

# Module 3 – Part 3 – Brent Wenerstrom

- Numeric promotion
- Casting

# Numeric Promotion

- How does Java interpret the following expression:

```
'a' + 2
```

- How would you perform the following:

```
5 cm + 2 inches
```

# Numeric Promotion

- Java can only use basic operators when the operands are the same type
  - For example: `2.0 + 1.0`
- Java has specific rules about adjusting type before applying operands such `+`, `-`, `/`, `*`

# Numeric Promotion Rules

1. Promote smaller (in bytes) type to larger type
2. When mixing floating point and integral, promote integral to floating point
3. `byte` , `short` and `char` are promoted to `int` when part of operation
4. Output of expression is same type as operands

# Numeric Promotion: Rule 1

Promote smaller (in bytes) type to larger type

- Computer math on same types ONLY

# Numeric Promotion: Rule 1 Example

Example: `result = a + b`

- Before

variable	type	value	bytes	type max
a	int	40,000	[-][-][-][-]	2,147,483,647
b	short	3,000	[-][-]	32,767
result	-	-	-	-

# Numeric Promotion: Rule 1 Example

Example: `result = a + b`

- After

variable	type	value	bytes	type max
a	int	40,000	[-][-][-][-]	2,147,483,647
b	int	3,000	[-][-][-][-]	2,147,483,647
result	int	43,000	[-][-][-][-]	2,147,483,647

# Numeric Promotion: Rule 2

When mixing floating point and integral, promote integral to floating point

- Accounting for decimal requires different representation
- Only accurate result is floating point

Example:

- Before: `result = 2 + 4.2`
- After: `result = 2.0 + 4.2`



## Numeric Promotion: Rule 3

`byte` , `short` and `char` are promoted to `int` when part of operation

- Just a Java specific thing

# Numeric Promotion: Rule 3 Example

Example: `result = x + y`

- Before

variable	type	value	bytes	type max
x	byte	1	[-]	127
y	short	2	[-][-]	32,767
result	-	-	-	-

# Numeric Promotion: Rule 3 Example

Example: `result = x + y`

- After

variable	type	value	bytes	type max
x	int	1	[-][-][-][-]	2,147,483,647
y	int	2	[-][-][-][-]	2,147,483,647
result	int	3	[-][-][-][-]	2,147,483,647

# Numeric Promotion: Rule 4

Output of expression is same type as operands

- Whatever type is decided is the type of the result

Examples (after 1-3 rules performed):

- `long + long = long`
- `double / double = double`
- ~~`short * short = short`~~ (impossible: see rule 3)

# Primitive Ranks

- Reference table for numeric promotion

	smaller	->	->	larger
Integral	byte	short	int	long
Floating	float	double		
Other	char	int		

# Numeric Promotion Examples

What is the resulting type of the last expression?

```
int x = 1;  
long y = 33;  
x * y;
```

Note: Ugly way to find out the boxed type.

```
((Object) (x * y)).getClass().getName();
```

Demo run with `java -jar bsh-2.0b4.jar`

# Numeric Promotion Examples

What is the resulting type of the last expression?

```
double x = 39.21;  
float y = 2.1;  
x + y;
```

# Answer

Trick question answer

```
Fail.java:6: error: incompatible types: possible lossy  
conversion from double to float
```

```
    float y = 2.1;  
              ^
```

1 error



# Understanding the Trick

- Literal integral evaluates to `int`
  - `((Object) 5).getClass().getName();`
  - Result: `java.lang.Integer`
  - Java does not complain converting to `byte`, `short`, `int`, `long`, `char`
- Literal float evaluates to `double`
  - Java complains shortening bytes from `double` to `float`

# Numeric Promotion Examples (take 2)

What is the resulting type of the last expression?

```
double x = 39.21;  
float y = 2.1f;  
x + y;
```

# Numeric Promotion Examples

What is the resulting type of the last expression?

```
short x = 10;  
short y = 3;  
x / y;
```

# Numeric Promotion Examples

What is the resulting type of the last expression?

```
short x = 14;  
float y = 13;  
double z = 30;  
x * y / z;
```

# Answer

```
x : short -> int -> float
```

```
x * y : float -> double
```

```
x * y / z : double
```

# Casting

- How to force numeric type change

Example:

```
System.out.println(2 / 3);
```

# Casting

```
2 / 3;
```

Sometimes we want to convert a type before math:

```
int x = 2;  
int y = 3;  
(float) 2 / 3;
```

# Casting with Classes

- An object reference is of multiple types at the same time

```
String s = "ab";
```

- Reference `s` is both a `String` and an `Object`.
- `(Object) s` outputs a reference of type `Object` but the bytes don't change.



# Casting with Classes

Contrived example:

```
Object o = "abc";  
o.length();
```

```
$ javac Fail.java  
Fail.java:8: error: cannot find symbol  
                System.out.println(o.length());  
                                   ^  
    symbol:   method length()  
    location: variable o of type Object  
1 error
```

# Casting with Classes

Fixed example:

```
Object o = "abc";  
System.out.println( ((String)o).length() );
```

```
$ javac Fail.java  
3
```

Casting objects will make more sense when we have more class types and hierarchies to talk about.

# Definitions

- class: "blueprint" of object structure.
- method: block of code executable through its name.
- object: instance of class.
- instantiation: to create an object from class definition.
- constructor: initializing method within a class.
- parameter: receiving variable in a method definition
- numeric promotion: how Java unifies types before performing binary operations
- casting: forcing a change of the evaluated type of a variable
- binary operator: mathematical mapping of two parameters to one (example: +)