Type: Set of values and the operations on them

- **Type int:**
  - **Values:** integers
  - **Ops:** +, −, *, /, %, *

- **Type double:**
  - **Values:** real numbers
  - **Ops:** +, −, *, /, *

- **Type boolean:**
  - **Values:** true and false
  - **Ops:** && (and), || (or), ! (not)

- **Type char:**
  - **Values:** single characters
  - **Ops:** +, −, *, /, %, *
  - **Essentially a number (?!)**

- **Type String:**
  - **Values:** string literals
  - **Ops:** + (concatenation)

Variables (p. 26)

- A variable is:
  - a named memory location (box),
  - a value (in the box), and
  - a type (limiting what can be put in box)

- Possible values for the variable `x`:
  - `int` with value 5
  - `double` with value 20.1

Variables (p. 26)

- **A declaration of a variable** gives the name of the variable and the type of value it can contain

  ```
  int x;
  double area;
  ```

- **Assignment Statement (p. 27)**

  ```
  x = x + 1;
  ```

  - This can get really messy

Objects: Organizing Data in Folders

- An object is like a manila folder
  - It contains other variables
    - These variables are called fields
    - Can change their values (with assignments)
  - It has a “tab” that identifies it
    - You cannot change this
    - Java assigns it automatically
    - More on this in demo later

Object Variables

- Variable stores object name
  - **Reference** to the object
  - Reason for folder analogy
- Use “dot” to access folder
  - Use `p.x` to access to field `x`
  - **Example:** `p.x = 3;`
- How do we create objects?
  - Other types have literals
  - **Example:** `1, "abc", true`
  - No such thing for objects
Object Initialization (the new keyword)

- `new Point3d()`
  - An expression (produces a value)
  - It creates an object (folder)
  - Value is the "tab name"
- `p = new Point3d();`
  - Assignment statement
  - Computes value
  - Stores value (tab name) in the variable `p`

Methods: Operations on Objects

- **Method**: instruction for an object
  - Similar to a function/procedure
  - But attached to an object
  - Can access all of object’s fields
- Use of a method is a **method call**
  - `<object-variable>.<method-call>`
  - Method calls end in parentheses
  - Values in parens are arguments
- Example: `p.getX()`
- Example: `p.setX(3.4)`

Packages and Built-in Classes

- Java has built-in classes
  - No need to compile them
  - But you have to import them
- Built-in classes are in **packages**
  - Use a command to import
    - `import <package>.<class>;`
    - `import <package>.*;`
      - imports everything in package
- Example: `JFrame`
  - Java class for (empty) Window
  - In package `javax.swing`

String is a Class!

```
String s = "Hello World";
```

String Methods

- Different from other classes
  - Do not create with new
- In package `java.lang`
  - Imported by default
  - Never need to import
- Great class to "play with"
  - All methods are functions
  - Use in interactions pane

- `charAt(int p)` Get letter at position p
- `substring(int p)` Get suffix starting at position p
- `substring(int p, int e)` Get suffix starting at position p, ending at e-1