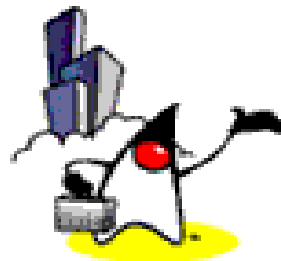




Command-line Arguments

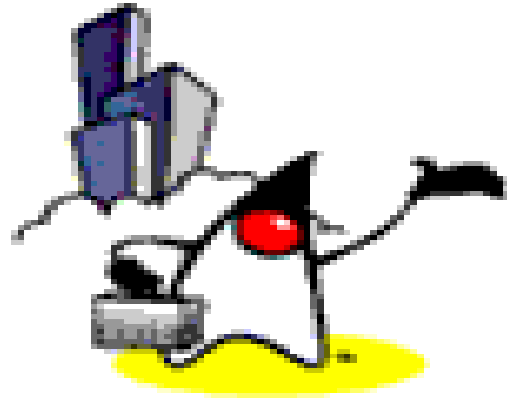


Objectives

At the end of the lesson, the student should be able to:

- Know and explain what a command-line argument is
- Get input from the user using command-line arguments
- Learn how to pass arguments to your programs in NetBeans





Command-line Arguments

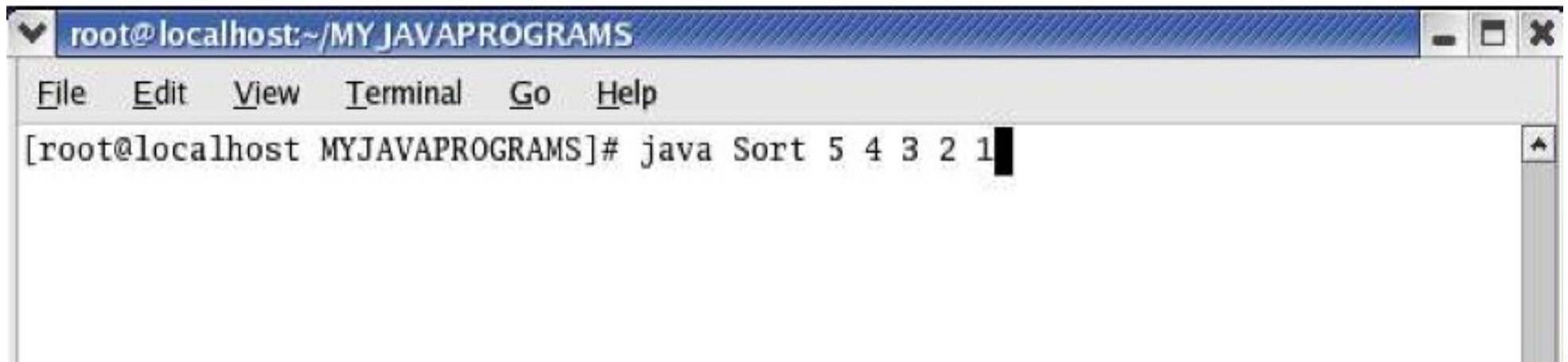
Command-line Arguments

- A Java application can accept any number of arguments from the command-line.
- Command-line arguments allow the user to affect the operation of an application.
- The user enters command-line arguments when invoking the application and specifies them after the name of the class to run.



Command-line Arguments

- For example, suppose you have a Java application, called Sort, that sorts five numbers, you run it like this:



```
root@localhost:~/MYJAVAPROGRAMS
File Edit View Terminal Go Help
[root@localhost MYJAVAPROGRAMS]# java Sort 5 4 3 2 1
```

- Note: The arguments are separated by spaces.



How Java Application Receive Command-line Arguments

- In Java, when you invoke an application, the runtime system passes the command-line arguments to the application's main method via an array of Strings.

```
public static void main( String[] args )
```

Each String in the array contains one of the command-line arguments.

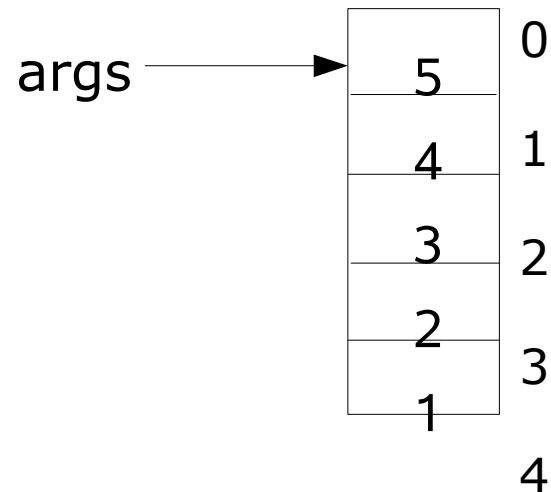


args[] array

- Given the previous example where we run:

```
java Sort 5 4 3 2 1
```

the arguments are stored in the `args` array of the main method declaration.



Example

- To print the array of arguments, we write:

```
1 public class CommandLineSample
2 {
3     public static void main( String[] args ){
4
5         for(int i=0; i<args.length; i++){
6             System.out.println( args[i] );
7         }
8
9     }
10 }
```



Conversion of Command-line Arguments

- If your program needs to support a numeric command-line argument, it must convert a String argument that represents a number, such as "34", to a number.
- Here's a code snippet that converts a command-line argument to an integer,

```
int firstArg = 0;
```

```
if (args.length > 0){  
    firstArg = Integer.parseInt(args[0]);  
}
```

- the parseInt() method in the Integer class throws a NumberFormatException (ERROR) if the format of args[0] isn't valid (not a number).

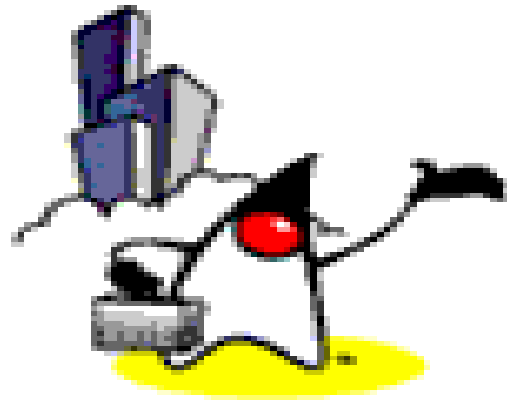


Command-line Arguments: Coding Guidelines

- Before using command-line arguments, always check the number of arguments before accessing the array elements so that there will be no exception generated.
- For example, if your program needs the user to input 5 arguments,

```
if( args.length!= 5 ){  
    System.out.println("Invalid number of arguments");  
    System.out.println("Please enter 5 arguments");  
}  
else{  
    //some statements here  
}
```





How to pass Command-line Arguments in NetBeans?

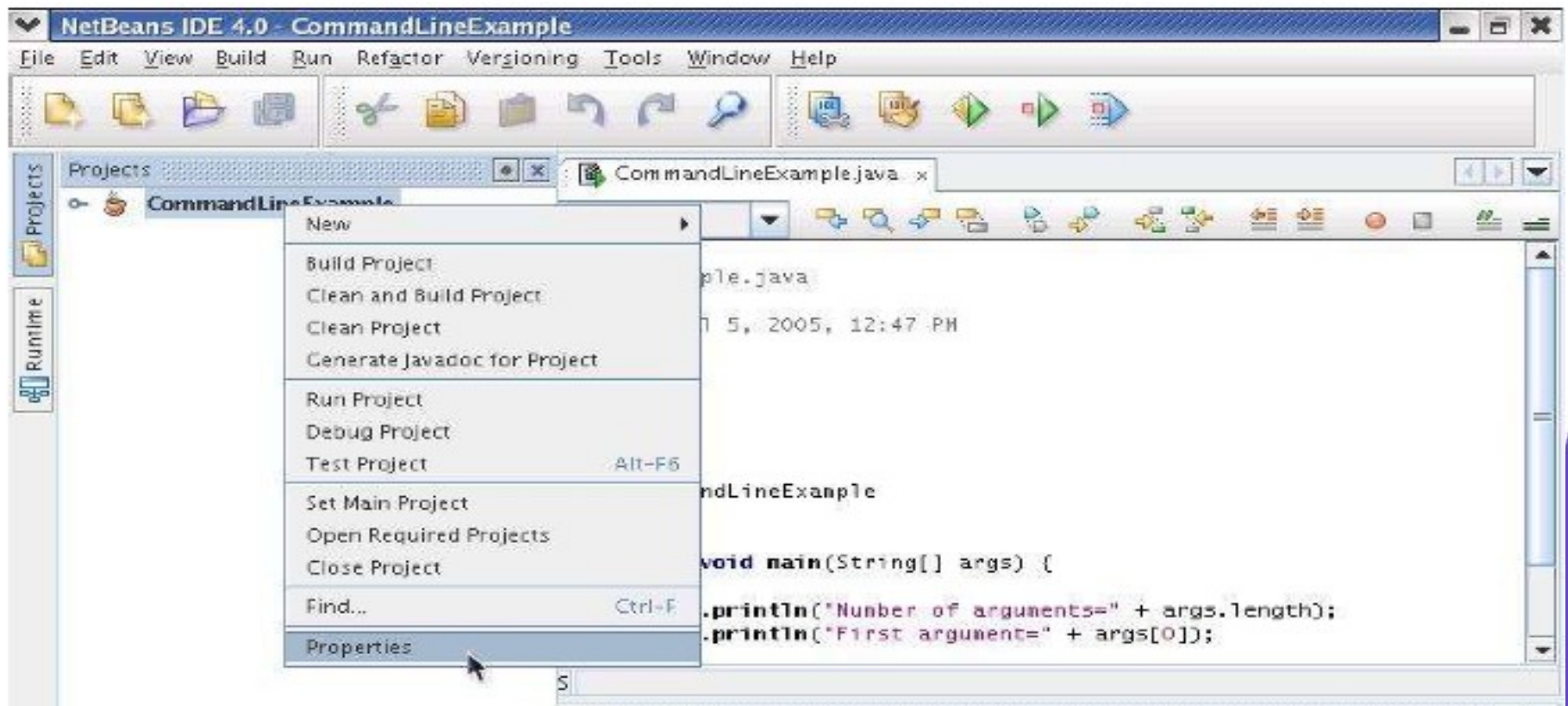
Command-line Arguments in NetBeans

- Now, how can we pass command-line arguments in NetBeans?
- Assuming you already have a project and you have compiled it successfully, follow the steps to pass command-line arguments in NetBeans



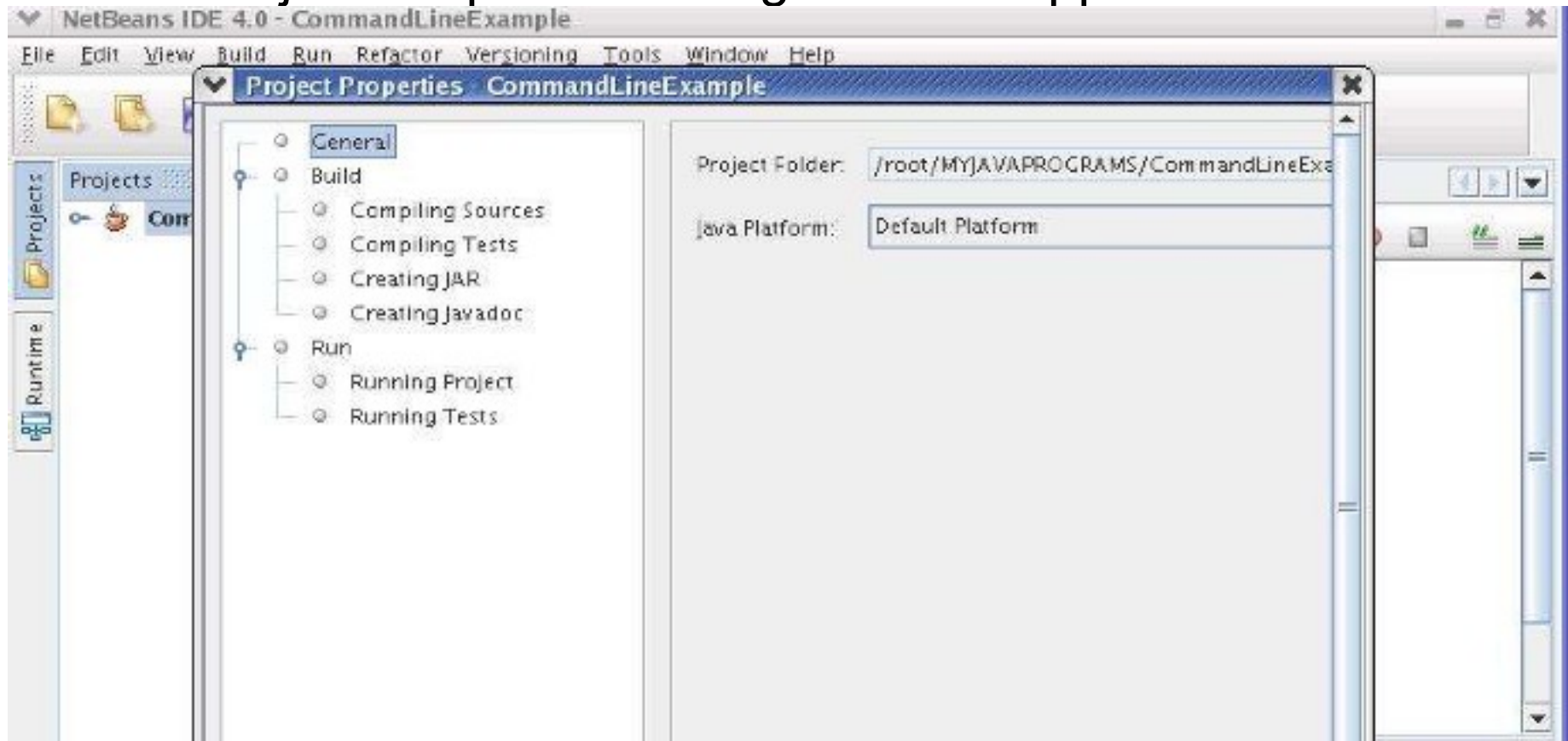
How to pass Command-line Arguments in NetBeans

- Right-click the project node, and a popup menu will appear.
- Click on Properties.



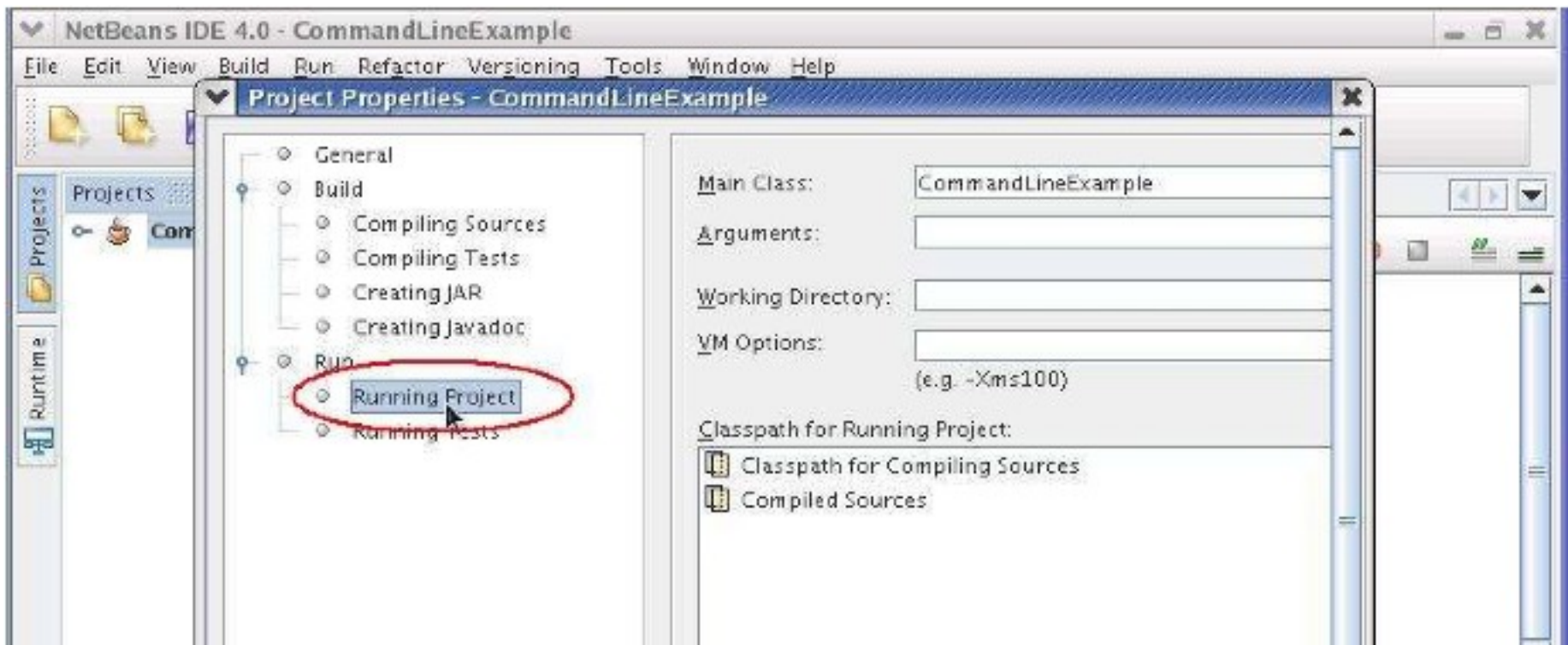
How to pass Command-line Arguments in NetBeans

- The Project Properties dialog will then appear.



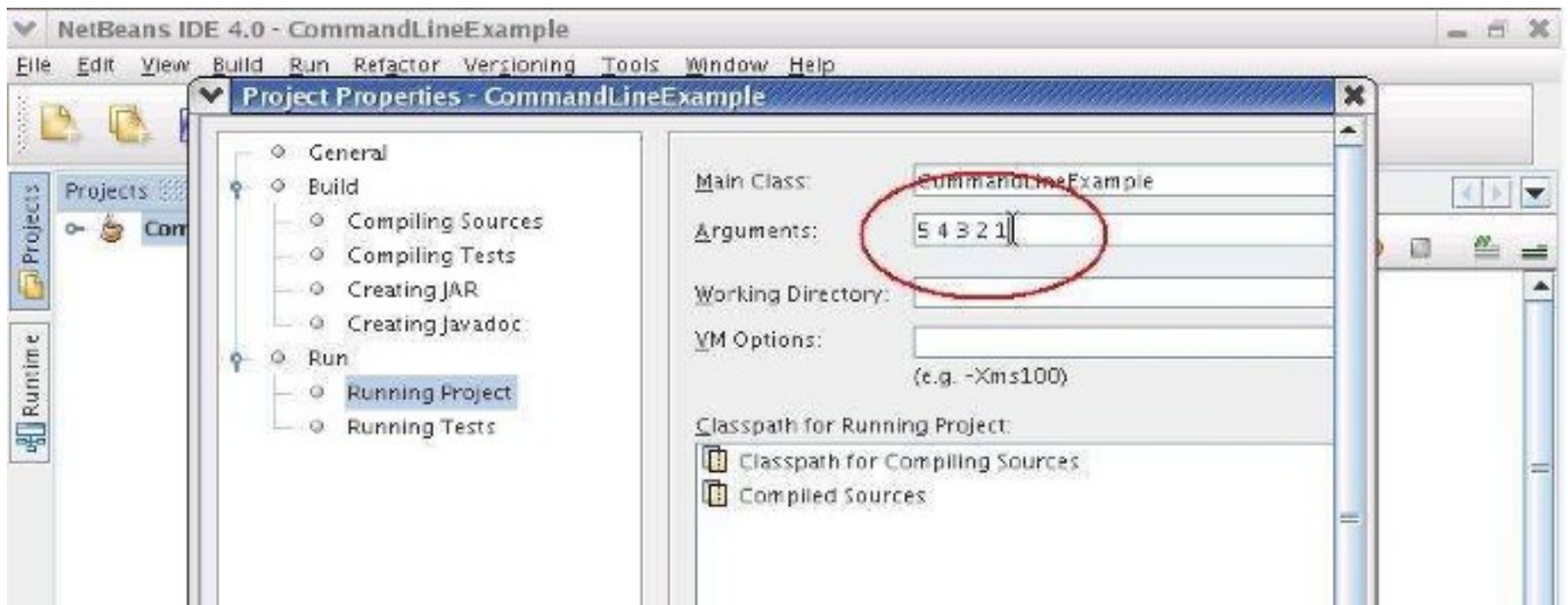
Command-line Arguments in NetBeans

- Now, click on Run-> Running Project



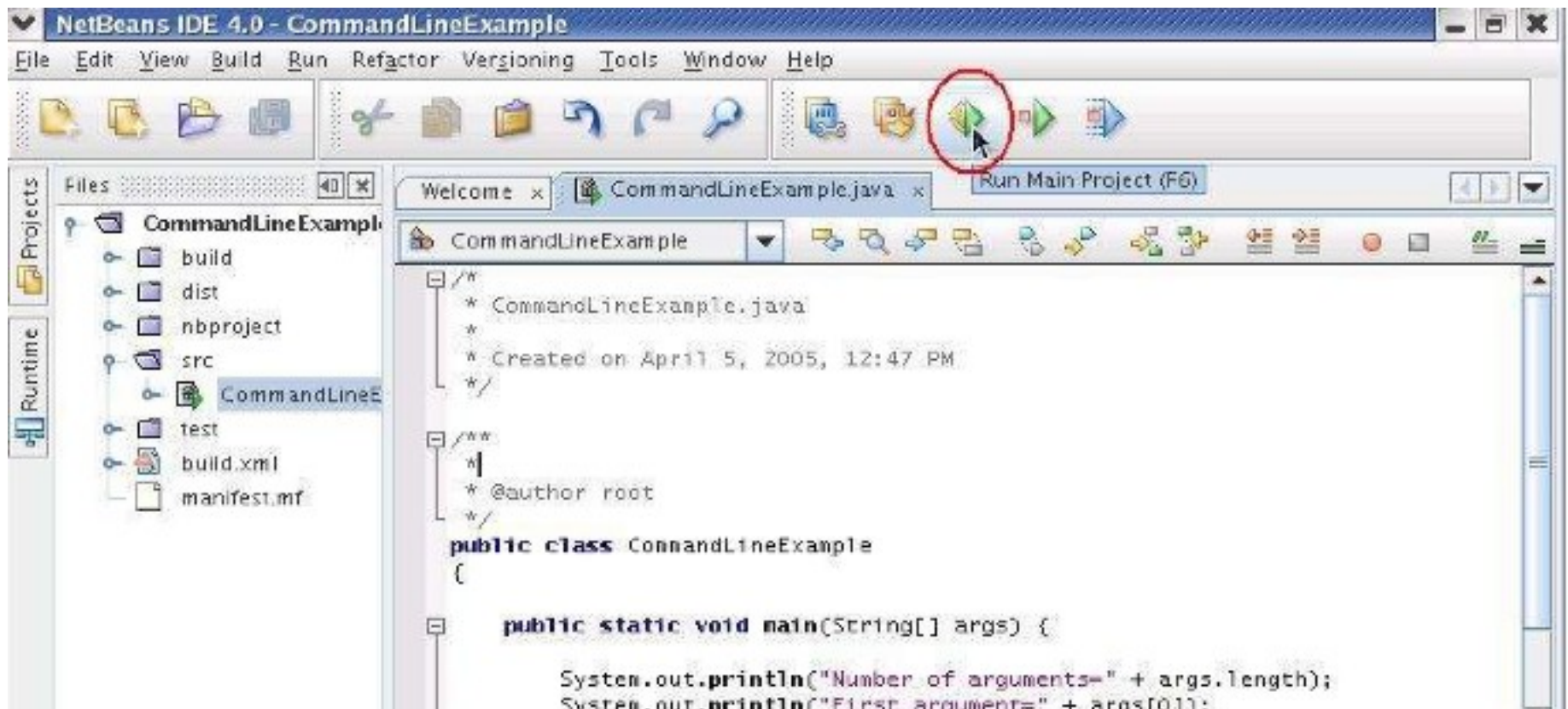
Command-line Arguments in NetBeans

- On the Arguments textbox, type the arguments you want to pass to your program. In this case we typed in the arguments 5 4 3 2 1. Then, click on the OK button.



Command-line Arguments in NetBeans

- Now try to RUN your program.



Summary

- Command-line arguments
 - How to access the arguments
 - How to convert String arguments to integer using Integer.parseInt method
 - How to pass command-line arguments in NetBeans

