

Javadoc utility

- ▶ Creates HTML documentation
- ▶ Comments of special form

```
/**
```

```
*/
```

- ▶ First sentence of each comment goes to summary table. It should begin with a capital letter and end with a period.
- ▶ A comment is supplied for each method and class
- ▶ Special keywords
 - @param
 - @return
- ▶ javadoc MyClass.java

2/20

Constants

Use keyword `final`

```
final double MY_CONST .7894;
```

```
private static final MY_CONST .784;
```

```
public static final MY_CONST .784;
```

```
double circumference = Math.PI * diameter;
```

1/20

Example (cont)

```
/**
```

```
  Add nickels to the purse.
```

```
  @param count the
```

```
  number of nickels to add
```

```
*/
```

```
public void addNickels(int count)
```

```
{
```

```
    nickels = nickels + count;
```

```
}
```

4/20

Example

```
/**
```

```
  A purse computes the total value
  of a collection of coins.
```

```
*/
```

```
public class Purse
```

```
{
```

```
    /**
```

```
      Constructs an empty purse.
```

```
    */
```

```
    public Purse()
```

```
{
```

```
    nickels = 0;
```

```
    dimes = 0;
```

```
    quarters = 0;
```

```
}
```

3/20

Example (cont)

```
/**
 * Add quarters to the purse.
 * @param count the number
 *   of quarters to add
 */
public void addQuarters(int count)
{
    quarters = quarters + count;
}
```

6/20

Example Continued

```
private static final double
    NICKEL_VALUE = 0.05;
private static final double
    DIME_VALUE = 0.1;
private static final double
    QUARTER_VALUE = 0.25;
private int nickels;
private int dimes;
private int quarters;
}
```

8/20

Example (cont)

```
/**
 * Add dimes to the purse.
 * @param count
 *   the number of dimes to add
 */
public void addDimes(int count)
{
    dimes = dimes + count;
}
```

5/20

Example Continued

```
/**
 * Get the total value of
 *   the coins in the purse.
 * @return the sum of all coin values
 */
public double getTotal()
{
    return nickels * NICKEL_VALUE
        + dimes * DIME_VALUE
        + quarters * QUARTER_VALUE;
}
```

7/20

Example

```
import javax.swing.JOptionPane;
public class InputTest
{
    public static void main(String[] args)
    {
        Purse myPurse = new Purse();
        String input JOptionPane.showInputDialog(
            "How many nickels do you have?");
        int count = Integer.parseInt(input);
        myPurse.addNickels(count);
        String input JOptionPane.showInputDialog(
            "How many dimes do you have?");
        count = Integer.parseInt(input);
        myPurse.addDimes(count);
    }
}
```

10/20

Reading Input

JOptionPane from javax.swing

```
String input =
    JOptionPane.showInputDialog("Give number");
int count = Integer.parseInt(input);
System.exit(0);
```

9/20

Type Conversion

```
double total;
int pennies = (int) total;

double price = 44.95;

int dollars = (int) (price + .5);

int dollars = (int) Math.round(100 * F);
```

12/20

Example

```
String input JOptionPane.showInputDialog(
    "How many quarters do you have?");
count = Integer.parseInt(input);
myPurse.addQuarters(count);
double totalValue = myPurse.getTotal();
System.out.println("The total is " + totalValue);
System.exit(0);
```

11/20

Methods

```
class BodyPrint {
    public static void main(String[] args) {
        Body sun = new Body("Sol", null);
        Body earth = new Body("Earth", sun);
        System.out.println("Body " +
            earth.name +
            "orbits " +
            earth.orbits.name +
            "and has ID " +
            earth.idNum);
    }
}
```

14/20

Method Execution and Return

```
public class Permissions {
    public boolean canDeposit,
        canWithdraw,
        canClose;
}
```

16/20

Static Initialization

```
class Primes {
    static int[] knownPrimes = new int[4];
    static {
        knownPrimes[0] = 2;
        for (int i = 1; i <
            knownPrimes.length; i++)
            knownPrimes[i] = nextPrime();
    }
    // declaration of nextPrime
}
```

13/20

Method Invocations

reference.method(arguments)

```
public String toString() {
    String desc = idNum + " (" + name + ")";
    if (orbits != null)
        desc += " orbits " + orbits.toString();
    return desc;
}
```

15/20

Parameter Values

```
class PassByValue {
    public static void main(String[] args) {
        double one = 1.0;
        System.out.println("before: one = " + one);
        halveit(one);
        System.out.println("after: one = " + one);
    }

    public static void halveit(double arg) {
        arg /= 2.0; // divide arg by two
        System.out.println("halved: arg = " + arg);
    }
}
```

18/20

Method Execution and Return (cont)

```
public class BankAccount {
    private long number; //account number
    private long balance; //current balance
    public Permissions
        permissionsFor(Person who) {
            Permissions perm = new Permissions();
            perm.canDeposit = canDeposit(who);
            perm.canWithdraw = canWithdraw(who);
            return perm;
        }
    // .. define canDeposit et al...
}
```

17/20

Using Method to Control Access

```
class Body {
    private long idNum;
    public String name = "<unnamed>";
    public Body orbits = null;
    private static long nextID = 0;
    Body () {
        n        idNum = nextID++;
    }
    public long getID() {
        return idNum;
    }
    // .....
}
```

20/20

Parameter Vals (cont)

```
class PassRef {
    public static void main(String[] args) {
        Body sirius = new Body("Sirius", null);
        System.out.println("before: " + sirius);
        commonName(sirius);
        System.out.println("after: " + sirius);
    }

    public static void commonNameet(Body bodyRef) {
        bodyRef.name = "Dog Star";
        bodyRef = null;
    }
}
```

19/20

Cont

```

public String getName ()
{ return name; }
public void setName (String newName) {
    name = newName;
}
public Body getOrbits ()
{ return orbits; }
public void setOrbits (Body orbitsAround) {
    orbits = orbitsAround;
}
}

```

22/20

Cont

```

class Body {
    private long idNum;
    public String name = "<unnamed>";
    public Body orbits = null;
    private static long nextID = 0;
    // Constructors
    public long getID () {
        return idNum;
    }
}

```

21/20

Console Input

- ▶ reads from System.in object which only reads bytes.
- ▶ An InputStreamReader reads characters


```

InputStreamReader reader =
    new InputStreamReader (System.in);

```
- ▶ A BufferedReader can read strings.


```

BufferedReader console =
    new BufferedReader (reader);

```

24/20

this

```

public Body (String name, Body orbits) {
    this ();
    this.name = name;
    this.orbits = orbits;
}

```

23/20

Example (cont)

```

public static void main(String[] args)
    throws IOException
{
    Purse myPurse = new Purse();
    BufferedReader console = new BufferedReader(
        new InputStreamReader(System.in));

    System.out.println
        ("How many nickels do you have?");
    String input = console.readLine();
    int count = Integer.parseInt(input);
    myPurse.addNickels(count);
}

```

26/20

Example

```

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

/**
 * This program tests input
 * from a console window.
 */
public class ConsoleInputTest
{

```

25/20

Example (cont)

```

    System.out.println
        ("How many dimes do you have?");
    input = console.readLine();
    count = Integer.parseInt(input);
    myPurse.addDimes(count);
    System.out.println
        ("How many quarters do you have?");
    input = console.readLine();
    count = Integer.parseInt(input);
    myPurse.addQuarters(count);
    double totalValue = myPurse.getTotal();
    System.out.println
        ("The total is " + totalValue);
    System.exit(0);
}

```

27/20