Lesson 34: String Conversions (W10D3)

Balboa High School

Michael Ferraro

October 21, 2015

Do Now

For each problem, determine the value of a.

---

1. `int a = 9 / 2;`
2. `double a = 9 / 2;`
3. `double a = 9.0 / 2.0;`
4. `double a = 9.0 / 2;`
5. `int a = 9.0 / 2;`
6. `double a = 9 / (double)2;`
7. `double a = 2 * 3 / 5 - ( 1 / 10 );`

Aim

Students will learn how to convert numerical values into `String` objects and start the *Solving Quadratic Equations* problem set exercise.

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;`

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;`

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;`

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;` 4.5

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;` 4.5
4. `double a = 9.0 / 2;`

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;` 4.5
4. `double a = 9.0 / 2;` 4.5

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;` 4.5
4. `double a = 9.0 / 2;` 4.5
5. `int a = 9.0 / 2;`

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;` 4.5
4. `double a = 9.0 / 2;` 4.5
5. `int a = 9.0 / 2;` error!

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;` 4.5
4. `double a = 9.0 / 2;` 4.5
5. `int a = 9.0 / 2;` error!
6. `double a = 9 / (double)2;`

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;` 4.5
4. `double a = 9.0 / 2;` 4.5
5. `int a = 9.0 / 2;` error!
6. `double a = 9 / (double)2;` 4.5

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;` 4.5
4. `double a = 9.0 / 2;` 4.5
5. `int a = 9.0 / 2;` error!
6. `double a = 9 / (double)2;` 4.5
7. `double a = 2 * 3 / 5 - ( 1 / 10 );`

Do Now Solutions

For each problem, determine the value of a.

---

1. `int a = 9 / 2;` 4
2. `double a = 9 / 2;` 4.0
3. `double a = 9.0 / 2.0;` 4.5
4. `double a = 9.0 / 2;` 4.5
5. `int a = 9.0 / 2;` error!
6. `double a = 9 / (double)2;` 4.5
7. `double a = 2 * 3 / 5 - ( 1 / 10 );` 1.0

Turning ints into Strings

- Create a new project in workspace1 called Lesson34
- Enter this code for the IntToString class:

```
public class IntToString {

    public static void main(String[] args) {
        int a = 5;
        String str = a;
        System.out.println("str = " + str);
    }

}
```

- When you try to compile and run IntToString, you'll notice a problem. How might you fix it?

Turning ints into Strings

One approach: Create an `Integer` object

```
public class IntToString {

    public static void main(String[] args) {
        int a = 5;

        Integer myInt = new Integer(a);
        String str = myInt.toString();

        System.out.println("str = " + str);
    }
}
```

This is too much work!

Turning ints into Strings

Better approach:
Concatentate the `int` with the empty string, `""`

```
public class IntToString {

    public static void main(String[] args) {
        int a = 5;

        String str = "" + a;

        System.out.println("str = " + str);
    }
}
```

This works for `floats` *and* `doubles`*, too!*

Turning ints into Strings

It turns out that `System.out.println()` does this for you; we've been sending it `int`s as its parameter for a long time now.

```
System.out.println("" + a);
```

$\approx$

```
System.out.println(a);
```

Evaluation Meets `String` Conversion

- Predict the value of `strOne` in each case:
  1. `String strOne = "" + 5 + 3;`
  2. `String strOne = 5 + "" + 3;`
  3. `String strOne = 5 + 3 + "";`
- Show your predictions to a neighbor.
- Alter your current class' `main()` to test your predictions!

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne = "" + 5 + 3;`

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne =` `"" + 5 + 3`;

    `"" + 5 + 3`

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne =` `"" + 5 + 3`;

    `"" + 5 + 3`

    `"" + 5` + 3

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne = "" + 5 + 3;`

```
"" + 5 + 3

"" + 5 + 3

"5" + 3
```

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne = "" + 5 + 3;`

```
"" + 5 + 3

"" + 5 + 3

"5" + 3

"53"
```

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne =` `5 + "" + 3`;

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne =` `5 + "" + 3`;

    `5 + "" + 3`

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne =` $\boxed{\texttt{5 + "" + 3}}$`;`

    `5 + "" + 3`

    $\underline{\texttt{5 + ""}}$ `+ 3`

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne = 5 + "" + 3;`

```
5 + "" + 3

5 + "" + 3

"5" + 3
```

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne = 5 + "" + 3;`

```
5 + "" + 3

5 + "" + 3

"5" + 3

"53"
```

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne =` `5 + 3 + ""` `;`

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne = 5 + 3 + "";`

    `5 + 3 + ""`

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne = 5 + 3 + "";`

    `5 + 3 + ""`

    `5 + 3 + ""`

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne = 5 + 3 + "";`

```
5 + 3 + ""

5 + 3 + ""

8 + ""
```

Evaluation Meets `String` Conversion

- Recall that Java evaluates the RHS of an assignment in a left-to-right manner
- Explanation:
  - `String strOne = 5 + 3 + "";`

```
5 + 3 + ""

5 + 3 + ""

8 + ""

"8"
```

Every Object has `toString()`

- You may recall that we've added a `toString()` method to classes in the past in order to send them to `System.out.println()`[1]

[1] In PS #2 and the *Polymorphism* lesson, the code for `BankAccount` and its subclasses had `toString()` methods so that we could simply print the account object to get details to display on the screen.

[2] `toString()` methods you write in a class *override* the default one that all objects inherit.

Every Object has `toString()`

- You may recall that we've added a `toString()` method to classes in the past in order to send them to `System.out.println()`[1]
- When an object doesn't have a programmer-supplied `toString()` method, the default one is used, which prints out the memory location of the object.[2]

[1] In PS #2 and the *Polymorphism* lesson, the code for `BankAccount` and its subclasses had `toString()` methods so that we could simply print the account object to get details to display on the screen.

[2] `toString()` methods you write in a class *override* the default one that all objects inherit.

Every Object has `toString()`

- You may recall that we've added a `toString()` method to classes in the past in order to send them to `System.out.println()`[1]
- When an object doesn't have a programmer-supplied `toString()` method, the default one is used, which prints out the memory location of the object.[2]
- You will read more about this in Litvin §5.10.

[1] In PS #2 and the *Polymorphism* lesson, the code for `BankAccount` and its subclasses had `toString()` methods so that we could simply print the account object to get details to display on the screen.

[2] `toString()` methods you write in a class *override* the default one that all objects inherit.

Next. . .

- Begin working §5.1 of PS #5, *Solving Quadratic Equations*.
  - Follow all instructions carefully! You'll do an initial push to GitHub today and update your GitHub repository as you make progress.
  - You may have some class time over the coming lessons to continue working on this program.
- Move on to HW slide. . .

HW

- Finish §4 of PS #5.
- Continue working on the programming exercises in §5 of PS #5. These could be time-consuming if you don't read everything carefully — and ask questions when you're stuck!