Variables, Operators, Conditionals and Loops

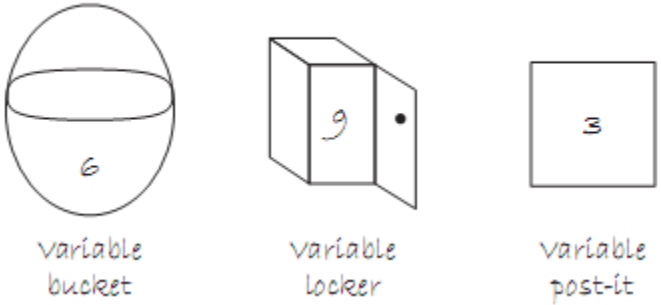
Everything You Need to Know

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Variables

- A variable is like a bucket.
- A variable is like a storage locker.
- A variable is like a post-it note.



- A variable is a named pointer to a location in the computers memory.
- Variable must have a name.

- The type of the variable defines the kind of data stored in that variable.
- Variable must have a type.

Variables

- A variable is a typed and named storage location
  - *<type> <name> [= <value>];*

- Simple types

		Limit
• **byte**	(-10, 103)	8-bits [-128, 127]
• **int**	(-10, 103)	32-bits [-2147483648 2147483647]
• **float**	(3.1415, -9.8)	32-bits [-3.40282347E+38 3.40282347E+38]
• **boolean**	(true, false)	1-bit [0, 1]
• **char**	(´a´,´!´)	
• **String**	("this is a string")	

Variable examples

<type> <name> [= <value>];

```
int myCount;
```
←Declaration

```
myCount = 40;
```
←Initialization

```
String myLanguage = "Processing";
```
←Declaration AND initialization

```
boolean isALanguage = true;
char myChar =  a ;
float myFloat = 3.01;
int yourCount = myCount;
```
←Initializing using another variable's value

Variables Note

- Give variables meaningful names
- Initialize variables before use
- Adhere to naming conventions (camelBack notation)
  - `isABoy` ok
  - `IsABoy` wrong (well…sort of…)
- Size matters (or in this case, case)
  - `myVariable` **IS NOT** `myvariable`

Operators

Operators perform transformations on variables

- `=` (assign)
- `+, -, *, /, %` (addition, multiply..)
- `+=, -=, *=, /=, %=` (add assign, multiply assign..)
- `> , >= , <, <=, ==, !=` (greater than, greater or equal..)

- `&&, ||` (logical AND, logical OR)
- `++, --` (increment, decrement)
- `(?:)` (conditional)
- `()` for precedence

Operator examples (non exhaustive)

- `int x = 12;`
- `int y = 6;`
- `int xDivY = x / y;`
- `boolean xDivYIsTwo = (xDivY == 2);`
- `x++; --y;`
- `x = x - y; y = y + x; x = y - x;`
- `float temp = 98.2;`
- `temp = temp % 5;`
- `x = (y > 6) ? 2 : 1;`
- `temp = x` (allowed… no precision loss)
- `x = temp` (not allowed…precision loss)

Conditionals

Two sorts of conditionals

- ```
  if (<boolean condition>) {
      [<statement>;]*
  } else {
      [<statement>;]*
  }
  ```

- ```
  switch (<variable>) {
  [case <value>:[<statement>;]*]*
  [default:[<statements>;]*]
  }
  ```

if (<boolean condition>) { [<statement> ;] }*
else { [<statement> ;] }*

```
String val;
int x = 5;

if (x == 5) { val = "five"; }
else { val = "not five"; }
println("x = " + val);
```

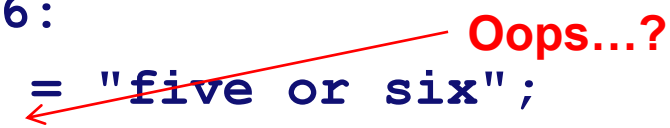
A shortcut for writing an if and else structure *(?:)*

```
result = condition ? expression1 : expression2
```

```
println("x = " + ((x == 5) ? "five" : "not five"));
```

switch (<variable>) {
[case <value>: [<statement> ;]]**
[default: [<statements> ;]]*
}

```
int x = 5; String val;
switch (x) {
case 0:
  val = "zero";
  break;
case 5:
case 6:
  val = "five or six";
default:
  val = "unknown";
 break;
}
```



Conditionals Do's and don'ts

```
int x = 0;
if (x = 0) { println("x is zero"); }
else { println("x is not zero"); }
```

= != ==

```
if (x == 0) {
   println("x is zero");
} else {
   println("x is not zero");
}
```

← Use indentation

Loops (for repetitive actions)

3 sorts of loops

- ***for (<start>; <condition>; <action>) {***
  [<statement>;]*
  }

- ***while (<condition>) {***
  [<statement>;]*
  }

- ***do {***
  [<statement>;]*
  } while (<condition>);

for (<start>; <condition>; <action>) {
*   <statements>;*
}

```
int i;
for (i=0; i<5; i++) print(i);
println();

int  j;
for (i=0,j=5; i<=5; i++,j--) {
  println(i);
  println(j);
}
```

while (<condition>) { <statements>; }

```
int i = 0;
while (i<5) {
  i++;
  println(i);
}
while (i<5) {
  i++;
  println(i);
}
```

do {
*   <statements>;*
} while (<condition>);

```
int i = 0;
do {
  i++;
  println(i);
} while (i < 5);
do {
  i++;
  println(i);
} while (i < 5);
```

Homework

- Read Chapter 4 , 5 , 6 of “Learning Processing”.
- Make the exercises as pointed out in the Tasks file.
- Hand in Exercise 7 (You may be creative).