Boolean expressions

COMP 102

Victoria University of Wellington

Conditions

```
if ( 〈condition〉 ) {
   〈actions to perform if condition is true〉
      :
}
```

- Two possibilities, either something should happen, or it should not happen
  - In Computer Science we say that something is either true or false
  - A value that is either true or false is known as a Boolean value

Boolean expressions

- A Boolean value – a value that is either true or false.
- Boolean expressions:
  - constant values: true, false
  - numeric comparisons: (x > 0) (day <= 7), (x == y), (day != 7)
  - boolean method calls: month.equals("July") word.contains("th")
  - boolean variables: outlineOnly
    [if declared boolean outlineOnly;]
  - logical operators: !, &&, || (not, and, or)

```
( x > 0  &&  x < 7  && outlineOnly )
( month.startsWith("Ju") || month.equals("May") )
( ! fileModified ||  ! (cmd.equals("exit")) )
```

more methods on String
.equalsIgnoreCase("John”)
.startsWith(“Ab”)
.endsWith(“ies”)

Writing Boolean expressions

Mostly, boolean expressions are straightforward,

There are just a few traps:

- == is the "equals" operator for simple values,
  = is assignment

  (age == 15) *vs* ~~(age = 15);~~

  - But only use == for numbers (or characters, or references)

- Use the <u>equals</u> method for Strings, not ==
  (occasionally == will give the right answer by chance!)

  cur.equals("US") *vs* ~~cur == "US"~~

- String equality is case sensitive:

  “NZ".equals(“nz") → false

  “NZ".equalsIgnoreCase(“nz") → true

Boolean Variables

- A boolean value is a value!

  ⇒ it can be stored in a variable.

- Useful if the program needs to remember some option.
- Must declare the variable, and assign to it, before using it

```
boolean printSteps = UI.askBoolean("Print all steps?");

:

if ( printSteps )
    UI.println("Processed input");

:

if ( printSteps )
    UI.println("Computed Statistics");
:
```

Compound Boolean expressions: operators

Using logical operators:

Not: **!** eg (! currency.equalsIgnoreCase(“US"))

And: **&&** eg (x > 0 && x < 7 && outlineOnly)

Evaluates each conjunct in turn.
If any conjunct false, then value of whole expression is false
If all conjuncts true, then value of whole expression is true

Or: **||** eg (month.startsWith("Ju") || month.equals("May"))

Evaluates each disjunct in turn.
If any disjunct true, then value of whole expression is true
If all disjuncts false, then value of whole expression is false

Can combine into complicated expressions:

```
( ! fileModified   ||  (  cmd.equals("exit")  &&   lastSaveTime > 5000) )
```

safest to use lots of (…)

Traps with Boolean expressions

- When combining with && and ||, which binds tighter?

  if (x > 5 && y <= z || day == 0) { ….

  - Use (and) whenever you are not sure!

    if ((x > 5 && y <= z **)** || day == 0 **) {** …

    if (x > 5 && **(** y <= z || day == 0 **)) {** …

- The not operator ! goes in front of expressions:

  - **if (** !(x > 5 && y <= z **)** { …   **NOT**   **if (** (x !> 5 && y !<= z **)**
  - **if (** ! cur.equals("US") **)** { …   **NOT**   **if (** cur.!equals("US") **)** { …

  - **exception:**   **if (** ! (count == 0) **)** { …   *OR*   **if (** count != 0 **)** { …