

The logo for Oracle Academy is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by a thin black border, with dark gray horizontal bars at the top and bottom.

# ORACLE

## Academy

# Java Fundamentals

**2-14**

## Java Methods and Classes

**ORACLE**  
Academy



Copyright © 2022, Oracle and/or its affiliates. Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

# Objectives

- This lesson covers the following objectives:
  - Describe a method, class, and instance
  - Describe a scenario where an IF control structure would be used
  - Describe a scenario where a WHILE control structure would be used
  - Recognize the syntax for a method, class, function, and procedure
  - Describe input and output



# Alice 3 Versus Java

Alice 3	Java
3D programming environment that uses visual representations for the Java programming language	A programming language with syntax that can be edited using an integrated development environment (IDE)
Used to create animations or interactive games while working with programming constructs	Used to create applications that run on any platform, including the web, using Java syntax
Drag and drop interface designed to reduce syntax errors and make it easier to learn how to program	IDE helps you model real world objects, allows for re-use as well as easier maintenance

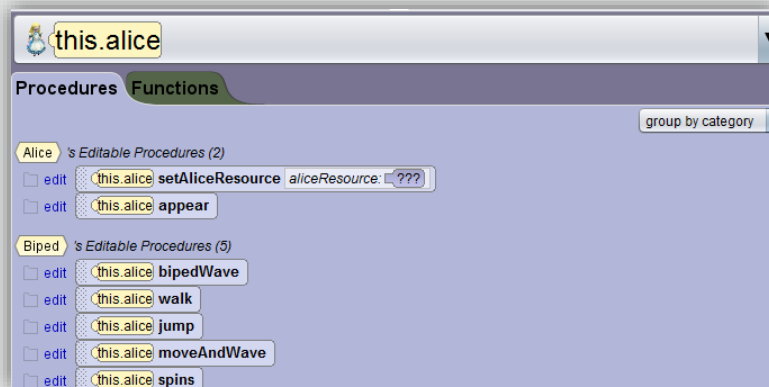
## Procedures in Alice 3

- In Alice 3, a procedure is a piece of code that sends a message to an object asking it to perform an action
- A procedure does not return a value
- A set of procedures are available for each class

Procedures are what is used in Alice 3 to make our objects carry out an action.

# Methods Panel in Alice 3

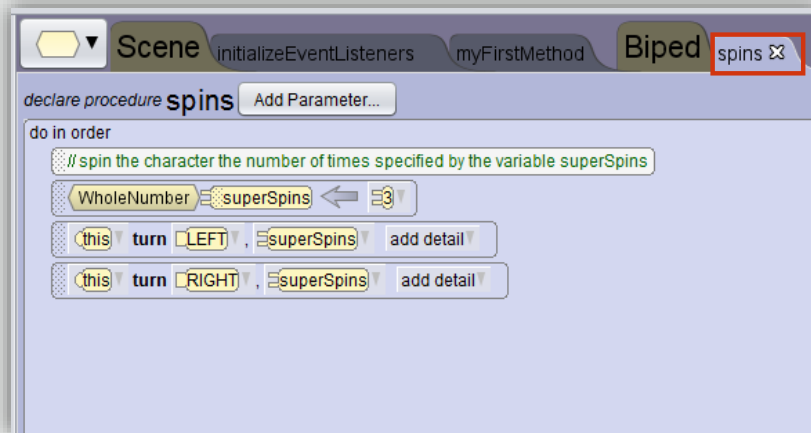
- The methods panel lists all of an object's procedures and functions
- It contains all built-in and user-defined procedures and functions available for each object in your animation



The methods panel gives you procedures and functions relevant to the object that you picked from the drop down list.

# Declaring Procedures in Alice 3

- You can declare (create) your own procedures in Alice 3



You should create your own procedures if you have identified multiple objects that will carry out the same action or if you are writing the same code on more than one occasion.

# Methods in Java

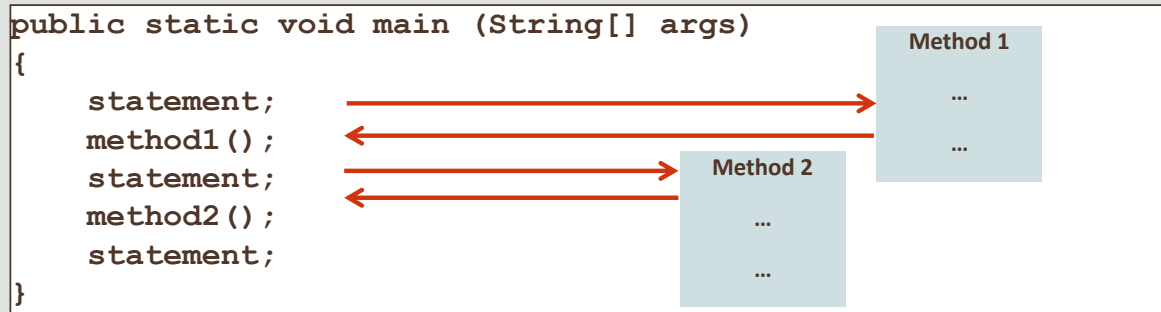
- Methods in Java are the same as procedures in Alice 3
- A method is a piece of code that sends a message to an object asking it to perform an action
- Methods:
  - Belong to a class
  - Are referred to by name
  - Can be called at any point in a program using the method's name
- When a method name is encountered in a program, it is executed

When you move on to Greenfoot and Eclipse all of your code will be described as methods. You will have void methods (procedures) that carry out actions and non-void methods (functions) that return information.



# Methods in Java Example

- When the method is finished, execution returns to the area of the program code from which it was called, and the program continues on to the next line of code



Method calls are executed in the same way as when we use functions or user defined procedures in Alice 3.

## Decisions for Each Method

- There are three decisions to make for any method:
  - What the method should do
  - What inputs the method needs
  - What answer the method gives back
- Java syntax for a method:

```
[modifiers] dataType methodName(parameterList) {  
    //methodBody  
    return result;  
}
```

The modifier, data type, name and parameter list is known as the method signature in Java.

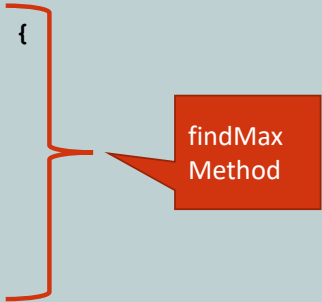
# Method Properties

Method Property	Description
modifiers	These are optional and can be public, private, protected, or left blank
dataType	Type of data, like int
methodName	Name of your method
parameterList	Comma-separated list of parameter names with their data types; each parameter is listed with its data type first, then its name
methodBody	Set of statements that perform the task
return	Keyword that sends the result value back to the code that called the method

All of these terms will be fully explained in the future sections of this course. For now just try to be aware of what they are used for.

# findMax() Method

```
public class TestFindMax {  
    /** Main method */  
    public static void main(String[] args) {  
        int i = 5;  
        int j = 2;  
        int k = findMax(i, j);  
        System.out.println("The maximum between " + i + " and " + j +  
            " is " + k);  
    } //end method main  
  
    /** Return the max between two numbers */  
    public static int findMax(int num1, int num2) {  
        int result;  
        if (num1 > num2)  
            result = num1;  
        else  
            result = num2;  
        //endif  
        return result;  
    } //end method findMax  
}
```



**ORACLE**  
Academy

JF 2-14  
Java Methods and Classes

Copyright © 2022, Oracle and/or its affiliates. Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

12

When looking at the code in this section try to identify the sections that you are familiar with. In this example there is some variable declaration (`int i = 5;`), a method call (`findMax(i, j);`) and an If Else statement. You have used all of these statements in Alice 3.

## Classes in Alice 3

- A Dalmatian is a dog
- When a Dalmatian object is added to a scene, it has the properties of the Dalmatian class:
  - four legs, two ears, a white and black spotted coat, and the ability to walk

A class is a specification, such as a blueprint or pattern, of how to construct an object.



**ORACLE**  
Academy

JF 2-14  
Java Methods and Classes

Copyright © 2022, Oracle and/or its affiliates. Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

13

A class is the code that defines what information to store about an object and what that object can do. A class becomes an object when we give it values. A dog is a class, but a Dalmatian is an object or "instance" of the dog class because it is a dog with specific values.

# Classes in Java

- The first group:
  - Optional; refers to the visibility from other objects
  - public means visible everywhere
  - The default is package or visible within the current package only

```
["public"] ["abstract"|"final"] "class" Class_name  
  ["extends" object_name] ["implements" interface_name]  
"{  
  // properties declarations  
  // behavior declarations  
"}"
```

The following few slides are quite technical, don't worry about fully understanding. The information will be covered in more depth later in the course.

# Classes in Java

- The second group:
  - Optional; defines whether the class can be inherited or extended by other classes
  - Abstract classes must be extended and final classes can never be extended by inheritance
  - The default indicates that the class may or may not be extended at the programmers discretion

```
[ "public" ] [ "abstract" | "final" ] "class" Class_name  
    [ "extends" object_name ] [ "implements" interface_name ]  
"{"  
    // properties declarations  
    // behavior declarations  
"}"
```

# Classes in Java

- Class\_name is the name of the class
- The third option of extends is related to inheritance
- The fourth option of implements is related to interfaces

```
[ "public" ] [ "abstract" | "final" ] "class" Class_name  
    [ "extends" object_name ] [ "implements" interface_name ]  
"{  
    // properties declarations  
    // behavior declarations  
}"
```



# Code for Creating Cat Class in Java

```
class Cat{
    int catAge;
    String catName;

    public Cat(String name){
        catName = name;
    } //end of constructor

    public void setAge(int age){
        catAge = age;
    } //end method setAge

    public int getAge(){
        return catAge;
    } //end method getAge

    public static void main(String []args){
        Cat myCat = new Cat("Garfield");
        myCat.setAge(6);
        System.out.println("Cat age: " + myCat.getAge());
    } //end method main
} //end class Cat
```

**ORACLE**  
Academy

JF 2-14  
Java Methods and Classes

Copyright © 2022, Oracle and/or its affiliates. Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

17

This class has two variables for the name and age of the cat. It also has two methods, one that allows you to set the age of the cat and one that gives us the age of the cat. Every variable in a class should have a set and get method attached to it.

## Instances in Alice 3

- When you add an object to a scene, this creates an instance of the class

An object is an instance of a class.



**ORACLE**  
Academy

JF 2-14  
Java Methods and Classes

Copyright © 2022, Oracle and/or its affiliates. Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

18

You can have multiple instances of the same class within a scene. In this example there is one cat instance however there are multiple instances of the playing card. All instances of a class will have the same behaviour.

# Instances in Java

- In the main method, the myCat instance of the cat class is created

```
class Cat{
    int catAge;
    String catName;
    ...
    ...
    ...
    public static void main(String []args){
        Cat myCat = new Cat("Garfield");
        myCat.setAge(6);
        System.out.println("Cat age: " + myCat.getAge());
    } //end method main
} //end class Cat
```

You create an instance of a class by using the new statement in Java.

# Create Instance of a Class

- An instance of a class is created using the new operator, followed by the class name

```
class Cat{
    int catAge;
    String catName;
    ...
    ...
    ...
    public static void main(String []args){
        Cat myCat = new Cat("Garfield");
        myCat.setAge(6);
        System.out.println("Cat age: " + myCat.getAge());
    } //end method main
} //end class Cat
```

Create new instance

# Control Structures

- Control structures allow you to change the order of how the statements in your programs are executed
- Both Alice and Java allow for these types of control structures

Type	Description	Example
Decision Control Structures	Allow you to select specific sections of code to be executed	if ... then ... else
Repetition Control Structures	Allow you to execute specific sections of code a number of times	while loop

You accessed the control structures in Alice by dragging them into the code editor bar at the bottom the screen.

## if Control Structure

- if control structures are statements that allow you to select and execute specific blocks of code while skipping other sections
- These structures have the following form

```
if (boolean_expression) {  
    doSomething();  
}  
else {  
    doSomethingElse();  
} //endif
```

An If statement executes if the condition is true. Remember the Else statement is optional and does not need to be included.

## if Control Structure Example

- For example:
  - If a student receives a score of **90%** or greater on their test, then give them an **"A"**
  - If a student receives a score that is greater than or equal to **80%** and less than **90%**, then give them a **"B"**
  - If a student receives a score that is greater than or equal to **65%** and less than **80%**, then give them a **"C"**
  - If a student receives less than **65%**, then give them an **"F"**

## if Control Structure Example Code

```
public class Grade {  
    public static void main( String[] args ){  
        //variable declaration section  
        double grade = 89.0;  
  
        if( grade >= 90 ){  
            System.out.println( "A" );  
        }  
        else if( (grade < 90) && (grade >= 80) ){  
            System.out.println("B" );  
        }  
        else if( (grade < 80) && (grade >= 65) ){  
            System.out.println("C" );  
        }  
        else{  
            System.out.println("F");  
        } //endif  
    } //end method main  
} //end class Grade
```

In this code once a true condition has been discovered the program will stop checking the nested Else If statements. This is the purpose of nesting them within the Else statement.



## while Control Structure

- A while control structure, or while loop, is a Java statement or block of statements that allows you to execute specific blocks of code repeatedly as long as some condition is satisfied
- The condition is tested before each loop
- while loop format:

```
while( boolean_expression ){  
    statement1;  
    statement2;  
    . . .  
}
```

## while Control Structure Example

```
class WhileDemo {  
    public static void main(String[] args){  
        //variable declaration section  
        int count = 1;  
  
        while (count < 11) {  
            System.out.println("Count is: " + count);  
            count++;  
        } //end while  
  
    } //end method main  
} //end class WhileDemo
```

In a while loop you need to ensure that the terminating condition can be met or else you will end up in an infinite loop. That is why the value of count is increased by one every time the code loops. That way once the value of count is no longer less than eleven the condition will be false and the loop will terminate.

# Input and Output

- Java programs work on many platforms
- They can be simple programs that run from the command line, or they can have complex graphical user interfaces
- When learning to program, you will create programs that use the command line for its input and output exclusively
- With more experience, you can build graphical user interfaces and learn about the libraries required to build them

# Input and Output Code

- Print to the screen example

```
System.out.println("Hello World!");
```

- `System.out.println()` is a piece of code that is automatically available to each and every Java program

This is similar to the Say or Think procedure in Alice 3.

# Input and Output Code Example

- User input example using the java.util.Scanner package

```
import java.util.Scanner;
public class ReadString {
    public static void main (String[] args) {
        System.out.print("Enter your name: ");
        Scanner in = new Scanner(System.in);
        String userName = null;
        userName = in.nextLine();
        System.out.println("Thanks for the name, " + userName);
    } //end method main
} //end class ReadString
```

Here an object of the Scanner class is being used to allow input. Most programs will require some interaction with the user so input and output is very important. In Alice there were the getWholeNumber, getDecimalNumber etc. functions that performed the input.

# Terminology

- Key terms used in this lesson included:
  - Class
  - Control structure
  - IF control structure
  - Instance
  - Method
  - WHILE control structure

# Summary

- In this lesson, you should have learned how to:
  - Describe a method, class, and instance
  - Describe a scenario where an IF control structure would be used
  - Describe a scenario where a WHILE control structure would be used
  - Recognize the syntax for a method, class, function, and procedure
  - Describe input and output



