Special Topic E: Scope

- The Rule
- The Reasons
The Rule

- Names are only valid in the scope in which they were declared
  - Class
  - Method
  - Block statement
    - (With or without {}) 
  - Loop

- Limitations + Flexibility
What's wrong with this code?

for (int i=0; i<10; i++)
    if (val % i == 0)
        break;

if (i == 10)
    System.out.printf("Loop completed.\n");
else
    System.out.printf("Loop ended at %d.\n", i);
What's wrong with this code?

```java
for (int i=0; i<10; i++)
    if (val % i == 0)
        break;

if (i == 10)
    System.out.printf("Loop completed.\n");
else
    System.out.printf("Loop ended at %d.\n", i);
```

CompileError1.java:11: error: cannot find symbol
  if (i == 10)
     ^
symbol:   variable i
location: class CompileError1
```
What's wrong with this code?

for (int i=0; i<10; i++)
    if (val % i == 0)
        break;

if (i == 10)
    System.out.printf("Loop completed.\n");
else
    System.out.printf("Loop ended at %d.\n", i);

Variables declared as part of a for() loop are not accessible outside of the loop.
The Same Code, Fixed

```java
int i;
for (i=0; i<10; i++)
    if (val % i == 0)
        break;

if (i == 10)
    System.out.printf("Loop completed.\n");
else
    System.out.printf("Loop ended at %d.\n", i);
```

Declare the variable before the loop begins.
What's wrong with this code?

```java
if (args.length == 0)
{
    Scanner in = new Scanner(System.in);
}
else
{
    Scanner in = new Scanner(new File(args[0]));
}
while (in.hasNext())
    System.out.println(in.next());
```
What's wrong with this code?

```java
if (args.length == 0)
{
    Scanner in = new Scanner(System.in);
}
else
{
    Scanner in = new Scanner(new File(args[0]));
}
while (in.hasNext())
    System.out.println(in.next());
```

CompileError2.java:20: error: cannot find symbol
while (in.hasNext())
  ^
symbol:   variable in
location: class CompileError2
What's wrong with this code?

```java
if (args.length == 0) {
    Scanner in = new Scanner(System.in);
} else {
    Scanner in = new Scanner(new File(args[0]));
}
while (in.hasNext())
    System.out.println(in.next());
```

Variables declared inside a block statement are not accessible outside of the block.
What's wrong with this code?

```java
Scanner in;
if (args.length == 0) {
    in = new Scanner(System.in);
} else {
    in = new Scanner(new File(args[0]));
}
while (in.hasNext())
    System.out.println(in.next());
```
What's wrong with this code?

void foo()
{
    int counter = 0;
    bar();
}

void bar()
{
    System.out.println("count="+counter);
}
What's wrong with this code?

```java
void foo()
{
    int counter = 0;
    bar();
}

void bar()
{
    System.out.println("count=\"+counter);;
}
```

Local variables of one method are not accessible from any other method – include other copies of the same method.
Special Topic E: Scope

- The Rule
- The Reasons
Flexibility

- Scope allows you re-use names without worry
  - Local variables of methods
  - Recursion
  - Re-use loop counters: `for(int i=0`
  - Method names & constants

- Scope allows the compiler to know the type of variables
Static Methods and Constants

- What is the type of these names?

  `parseInt()`
  `main()`
  `PI`
  `MAX_VALUE`
  `format()`

These names are incomplete.

We don't know what class they are part of.
Static Methods and Constants

• What is the type of these names?

- `parseInt()`
- `main()`
- `PI` (Pi)
- `MAX_VALUE`
- `format()`

```
Byte.MAX_VALUE
Short.MAX_VALUE
Integer.MAX_VALUE
Long.MAX_VALUE
Float.MAX_VALUE
Double.MAX_VALUE
```
The **Bad Old Days of C**

- C had scope rules
  - Loops
  - Blocks
  - Methods (functions)

- **But no static methods or fields**\(^1\)

\(^1\) C used the keyword static, but for a different purpose than Java!
The **Bad Old Days of C**

- C programmers made longer names to make up for no scope at the global level:
  
  ```c
  MAX_INT
  CRYPTO_destroy_dynlockid()
  ```

- Or came up with terrible “standard” abbreviations:
  
  ```c
  vsnprintf()
  atoi()
  stroull()
  yylval
  ```
Special Topic E: Scope

- The Rule
- The Reasons

Summary