Formal Methods for Java
Lecture 16: Abnormal Termination in Key

Jochen Hoenicke

Software Engineering
Albert-Ludwigs-University Freiburg

December 16, 2011
Abnormal termination in Java is caused by

- a `break` statement,
- a `continue` statement,
- a `return` statement,
- a `throw` statement, or
- a statement that throws an exception.
The formula $\langle \alpha \rangle \phi$ holds,
- iff $\alpha$ terminates normally and $\phi$ holds afterwards.

The formula $[\alpha] \phi$ holds,
- if $\alpha$ terminates normally and $\phi$ holds afterwards.
- if $\alpha$ terminates abnormally.
- if $\alpha$ does not terminate at all.
Reasoning about exceptions.

How can we express that statement $\alpha$ throws an exception?

- $\langle\{\alpha\}\rangle \phi$ is equivalent to $\text{false}$ if $\alpha$ throws an exception or does not terminate
- $[\{\alpha\}] \phi$ is equivalent to $\text{true}$ if $\alpha$ throws an exception or does not terminate
- The trick is to put an exception handler into the code:

  $\langle\{\text{Throwables } \text{thrown} = \text{null};$
  
  try { $\alpha; \}$
  
  catch (Throwables $ex)\{ \text{thrown} = \text{ex}; \}} \rangle \text{thrown} \neq \text{null}$
Reasoning with try-catch blocks

Many DL-rules in KeY just skip opening of try blocks, e.g.
\[ \text{\textbackslash find} ( \langle \{ .. \# loc = \# se .. \} \rangle \text{ post } ) \]
\[ \text{\textbackslash replace with} ( \{ \# loc := \# se \} \langle \{ .. .. \} \rangle \text{ post } ) \]

Here .. stands for an arbitrary number of opening try-blocks, labelled blocks and normal blocks.

Example:

\[ \langle \{ \textbf{try} \{ label : \{ \textbf{try} \{ x = 5 .. \} \} \} \rangle \phi \]

is replaced with

\[ \{ x := 5 \} \langle \{ \textbf{try} \{ label : \{ \textbf{try} \{ .. \} \} \} \rangle \phi \]
Reasoning with try-catch blocks (2)

When an exception is thrown, the surrounding try blocks become important:
\[\text{try} \{ \text{throw } \#se; \#slist1 \} \]
\[\text{catch} (\#t \#v0) \{ \#slist2 \} \ldots \}\] \[\text{post} \]

1. **Throwing a handled exception:** \[\#se \text{ instanceof } \#t\]
\[\text{replacewith}( \langle \{ \ldots \#t \#v0 = \#se; \#slist2 \ldots \} \rangle \text{post} \)

2. **Throwing an unhandled exception:** \[! (\#se \text{ instanceof } \#t)\]
\[\text{replacewith}( \langle \{ \ldots \text{throw } \#se; \ldots \} \rangle \text{post} \)

3. **Throwing a null pointer:** \[\#se = \text{null}\]
\[\text{replacewith}( \langle \{ \ldots \text{try} \{ \text{throw new } \text{NullPointerException}(); \#slist1 \}
\text{catch} (\#t \#v0) \{ \#slist2 \} \ldots \} \rangle \text{post} \)

The KeY system defines a single rule:
\[\text{replacewith}( \langle \{ \ldots \text{if (}\#se = \text{null}) \text{then}
\text{try} \{ \text{throw new } \text{NullPointerException}(); \#slist1
\text{catch} (\#t \#v0) \{ \#slist2 \}
\text{else if (}\#se \text{ instanceof } \#t) \text{then}
\#t \#v0 = \#se; \#slist2
\text{else throw } \#se;
\ldots \} \rangle \text{post} \)