In this lecture, we will develop a proof of type soundness for Featherweight Java in the usual way, as a corollary of progress and preservation. The details of these proofs will be a little different than the ones we have seen before, however, due to the presence of subtyping and casts.

1 Preservation

The proof of preservation relies on several supporting lemmas.

**Lemma (Method Typing).** If \( \text{mtype}(m, C) = D \rightarrow D \) and \( \text{mbody}(m, C) = (\pi, e) \) then there exists types \( C' \) and \( D' \) such that \( \pi : D \), this : \( C' \vdash e : D' \) and \( D' \leq D \).

**Lemma (Substitution).** If \( \Gamma, \pi : B \vdash e : C \) and \( \Gamma \vdash \overline{u} : B' \) with \( B' \leq B \) then there exists \( C' \) such that \( \Gamma \vdash [\pi \mapsto \overline{u}] e : C' \) and \( C' \leq C \).

**Lemma (Weakening).** If \( \Gamma \vdash e : C \) then \( \Gamma, x : B \vdash e : C' \).

**Lemma (Decomposition).** If \( \Gamma \vdash E[e] : C \) then there exists a type \( B \) such that \( \Gamma \vdash e : B \)

**Lemma (Context).** If \( \Gamma \vdash E[e] : C \) and \( \Gamma \vdash e : B \) and \( \Gamma, \delta : B' \vdash e' : B' \) with \( B' \leq B \) then there exists a type \( C' \) such that \( \Gamma \vdash E[e'] : C' \) and \( C' \leq C \).

**Lemma (Preservation).** If \( \Gamma \vdash e : C \) and \( e \rightarrow e' \) then there exists a type \( C' \) such that \( \Gamma \vdash e' : C' \) and \( C' \leq C \).

*Proof.* By induction on \( e \rightarrow e' \), with a case analysis of the last rule used in the derivation.

**Case E-Context:** \( e = E[e_1] \) and \( e_1 \rightarrow e'_1 \) and \( e' = E[e'_1] \)

By the decomposition lemma we have that there exists a type \( B \) such that \( \Gamma \vdash e_1 : B \). By the induction hypothesis applied to \( e_1 \) we have that there exists a type \( B' \) such that \( \Gamma \vdash e'_1 : B' \) and \( B' \leq B \). Then, by the context lemma we have that there exists a type \( C' \) such that \( \Gamma \vdash E[e'_1] : C' \) and \( C' \leq C \), as required.

**Case E-Proj:** \( e = \text{new } C_0(\overline{v}).f_1 \) and \( e' = v_i \) with \( \text{fields}(C_0) = \overline{C_f} \)

As the typing rules for Featherweight Java are syntax-directed, the last rule used in the derivation of \( \Gamma \vdash e : C \) must have been T-Field. Therefore we must also have a derivation \( \Gamma \vdash \text{new } C_0(\overline{v}) : D_0 \) with \( \text{fields}(D_0) = D g \) and \( C = D_i \). By a similar argument, the last rule used in this derivation must have been T-New and so \( D_0 = C_0 \) and we have derivations \( \Gamma \vdash \overline{v} : B \) with \( B \leq D_i \). From \( D_0 = C_0 \) (and as \( \text{fields} \) is a function) we have \( \overline{C_f} = D g_i \), and hence \( C = C_i \). Thus, \( \Gamma \vdash v_i : B_i \) with \( B_i \leq C_i \), as required.
Case E-Invk: \[ e = (\text{new } C_0(\overline{v})).m(\overline{u}) \text{ and } e' = [\overline{x} \mapsto \overline{u}, \text{this} \mapsto \text{new } C_0(\overline{v})]e \text{ with } m_{\text{body}}(m, C_0) = (\overline{x}, e) \]

By similar reasoning as in the previous case, the last two rules in the derivation of \( \vdash e : C \) must have been T-Invk and T-New with \( \Gamma \vdash \text{new } C_0(\overline{v}) : C_0 \) and \( \Gamma \vdash \pi : B \) and \( m_{\text{type}}(m, C_0) = \overline{C} \rightarrow C \) with \( B \leq \overline{C} \). By the method typing lemma, there exist types \( C'_0 \) and \( C' \) such that \( \overline{x} : C ; \text{this} : C'_0 \vdash e : C' \). By the substitution lemma we have \( \vdash [\overline{x} \mapsto \overline{u}, \text{this} \mapsto \text{new } C_0(\overline{v})]e : C'' \) with \( C'' \leq C' \). By weakening we have \( \vdash [\overline{x} \mapsto \overline{u}, \text{this} \mapsto \text{new } C_0(\overline{v})]e : C'' \). The required result follows as \( C'' \leq C \) by S-Trans.

Case E-Cast: \[ e = (C) \text{ (new } C_0(\overline{v})) \text{ and } e' = \text{new } C_0(\overline{v}) \text{ with } C_0 \leq C \]

By similar reasoning as the previous cases, the last two rules in the derivation of \( \Gamma \vdash e : C \) must have been T-UCast and T-New with \( \Gamma \vdash \text{new } C_0(\overline{v}) : C_0 \). The result is immediate as \( C_0 \leq C \).

\[ \square \]

## 2 Progress

The proof of progress also relies on a few supporting lemmas.

**Lemma (Canonical Forms).** If \( \vdash v : C \) then \( v = \text{new } C(\overline{v}) \).

**Lemma (Inversion).**

1. If \( \vdash (\text{new } C(\overline{v})).f_i : C_i \) then fields(C) = \( \overline{C} \setminus \overline{f} \) and \( f_i \in \overline{f} \).
2. If \( \vdash (\text{new } C(\overline{v})).m(\overline{u}) : C \) then \( m_{\text{body}}(m, C) = (\overline{x}, e) \) and \( |\overline{u}| = |\overline{v}| \).

**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 \( e \rightarrow e' \), or
3. \( e = E[(B) \ (\text{new } A(\overline{v}))] \) with \( A \nleq B \).

**Proof.** By induction on \( \vdash e : C \), with a case analysis on the last rule used in the derivation.

**Case T-Var:** \( e = x \) with \( \emptyset(x) = C \)

Can’t happen, as \( \emptyset(x) \) is undefined.

**Case T-Field:** \( e = e_0.f \) with \( \vdash e_0 : C_0 \) and \( \text{fields}(C_0) = \overline{C} \setminus \overline{f} \) and \( C = C_i \)

By the induction hypothesis applied to \( e_0 \) we have that either \( e_0 \) is a value, there exists \( e'_0 \) such that \( e_0 \rightarrow e'_0 \), or there exists \( E \) such that \( e_0 = E_0[(B) \ (\text{new } A(\overline{v}))] \) with \( A \nleq B \). We analyze each of these subcases:

1. If \( e_0 \) is a value then by the canonical forms lemma, \( e_0 = \text{new } C_0(\overline{v}) \) and by the inversion lemma \( f \in \overline{f} \). By E-Proj we have \( e \rightarrow v_i \).
Alternatively, if there exists an expression such that \( e_0 \to e'_0 \) then by E-CONTEXT we have \( e = E[e_0] \to E[e'_0] \) where \( E = [\cdot].f \).

3. Otherwise, if \( e_0 = E_0[(B) \new A(\overline{\sigma})] \) with \( A \not\subseteq B \) then we have \( e = E[(B) \new A(\overline{\sigma})] \) where \( E = [\cdot].f \), which finishes the case.

**Case T-Invk:** \( e = e_0.m(\overline{\sigma}) \) with \( e_0 : C_0 \) and \( m\text{type}(m, C_0) = \overline{B} \to C \) and \( \vdash \overline{\sigma} : \overline{A} \) and \( \overline{A} \leq \overline{B} \)

By the induction hypothesis applied to \( e_0 \) we have that either \( e_0 \) is a value, there exists \( e'_0 \) such that \( e_0 \to e'_0 \), or there exists \( E \) such that \( e_0 = E_0[(B) \new A(\overline{\sigma})] \) with \( A \not\subseteq B \). We analyze each of these subcases:

1. If \( e_0 \) is a value then by the canonical forms lemma, \( e_0 = \text{new} C_0(\overline{\sigma}) \). If \( \overline{\sigma} \) is a list of values \( \overline{\sigma} \), then by the inversion lemma we have \( |\overline{\sigma}| = |\overline{\tau}| \) where \( m\text{body}(m, C_0) = (\overline{\sigma}, e'_0) \). By E-INVK we have \( e \to [\overline{\sigma} \mapsto \overline{\tau}, \text{this} \mapsto \text{new} C_0(\overline{\sigma})] e'_0 \). Otherwise, let \( i \) be the least index of an expression in \( \overline{\sigma} \) that is not a value. By the induction hypothesis applied to \( e_i \) we have that \( e_i \) is a value, or there exists \( e'_i \) such that \( e_i \to e'_i \) or there exists \( E_i \) such that \( e_i = E_i[(B) \new A(\overline{\sigma})] \) and \( A \not\subseteq B \). In the first subsubcase, then we have a contradiction to our assumption that \( i \) is the index of the least expression in \( \overline{\sigma} \) that is not a value. Otherwise let \( E = (\text{new} C_0(\overline{\sigma})).m(e_1, \ldots, e_{i-1}, e_i, e_{i+1}, \ldots, |\overline{\sigma}|) \). In the second subcase, we have \( e = E[e_i] \to E[e'_i] \) by E-CONTEXT. In the third subcase, we have \( e = E[(B) \new A(\overline{\sigma})] \) with \( A \not\subseteq B \).

2. Alternatively, if there exists an expression such that \( e_0 \to e'_0 \) then by E-CONTEXT we have \( E[e_0] \to E[e'_0] \) where \( E = [\cdot].m(\overline{\sigma}) \).

3. Otherwise, if \( e_0 = E_0[(B) \new A(\overline{\sigma})] \) with \( A \not\subseteq B \) then we have \( e = E[(B) \new A(\overline{\sigma})] \) where \( E = [\cdot].m(\overline{\sigma}) \), which finishes the case.

**Case T-New:** \( e = \text{new} C(\overline{\sigma}) \) and \( \text{fields}(C) = \overline{C} \) and \( \vdash \overline{\sigma} : \overline{B} \) and \( \overline{B} \leq \overline{C} \)

If \( \overline{\sigma} \) is a list of values \( \overline{\sigma} \), then \( e \) is a value. Otherwise, let \( i \) be the least index of an expression in \( \overline{\sigma} \) that is not a value. By the induction hypothesis applied to \( e_i \) we have that \( e_i \) is a value, or there exists \( e'_i \) such that \( e_i \to e'_i \) or there exists \( E_i \) such that \( e_i = E_i[(B) \new A(\overline{\sigma})] \) and \( A \not\subseteq B \). We analyze each of these subcases:

1. If \( e_i \) is a value then we have a contradiction to our assumption that \( i \) is the index of the least expression in \( \overline{\sigma} \) that is not a value.
2. If there exists \( e'_i \) such that \( e_i \to e'_i \) then let \( E = (\text{new} C(e_1, \ldots, e_{i-1}, e_i, e_{i+1}, \ldots, |\overline{\sigma}|) \). By E-CONTEXT we have \( e = E[e_i] \to E[e'_i] \).
3. Otherwise, if there exists \( E \) with \( e_i = E_i[(B) \new A(\overline{\sigma})] \) and \( A \not\subseteq B \) then let \( E = (\text{new} C(e_1, \ldots, e_{i-1}, e_i, e_{i+1}, \ldots, |\overline{\sigma}|) \). By construction we have \( e = E[(B) \new A(\overline{\sigma})] \), which finishes the case.

**Case T-UCast:** \( e = (C) e \) with \( \vdash e_0 : D \) and \( D \leq C \)

By the induction hypothesis applied to \( e_0 \) we have that either \( e_0 \) is a value, there exists \( e'_0 \) such that \( e_0 \to e'_0 \), or there exists \( E \) such that \( e_0 = E_0[(B) \new A(\overline{\sigma})] \) with \( A \not\subseteq B \). We analyze each of these subcases:

1. If \( e_0 \) is a value then by the canonical forms lemma, \( e_0 = \text{new} D(\overline{\sigma}) \). By E-Cast we have \( e \to \text{new} D(\overline{\sigma}) \).
2. Alternatively, if there exists an expression such that $e_0 \to e'_0$ then by E-CONTEXT we have $e = E[e_0] \to E[e'_0]$ where $E = (C) \mathllap{[\cdot]}$.

3. Otherwise, if $e_0 = E_0[(B) \text{new } A(\overline{v})]$ with $A \not\subseteq B$ then we have $e = E[(B) \text{new } A(\overline{v})]$ where $E = (C) \mathllap{[\cdot]}$, which finishes the case.

**Case T-DCast:** $e = (C) e$ with $\vdash e_0 : D$ and $C \leq D$ and $C \not= D$

By the induction hypothesis applied to $e_0$ we have that either $e_0$ is a value, there exists $e'_0$ such that $e_0 \to e'_0$, or there exists $E$ such that $e_0 = E_0[(B) \text{new } A(\overline{v})]$ with $A \not\subseteq B$. We analyze each of these subcases:

1. If $e_0$ is a value then by the canonical forms lemma we have that $e = \text{new } D(\overline{v})$. Let $E = [\cdot]$. We immediately $e = E[(C) \text{new } C(\overline{v})]$ with $D \not\subseteq C$.

2. Alternatively, if there exists an expression such that $e_0 \to e'_0$ then by E-CONTEXT we have $e = E[e_0] \to E[e'_0]$ where $E = (C) \mathllap{[\cdot]}$.

3. Otherwise, if $e_0 = E_0[(B) \text{new } A(\overline{v})]$ with $A \not\subseteq B$ then we have $e = E[(B) \text{new } A(\overline{v})]$ where $E = (C) \mathllap{[\cdot]}$, which finishes the case.

**Case T-SCast:** similar to the previous case.