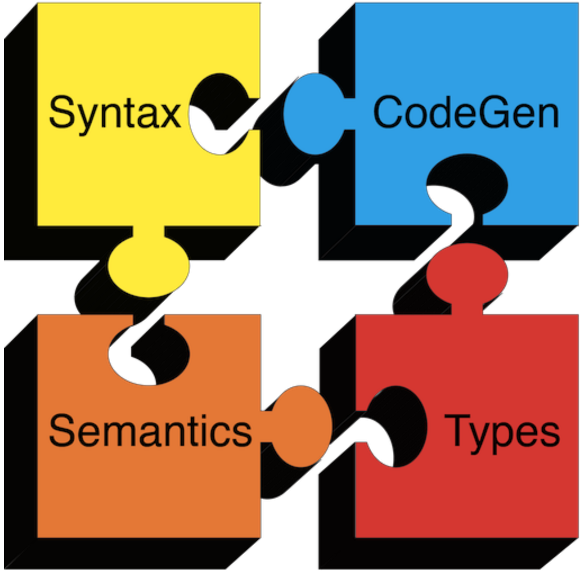


Compilers

Java Exceptions

Java Exceptions

- Deep in a section of code, you encounter an unexpected error
  - Out of memory
  - A list that is supposed to be sorted is not
  - etc.

- What do you do?

Java Exceptions

- Add a new type (class) of *exceptions*

- Add new forms
  - **try** { something } **catch**(x) { cleanup }
  - **throw** exception

expr

Java Exceptions

```
class Foo {
    public static void main(String[] args) {
        try { X(); } catch (Exception e) {
        System.out.println(“Error!”) } }

    public void X() throws MyException {
        throw new MyException();
    }
}
```

Java Exceptions

T(v) = an exception that has been thrown with value v

v = an ordinary value (an object)

$$\frac{E \vdash e_1 : v_1}{E \vdash \mathrm{try}\{e_1\}\ \mathrm{catch}(x)\ \{e_2\} : v_1}$$

$$\frac{\begin{array}{c} E \vdash e_1 : T(v_1) \\ E[x \leftarrow v_1] \vdash e_2 : v_2 \end{array}}{E \vdash \mathrm{try}\{e_1\}\ \mathrm{catch}(x)\ \{e_2\} : v_2}$$

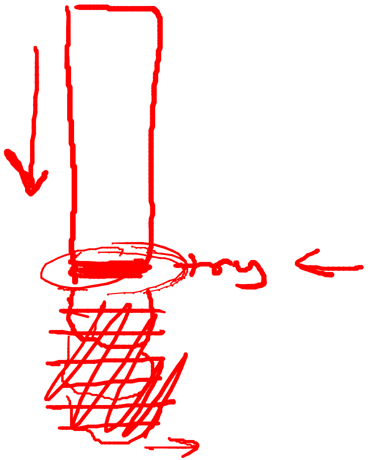
Java Exceptions

$$\frac{E \vdash \underline{e} : \underline{v}}{E \vdash \text{throw } \underline{e} : \underline{T(v)}}$$

$$\frac{E \vdash \underline{e_1} : \underline{T(v_1)}}{E \vdash \underline{e_1 + e_2} : \underline{T(v_1)}}$$

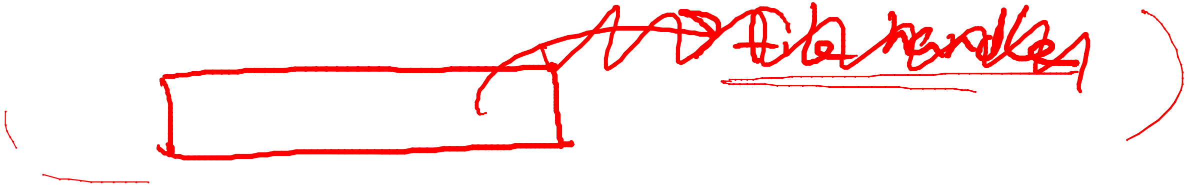
Java Exceptions

- When we encounter a try
  - Mark current location in the stack
- When we throw an exception
  - Unwind the stack to the first try
  - Execute corresponding catch
- More complex techniques reduce the cost of try and throw



Java Exceptions

What happens to an uncaught exception thrown during object finalization?



No one

Java Exceptions

- Methods must declare types of exceptions they may raise

  public void X() throws MyException

  - Checked at compile time
  - Some exceptions need not be part of the method signature
    - e.g., dereferencing null

- Other mundane type rules
  - throw must be applied to an object of type Exception