

Assertions

Assertion

- A command using which we can check our assumptions about the program
 - E.g. a method measuring the speed of a particle requires that the calculated speed should be lower than the speed of light
- Since J2SE 1.4
- Contains a logical expression
 - Which has to be (should be) true at some point of the code
 - If it isn't true, an error occurs
- One of the quickest and most efficient ways to find and repair errors
 - And it also increases maintainability

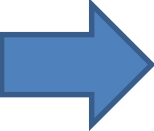
assert command

- It has two forms:
 - `assert expression ;`
 - When it is executed, the logical expression is evaluated, and if it is false, an `AssertionError` is raised (without any additional information)
 - `assert expression1 : expression2 ;`
 - Logical *expression*₁ is evaluated, if it is false, an `AssertionError` is raised, the constructor of which receives the value of *expression*₂ as a parameter
 - Which can be any non-void value!

When to use?

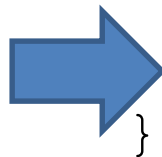
- Inner invariants
- Control structure invariants
- Pre- and postconditions, class invariants

Inner invariant

```
if (i % 3 == 0) {  
    ...  
}  
else  
    if (i % 3 == 1) {  
        ...  
    }  
else {  
    // Here (i % 3 == 2)  assert i % 3 == 2 : i  
    ...  
}
```

Inner invariant in a switch

```
switch (color) {  
  case Color.SPADES:  
    ...  
    break;  
  case Color.HEARTS:  
    ...  
    break;  
  case Color.DIAMONDS:  
    ...  
    break;  
  case Color.CLUBS:  
    ...  
}
```

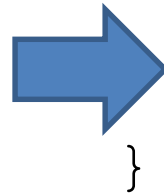


```
case Color.SPADES: ...  
  break;  
case Color.HEARTS: ...  
  break;  
case Color.DIAMONDS:  
  ...  
  break;  
case Color.CLUBS: ...  
  break;  
default:  
  assert false :  
  color;
```

```
switch (color) {
```

Control structure invariant

```
void foo() {  
    for (...) {  
        if (...)  
            return;  
    }  
    // This point can  
    never be reached  
}
```



```
void foo() {  
    for (...) {  
        if (...)  
            return;  
    }  
    assert false;  
}
```

Preconditions

- For nonpublic methods
 - Do not use for public methods! (see later)

```
private void setRefreshInterval(int interval) {  
// Check whether the parameter value is correct  
    assert interval > 0 &&  
           interval <= 1000/MAX_REFRESH_RATE : interval;  
  
    ...  
    // Set refresh interval  
}
```


Postconditions

```
//Returns value  $\text{this}^{-1} \bmod m$ 
public BigInteger modInverse(BigInteger m) {
    if (m.signum <= 0)
        throw new ArithmeticException("Modulus not positive: "
            + m);

    ...
    // Doing the calculation
    assert this.multiply(result).mod(m).equals(ONE) : this;
    return result;
}
```

Class invariants

- Special inner invariants that are true to all instances of a given class at a given point of time
 - Except for when the instance is being shifted from a consistent state to another consistent state
- For describing connections between attributes
- Have to be true before/after the execution of every method

```
// True if tree is balanced
```

```
private boolean balanced() { ... }
```

All public methods and constructors contain before the return statement:

```
assert balanced();
```

When not to use?

- Checking arguments of public methods
 - This has to be done if assertions are enabled or disabled as well
 - An appropriate runtime exception has to be thrown
 - E.g. `IllegalArgumentException`, `IndexOutOfBoundsException`, `NullPointerException`
- For activities that are necessary for the normal operation of the application, e.g.

```
assert names.remove(null);
```

 instead of this

```
boolean nullsRemoved = names.remove(null);  
assert nullsRemoved;
```
- Expressions in assertions have to be free of side-effects!
 - Except for only a state from another assertion is modified

Requiring Assertions to be enabled

```
static {  
    boolean assertsEnabled = false;  
    assert assertsEnabled = true;  
    // Deliberate side-effect!!!  
    if (!assertsEnabled)  
        throw new  
        RuntimeException("Asserts must be  
        enabled!!!");  
}
```