e})$	
	$\mid$	$(C)\, e$	
v	$::=$	$\texttt{new}\, C(\overline{v})$	*values*
E	$::=$	$[\cdot]$	*evaluation contexts*
	$\mid$	$E.f$	
	$\mid$	$E.m(\overline{e})$	
	$\mid$	$v.m(\overline{v}, E, \overline{e})$	
	$\mid$	$\texttt{new}\, C(\overline{v}, E, \overline{e})$	
	$\mid$	$(C)\, E$	

Example

```
class A extends Object { A() { super(); } }
class B extends Object { A() { super(); } }
```

Example

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

Example

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

Subtyping

$$\frac{}{C \leq C}\ \text{S-Refl}$$

Subtyping

$$\frac{}{C \leq C}\ \text{S-Refl}$$

$$\frac{C \leq D \quad D \leq E}{C \leq E}\ \text{S-Trans}$$

Subtyping

$$\frac{}{C \leq C}\ \textsc{S-Refl}$$

$$\frac{C \leq D \quad D \leq E}{C \leq E}\ \textsc{S-Trans}$$

$$\frac{P(C) = \texttt{class}\ C\ \texttt{extends}\ D\ \{\overline{C\ f};\ K\ \overline{M}\}}{C \leq D}\ \textsc{S-Class}$$

Field Lookup

$$\frac{}{\mathit{fields}(\texttt{Object}) = []}\ \textsc{F-Object}$$

Field Lookup

$$\frac{}{\mathit{fields}(\texttt{Object}) = []}\ \textsc{F-Object}$$

$$\frac{P(C) = \texttt{class}\ C\ \texttt{extends}\ D\ \{\overline{C\ f};\ K\ \overline{M}\} \qquad \mathit{fields}(D) = \overline{D\ g}}{\mathit{fields}(C) = \overline{D\ g}\ @\ \overline{C\ f}}\ \textsc{F-Class}$$

Method Body Lookup

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

Method Body Lookup

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

$$\frac{P(C) = \texttt{class}\ C\ \texttt{extends}\ D\ \{\overline{C\ f};\ K\ \overline{M}\} \qquad B\ m\ (\overline{B\ x})\ \{\texttt{return}\ e\} \notin \overline{M}}{mbody(m, C) = mbody(m, D)}\ \text{MB-Super}$$

Operational Semantics

$$E ::= [\cdot] \mid E.f \mid E.m(\overline{e}) \mid v.m(\overline{v}, E, \overline{e}) \mid \texttt{new}\, C(\overline{v}, E, \overline{e}) \mid (C)\, E$$

Operational Semantics

$$E ::= [\cdot] \mid E.f \mid E.m(\overline{e}) \mid v.m(\overline{v}, E, \overline{e}) \mid \texttt{new}\, C(\overline{v}, E, \overline{e}) \mid (C)\, E$$

$$\frac{e \to e'}{E[e] \to E[e']}\ \textsc{E-Context}$$

Operational Semantics

$$E ::= [\cdot] \mid E.f \mid E.m(\overline{e}) \mid v.m(\overline{v}, E, \overline{e}) \mid \texttt{new}\, C(\overline{v}, E, \overline{e}) \mid (C)\, E$$

$$\frac{e \to e'}{E[e] \to E[e']}\ \textsc{E-Context}$$

$$\frac{\mathit{fields}(C) = \overline{C\, f}}{\texttt{new}\, C(\overline{v}).f_i \to v_i}\ \textsc{E-Proj}$$

Operational Semantics

$$E ::= [\cdot] \mid E.f \mid E.m(\overline{e}) \mid v.m(\overline{v}, E, \overline{e}) \mid \texttt{new}\, C(\overline{v}, E, \overline{e}) \mid (C)\, E$$

$$\frac{e \to e'}{E[e] \to E[e']}\ \textsc{E-Context}$$

$$\frac{\mathit{fields}(C) = \overline{C\, f}}{\texttt{new}\, C(\overline{v}).f_i \to v_i}\ \textsc{E-Proj}$$

$$\frac{\mathit{mbody}(m, C) = (\overline{x}, e)}{\texttt{new}\, C(\overline{v}).m(\overline{u}) \to [\overline{x} \mapsto \overline{u}, \texttt{this} \mapsto \texttt{new}\, C(\overline{v})]e}\ \textsc{E-Invk}$$

Operational Semantics

$$E ::= [\cdot] \mid E.f \mid E.m(\overline{e}) \mid v.m(\overline{v}, E, \overline{e}) \mid \texttt{new}\, C(\overline{v}, E, \overline{e}) \mid (C)\, E$$

$$\frac{e \to e'}{E[e] \to E[e']}\ \textsc{E-Context}$$

$$\frac{\mathit{fields}(C) = \overline{C\, f}}{\texttt{new}\, C(\overline{v}).f_i \to v_i}\ \textsc{E-Proj}$$

$$\frac{\mathit{mbody}(m, C) = (\overline{x}, e)}{\texttt{new}\, C(\overline{v}).m(\overline{u}) \to [\overline{x} \mapsto \overline{u}, \texttt{this} \mapsto \texttt{new}\, C(\overline{v})]e}\ \textsc{E-Invk}$$

$$\frac{C \leq D}{(D)\, \texttt{new}\, C(\overline{v}) \to \texttt{new}\, C(\overline{v})}\ \textsc{E-Cast}$$

Method Type Lookup

$$\frac{P(C) = \texttt{class}\ C\ \texttt{extends}\ D\ \{\overline{C\ f};\ K\ \overline{M}\} \qquad B\ m\ (\overline{B\ x})\ \{\texttt{return}\ e\} \in \overline{M}}{mtype(m, C) = \overline{B} \to B}\ \text{MT-Class}$$

Method Type Lookup

$$\frac{P(C) = \texttt{class}\ C\ \texttt{extends}\ D\ \{\overline{C\ f};\ K\ \overline{M}\} \qquad B\ m\ (\overline{B\ x})\ \{\texttt{return}\ e\} \in \overline{M}}{mtype(m, C) = \overline{B} \to B}\ \text{MT-Class}$$

$$\frac{P(C) = \texttt{class}\ C\ \texttt{extends}\ D\ \{\overline{C\ f};\ K\ \overline{M}\} \qquad B\ m\ (\overline{B\ x})\ \{\texttt{return}\ e\} \notin \overline{M}}{mtype(m, C) = mtype(m, D)}\ \text{MT-Super}$$

Typing Rules

$$\frac{\Gamma(x) = C}{\Gamma \vdash x : C}\ \textsc{T-Var}$$

Typing Rules

$$\frac{\Gamma(x) = C}{\Gamma \vdash x : C}\ \textsc{T-Var} \qquad \frac{\Gamma \vdash e : C \qquad \mathit{fields}(C) = \overline{C\ f}}{\Gamma \vdash e.f_i : C_i}\ \textsc{T-Field}$$

Typing Rules

$$\frac{\Gamma(x) = C}{\Gamma \vdash x : C}\ \textsc{T-Var} \qquad \frac{\Gamma \vdash e : C \qquad \mathit{fields}(C) = \overline{C\ f}}{\Gamma \vdash e.f_i : C_i}\ \textsc{T-Field}$$

$$\frac{\Gamma \vdash e : C \qquad \mathit{mtype}(m, C) = \overline{B} \to B \qquad \Gamma \vdash \overline{e} : \overline{A} \qquad \overline{A} \leq \overline{B}}{\Gamma \vdash e.m(\overline{e}) : B}\ \textsc{T-Invk}$$

Typing Rules

$$\frac{\Gamma(x) = C}{\Gamma \vdash x : C}\ \textsc{T-Var} \qquad \frac{\Gamma \vdash e : C \qquad \mathit{fields}(C) = \overline{C\ f}}{\Gamma \vdash e.f_i : C_i}\ \textsc{T-Field}$$

$$\frac{\Gamma \vdash e : C \qquad \mathit{mtype}(m, C) = \overline{B} \to B \qquad \Gamma \vdash \overline{e} : \overline{A} \qquad \overline{A} \leq \overline{B}}{\Gamma \vdash e.m(\overline{e}) : B}\ \textsc{T-Invk}$$

$$\frac{\mathit{fields}(C) = \overline{C\ f} \qquad \Gamma \vdash \overline{e} : \overline{B} \qquad \overline{B} \leq \overline{C}}{\Gamma \vdash \texttt{new}\ C(\overline{e}) : C}\ \textsc{T-New}$$

Typing Rules

$$\frac{\Gamma(x) = C}{\Gamma \vdash x : C}\ \text{T-VAR} \qquad \frac{\Gamma \vdash e : C \qquad \mathit{fields}(C) = \overline{C\ f}}{\Gamma \vdash e.f_i : C_i}\ \text{T-FIELD}$$

$$\frac{\Gamma \vdash e : C \qquad \mathit{mtype}(m, C) = \overline{B} \to B \qquad \Gamma \vdash \overline{e} : \overline{A} \qquad \overline{A} \leq \overline{B}}{\Gamma \vdash e.m(\overline{e}) : B}\ \text{T-INVK}$$

$$\frac{\mathit{fields}(C) = \overline{C\ f} \qquad \Gamma \vdash \overline{e} : \overline{B} \qquad \overline{B} \leq \overline{C}}{\Gamma \vdash \texttt{new}\ C(\overline{e}) : C}\ \text{T-NEW}$$

$$\frac{\Gamma \vdash e : D \qquad D \leq C}{\Gamma \vdash (C)\, e : C}\ \text{T-UCAST}$$

Typing Rules

$$\frac{\Gamma(x) = C}{\Gamma \vdash x : C}\ \text{T-Var} \qquad \frac{\Gamma \vdash e : C \qquad \mathit{fields}(C) = \overline{C\ f}}{\Gamma \vdash e.f_i : C_i}\ \text{T-Field}$$

$$\frac{\Gamma \vdash e : C \qquad \mathit{mtype}(m, C) = \overline{B} \to B \qquad \Gamma \vdash \overline{e} : \overline{A} \qquad \overline{A} \leq \overline{B}}{\Gamma \vdash e.m(\overline{e}) : B}\ \text{T-Invk}$$

$$\frac{\mathit{fields}(C) = \overline{C\ f} \qquad \Gamma \vdash \overline{e} : \overline{B} \qquad \overline{B} \leq \overline{C}}{\Gamma \vdash \texttt{new}\ C(\overline{e}) : C}\ \text{T-New}$$

$$\frac{\Gamma \vdash e : D \qquad D \leq C}{\Gamma \vdash (C)\, e : C}\ \text{T-UCast} \qquad \frac{\Gamma \vdash e : D \qquad C \leq D \qquad C \neq D}{\Gamma \vdash (C)\, e : C}\ \text{T-DCast}$$

Typing Rules

$$\frac{\Gamma(x) = C}{\Gamma \vdash x : C}\ \text{T-VAR} \qquad \frac{\Gamma \vdash e : C \qquad \mathit{fields}(C) = \overline{C\ f}}{\Gamma \vdash e.f_i : C_i}\ \text{T-FIELD}$$

$$\frac{\Gamma \vdash e : C \qquad \mathit{mtype}(m, C) = \overline{B} \to B \qquad \Gamma \vdash \overline{e} : \overline{A} \qquad \overline{A} \leq \overline{B}}{\Gamma \vdash e.m(\overline{e}) : B}\ \text{T-INVK}$$

$$\frac{\mathit{fields}(C) = \overline{C\ f} \qquad \Gamma \vdash \overline{e} : \overline{B} \qquad \overline{B} \leq \overline{C}}{\Gamma \vdash \texttt{new}\ C(\overline{e}) : C}\ \text{T-NEW}$$

$$\frac{\Gamma \vdash e : D \qquad D \leq C}{\Gamma \vdash (C)\, e : C}\ \text{T-UCAST} \qquad \frac{\Gamma \vdash e : D \qquad C \leq D \qquad C \neq D}{\Gamma \vdash (C)\, e : C}\ \text{T-DCAST}$$

$$\frac{\Gamma \vdash e : D \qquad C \not\leq D \qquad D \not\leq C \qquad \textit{stupid warning}}{\Gamma \vdash (C)\, e : C}\ \text{T-SCAST}$$

Method Typing

$$\frac{mtype(m, D) = \overline{A} \to A \; implies \; \overline{A} = \overline{B} \text{ and } A = B}{override(m, D, \overline{B} \to B)} \textsc{Override}$$

Method Typing

$$\frac{mtype(m, D) = \overline{A} \to A \; implies \; \overline{A} = \overline{B} \text{ and } A = B}{override(m, D, \overline{B} \to B)} \textsc{Override}$$

$$\frac{\begin{array}{c}\overline{x : B}, \texttt{this} : C \vdash e : A \qquad A \leq B \\ P(C) = \texttt{class}\; C\; \texttt{extends}\; D\; \{\overline{C\; f};\; K\; \overline{M}\} \\ override(m, D, \overline{B} \to B)\end{array}}{B\; m(\overline{B\; x})\{\texttt{return}\; e\}\; OK \; in \; C} \textsc{Method-OK}$$

Class Typing

$$\frac{K = C(\overline{D\,g}, \overline{C\,f})\ \{\texttt{super}(\overline{g});\ \overline{\texttt{this}.f = \bar{f};}\ \} \qquad \mathit{fields}(D) = \overline{D\,g} \qquad \overline{M\ OK\ in\ C}}{\texttt{class}\ C\ \texttt{extends}\ D\ \{\overline{C\,f};\ K\,\overline{M}\}\ OK}\ \textsc{Class-OK}$$

Type Soundness

We can prove type soundness in *almost* the standard way...

Type Soundness

We can prove type soundness in *almost* the standard way...

Lemma (Preservation)

If $\Gamma \vdash e : C$ *and* $e \to e'$ *then there exists a type* C' *such that* $\Gamma \vdash e' : C'$ *and* $C' \leq C$.

Type Soundness

We can prove type soundness in *almost* the standard way...

Lemma (Preservation)

If $\Gamma \vdash e : C$ *and* $e \to e'$ *then there exists a type* C' *such that* $\Gamma \vdash e' : C'$ *and* $C' \leq C$.

Lemma (Progress)

Let e be an expression such that $\vdash e : C$. *Then either:*

1. *e is a value,*
2. *there exists an expression* e' *such that* $e \to e'$, *or*
3. $e = E[(B)\,(\mathit{new}\, A(\overline{v}))]$ *with* $A \not\leq B$.