

Ask the Object

Calling instance methods



What comes before the dot?

PAUSE AND TRY

You have written all three of these. What is the thing just before each dot?

```
int age = console.nextInt();
```

Is it a class? A variable? An object? Something else?

```
int size = name.length();
```

```
System.out.println("hi");
```

By the end of class you can...

- 1 Call an instance method on an object using the dot operator.
- 2 Use the value an instance method returns, and pass it the right arguments.
- 3 Explain why the same call can give different results for different objects.
- 4 Predict what happens when a method is called on null.

1

PART 1

The dot operator

objectName.methodName(arguments)

```
int size = word.length();
```

word

The object you are asking. Here, a String.

.

The dot operator joins the object to its method.

length

The instance method to run on that object.

()

The arguments. length needs none.

An instance method is called on an object, with the object's name, a dot, and the method.

Same method, different objects

```
String a = new String("frog");  
String b = new String("hopper");  
System.out.println(a.length());  
System.out.println(b.length());  
System.out.println(b.indexOf("p"));
```

OUTPUT

4
6
2

Call	Asks	Value
a.length()	frog	4
b.length()	hopper	6
b.indexOf("p")	hopper	2

The object before the dot decides **whose** data the method uses.

An instance method needs an object



Put an **object** before the dot: `city.length()`, not the class name, and not nothing at all.

NO OBJECT

```
int n = length(city);
```

JAVA SAYS

```
error: cannot find symbol
  symbol: method length(String)
```

A CLASS, NOT AN OBJECT

```
int n = String.length();
```

JAVA SAYS

```
error: non-static method length()
cannot be referenced from a
static context
```

A large, bold, teal number '2' is centered within a white rounded square that has a slight 3D effect and a shadow.

PART 2

Arguments and return values

The Topic 1.9 rules still apply

```
Scanner console = new Scanner(System.in);
int age = console.nextInt();
String city = console.next();
System.out.println(age + 1);
System.out.println(city.indexOf("ver"));
```

Header	So the call
int nextInt()	returns an int
String next()	returns a String
int indexOf(String str)	needs one String

THE USER TYPES

15 Denver

OUTPUT

16
3

Match the **signature**, and **store or use** the return value.

Each object keeps its own data

SCOREBOARD, A MADE-UP CLASS

```
Scoreboard()           // a new score starts at 0
void addPoints(int points) // adds points to the score
int getScore()          // returns the score
```

```
Scoreboard home = new Scoreboard();
Scoreboard away = new Scoreboard();
home.addPoints(3);
away.addPoints(7);
System.out.println(home.getScore());
System.out.println(away.getScore());
home.addPoints(2);
System.out.println(home.getScore());
```

OUTPUT

```
3
7
5
```

void call: stands alone. **Non-void** call:
store it or use it.

Two names, one scoreboard

PAUSE AND TRY

What prints? How many Scoreboard objects are there?

```
Scoreboard s1 = new Scoreboard();  
Scoreboard s2 = s1;  
s2.addPoints(4);  
System.out.println(s1.getScore());
```

Hint: draw a box for each object and an arrow from each variable.

A white rounded square with a teal number '3' inside, tilted slightly to the right.

3

PART 3

When there is no object

A call on null crashes

```
String city = null;  
System.out.println("before");  
System.out.println(city.length());  
System.out.println("after");
```

OUTPUT

before

JAVA SAYS

```
Exception in thread "main"  
java.lang.NullPointerException: Cannot invoke  
"String.length()" because "<local1>" is null
```



A method call on a **null** reference results in a **NullPointerException**. It compiles, then crashes while running.

Compile error or crash?

Code	When it fails	Why
<code>city.length</code>	compile time	missing (): length is a method
<code>String.length()</code>	compile time	no object before the dot
<code>city.indexOf(4.5)</code>	compile time	indexOf has no double version
<code>city.length()</code>	run time, if city is null	no object to ask

Compiles **does not** mean it works. The compiler checks types; only running shows what a variable holds.

Print it, or crash?

PAUSE AND TRY

For each one: what prints, or does it crash?

A

```
String s = null;  
System.out.println(s);
```

B

```
String s = null;  
System.out.println(s.length());
```

C

```
String s = new String();  
System.out.println(s.length());
```

D

```
Scoreboard b = null;  
b.addPoints(1);
```

A large, white, rounded square containing a bold, dark teal number '4'.

4

PART 4

Putting it together

Two objects, three calls

```
Scanner console = new Scanner(System.in);
String word = console.next();
int size = word.length();
int firstE = word.indexOf("e");
System.out.println(size);
System.out.println(firstE);
```

Call	Object
console.next()	the Scanner
word.length()	the String
word.indexOf("e")	the String

THE USER TYPES

breeze

OUTPUT

6
2

Two bugs, two lines

PAUSE AND TRY

This program does not compile. Find and fix the two mistakes.

```
1 Scanner console = new Scanner(System.in);  
2 String word = console.next;  
3 int size = length(word);  
4 System.out.println(size);
```

It should print the length
of the word the user types.

Words to know

Instance method

A method called on an object of its class.

Dot operator

Joins an object's name to the method being called.

Object reference

What a variable holds: which object to use.

null

A reference that points to no object at all.

NullPointerException

The crash when a method is called on null.

Return value

What a non-void call gives back to store or use.

Who are you asking?



Explain to your partner what the object before the dot does, and what goes wrong when it is null.

SENTENCE STARTER

The object before the dot _____, so when the variable is null, _____.

Next: the Ask the Object puzzle worksheet.