Methods of Class Object

Methods of Class Object

Method	Description
boolean equals (Object obj)	**Decides whether two objects are meaningfully equivalent**
void finalize()	**Called by the garbage collector when the garbage collector sees that the object cannot be referenced**
int hashCode()	**Returns a hashcode int value for an object so that the object can be used in Collection classes that use hashing, including Hashtable , HashMap , and HashSet**

Methods of Class Object

Method	Description
final void notify()	**Wakes up a thread that is waiting for this object's lock**
final void notifyAll()	**Wakes up all threads that are waiting for this object's lock**
final void wait()	**Causes the current thread to wait until another thread calls notify() or notifyAll() on this object**
String toString()	**Returns a "text representation" of the object**

The toString() Method

- **Override toString() when you want to read something meaningful about the objects of your class.**
  - public class Test {
  - public static void main (String [] args) {
  - Test h = new Test();
  - System.out.println(h);
  - }
  - }

- **It gives object's state (in other words, the current values of the important instance variables).**
  - public String toString() {
  - return “return string”;
  - }

Equals() method

- When you really need to know if two references are identical, use == .
- But when you need to know if the objects themselves (not the references) are equal, use the equals() method.

Equals() method

- If you want objects of your class to be used as keys for a hashtable (or as elements in any data structure that uses equivalency for searching for—and/or retrieving—an object),
- then you must override equals() so that two different instances can be considered the same.

Implementing an equals() Method

- public boolean equals(Object o) {
- if ((o instanceof Student) && (((Student)o).getRollno()
- == this.rollno)) {
- return true;
- } else {
- return false;
- }
- }

Equals() method

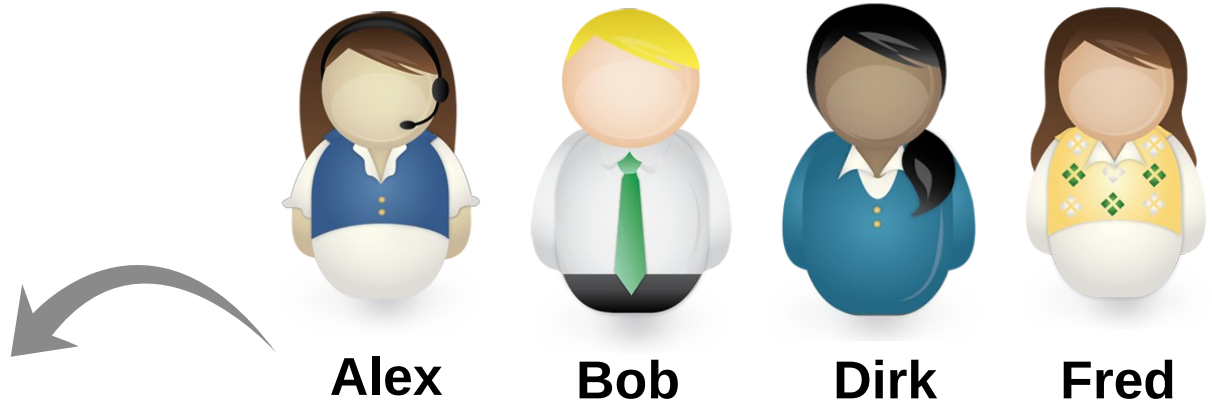
- If you don't override a class's equals() method,
- you won't be able to use those objects as a key in a hashtable.
- probably won't get accurate Sets such that there are no conceptual duplicates.

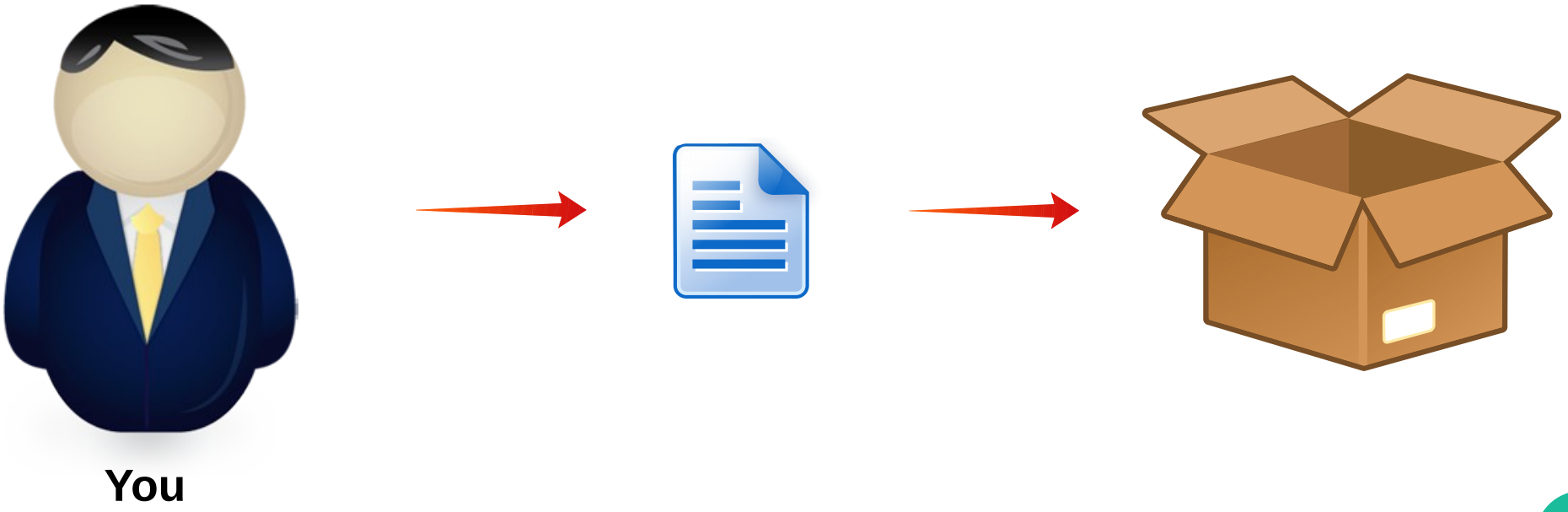
Overriding hashCode()

- Hashcodes are typically used to increase the performance of large collections of data.
- Although we can think of it as kind of an object ID number, it isn't necessarily unique.

- **Collections such as HashMap and HashSet use the hashcode value of an object to determine how the object should be stored in the collection,**
- **and the hashcode is used again to help locate the object in the collection.**

A simplified hashcode example

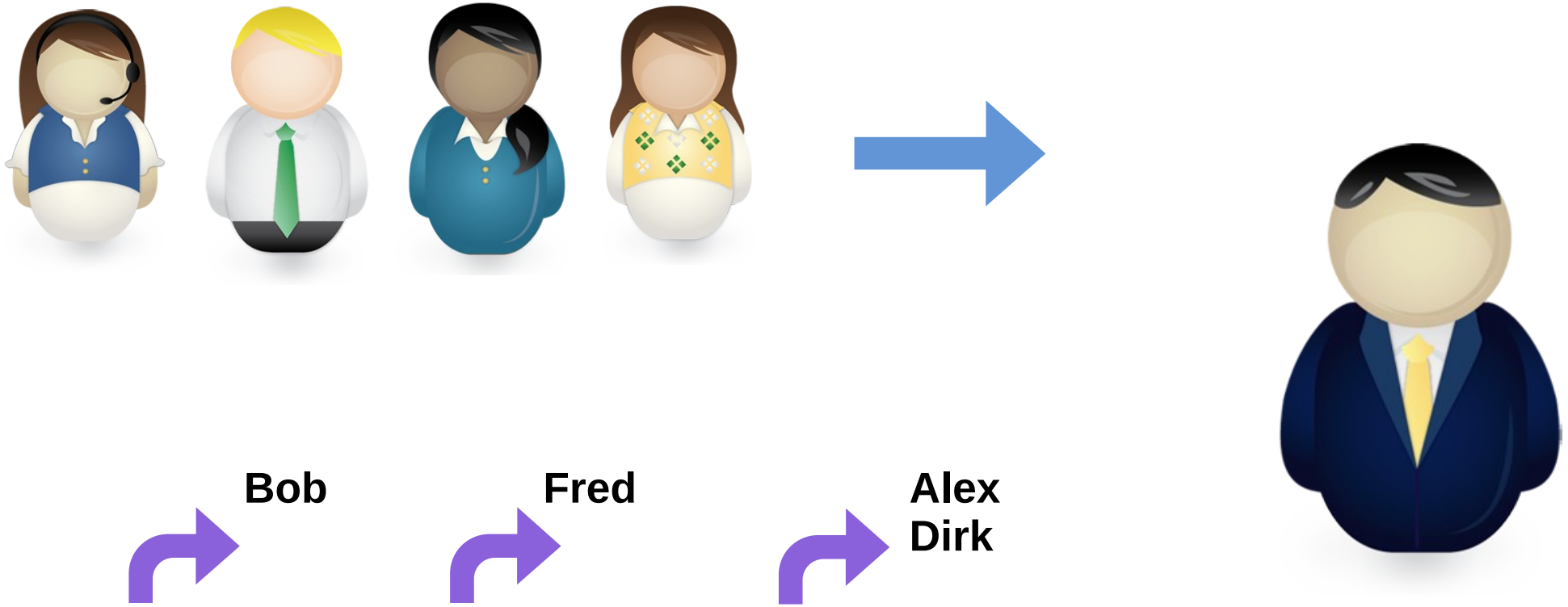




A simplified hashcode example

Key	Hashcode Algorithm	Hashcode
Alex	A(1) + L(12) + E(5) + X(24)	42
Bob	B(2) + O(15) + B(2)	19
Dirk	D(4) + I(9) + R(18) + K(11)	42
Fred	F(6) + R(18) + E(5) + D(4)	33

A simplified hashcode example



- **Hashing retrieval is a two-step process.**
  - Find the right bucket (using hashCode()).
  - Search the bucket for the right element (using equals()).
- **Keep variables non- transient or, if they must be marked transient , don't use them to determine hashcodes or equality.**

- **If two objects are considered equal using the equals() method, then they must have identical hashcode values.**