Testing for Equality in Java

- In Java, you use == to compare *primitives*
- For example:

```
//e will be set to true if 2 is equal to 3
boolean e = (2 == 3);
```

- And you use the method *x.equals(y)* to compare *Objects*
- For example:

```
//e is set to true if "thisString" is equal to "thatString"
boolean e = "thisString".equals("thatString");
```

Penn Engineering

1

Testing for Equality in Java

- Rule: When comparing a literal String value (known String value) to an unknown String value, use the *equals* method of the known value
- For example:

```
//e is set to true if "thisString" is equal to String value stored in
someUnknownString
boolean e = "thisString".equals(someUnknownString);
```

- Why?
  - Because you know, at least, that "thisString" exists and is not null, so it MUST have an *equals* method
  - *someUnknownString*, on the other hand could be null, in which case calling it's *equals* method will return an error

Penn Engineering

2

Testing for Equality in Java

- Why is all of this important?
  - The JUnit method *assertEquals(expected, actual)* uses == to compare *primitives* and *equals* to compare *Objects*
- To define *equals* for your own objects, you'll have to define *exactly* this method in your class:

```
public boolean equals(Object obj) { ... }
```

  - The argument must be of type Object, which isn't what you want, so you must cast it to the correct type (e.g. Person)

Penn Engineering

3

Testing for Equality in Java

- Here's a full (sample) implementation of *equals* inside a Person class

Person.java

```
public class Person {

    //Name of person
    String name;

    //Age of person
    int age;

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

//equals method to compare people
public boolean equals(Object something) {
    //cast Object to Person

    Person p = (Person) something;

    //compare names of each Person
    return this.name.equals(p.name);
}
}
```

- Two people are "equal" if they have the same exact name
- We'll talk more about implementing methods, like *equals*, later in the course

4

Testing for Equality in Java

- Here's how we compare people in our unit testing class

Person.java | PersonTest.java

```
import static org.junit.jupiter.api.Assertions.*;

class PersonTest {

    @Test
    void testPerson() {

        Person person1 = new Person("Ted", 22);
        Person person2 = new Person("Ted", 22);

        //assertEquals uses == to compare primitives
        //person 1 and person 2 have the same age
        assertEquals(person1.age, person2.age);

        //assertEquals uses .equals method to compare Objects
        //person 1 and person 2 ARE NOT equal because
        //they don't have the same exact name
        assertNotEquals(person1, person2);

        Person person3 = new Person("Ted", 34);

        //person 1 and person 3 ARE equal because
        //they have the same exact name
        assertEquals(person1, person3);
    }
}
```


5