Boolean Expressions

&&

Decision Making Statements

Objectives:

- Learn about the boolean data type
- Learn the syntax for if-else statements
- Learn about relational and logical operators, De Morgan's laws, short-circuit evaluation
- Learn when to use nested if-else statements, if-else-if sequences, the switch statement

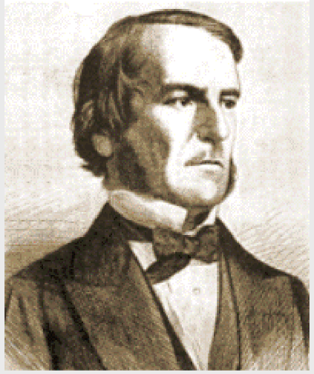
if-else statement

```
if ( <condition> )
{
    < statements >
} else {
    < other statements >
}
```

```
if ( <condition> )
{
    < statements >
}
```

else clause is optional

Brackets are mandatory; braces are optional however Java only executes only the first line of code without them which may cause logic errors.



Boolean Data Type

- George Boole (1815 - 1864)
- boolean variables may have only two values, true or false.
- You define boolean fields or boolean local variables the same way as other variables.

boolean
true
false

Reserved words

```
private boolean hasMiddleName;
boolean isRolling = false;
```

Boolean Expressions

- In if (<condition>) <condition> is a Boolean expression.
- A Boolean expression evaluates to either true or false.
- Boolean expressions are written using boolean variables and ***relational and logical*** operators.

Relational Operators

<, >, <=, >=, ==, !=

is equal to

is NOT equal to

Relational Operators (cont'd)

- Apply to numbers or chars:

```
if  ( count1 <= count2 ) ...
if  ( sum != 0 ) ...
if  ( letter == 'Y' ) ...
```

- Avoid use of == or != with doubles because false positive MAY be caused by rounding.

```
double x = 7.0;
double y = 3.5;
if (x / y == 2.0)
    ...
```

May be false!

Relational Operators (cont'd)

- Be careful: using == and != with objects (for example, Strings): references (addresses) are compared rather than values (the contents)

```
String cmd = c.readLine();
if ( cmd == "Help" ) ...
```

Wrong!
(always false)

Comparing strings

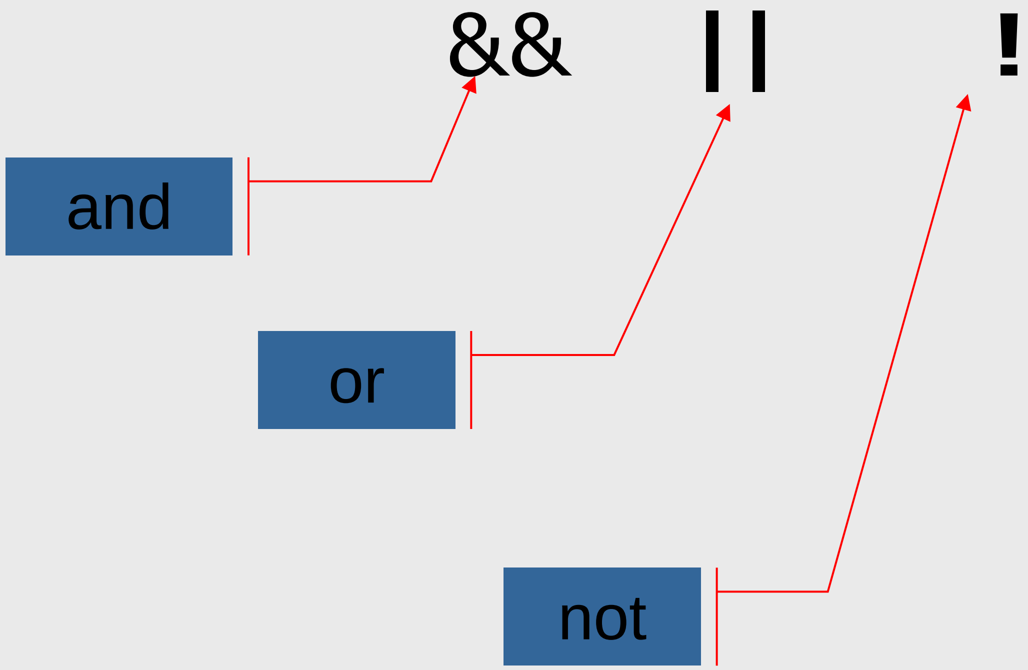
- Use the **equals** or ***equalsIgnoreCase*** methods to compare Strings:

```
String cmd = c.readLine();
if ( cmd.equals ("Help") )   ...
```

or

```
if (  "Help".equals (cmd) )   ...
```

Logical Operators



Logical Operators

- AND

(*condition1* && *condition2*) is true if both *condition1* and *condition2* are true

- OR

(*condition1* || *condition2*) is true if *condition1* or *condition2* (or both) are true

- NOT

!*condition1* is true if and only if *condition1* is false

Logical Operators (cont'd)

- &&, ||, and ! obey the laws of formal logic called *De Morgan's Laws*:

```
! (p  &&  q)   ==     ( !p  ||  !q )
! (p  ||  q)   ==    ( !p  &&  !q )
```

- Example:

```
if ( ! ( x => -10  &&  x <= 10 ) ) ...
if ( x < -10  ||  x > 10 ) ...
```

Easier to read

Ranks of Operators

Highest	! –(*unary*) ++ – – (*cast*)
	* / %
	+ –
	< <= > >= == !=
	&&
Lowest	||

Easier to read

```
if ( ( ( year % 4 ) == 0 ) && ( month == 2 ) ) ...
if ( year % 4 == 0 && month == 2 ) ...
```

Short-Circuit Evaluation

if (*condition1* **&&** *condition2*) ...

If *condition1* is false, then *condition2* is not evaluated (the result is false anyway)

```
if ( x >= 0  &&  Math.sqrt (x) < 15.0) ...
```

Always OK: won’t get to sqrt if x < 0

if (*condition1* **||** *condition2*) ...

If *condition1* is true, then *condition2* is not evaluated (the result is true anyway)

Nested if-else

```
if ("forward".equals(cmd))
{
    if (slide >= numSlides)
        beep.play();
    else
        slide++;
}
else
{
    if (slide <= 1)
        beep.play();
    else
        slide−−;
}
```

if-else-if

```
if (drinkSize.equals(”Large"))
{
    total += 1.39;
}
else  if (drinkSize.equals("Medium"))
{
    total += 1.19;
}
else   // if "Small" drink size
{
    total += 0.99;
}
```

Common if-else Errors

Extra semicolon:

```
if (...) ;
{
        statements
;
  ...
}
```

Missing braces:

```
if (...)
        statement 1;
        statement 2;
...
```

```
if (...)
   if (...)
      statement 1;
else
```

It is safer to always use braces in **if-else**

The switch Statement

switch
case
default
break

Reserved words

```
switch
   (expression)
{
   case value1:
        ...
        break;

   case value2:
        ...
        break;
   ...
   ...
   default:
        ...
        break;
}
```

Don't forget **break**s!

```
int x=10;
switch (x)
{
   case 1:
   c.println("too low");
   break;
   case 10:
   c.println("just right");
   break;
   default:
   c.println("dont know");
   break;
 }
```

```
if (x==1)
{
 c.println("too low");
 }
  else if (x==10)
 {
  c.println("just right");
  }
  else
  {
  c.println("don’t know");
   }
```

Review:

- What are the possible values of a boolean variable?
- What operators are used to compare values of numbers and chars?
- How can you test whether two Strings have the same values?
- Which binary operators have higher rank (are executed first), relational or logical?

Review (cont’d):

- Can you have an if statement without an else?
- What are De Morgan’s Laws?
- Explain short-circuit evaluation.
- How long can an if-else-if sequence be?