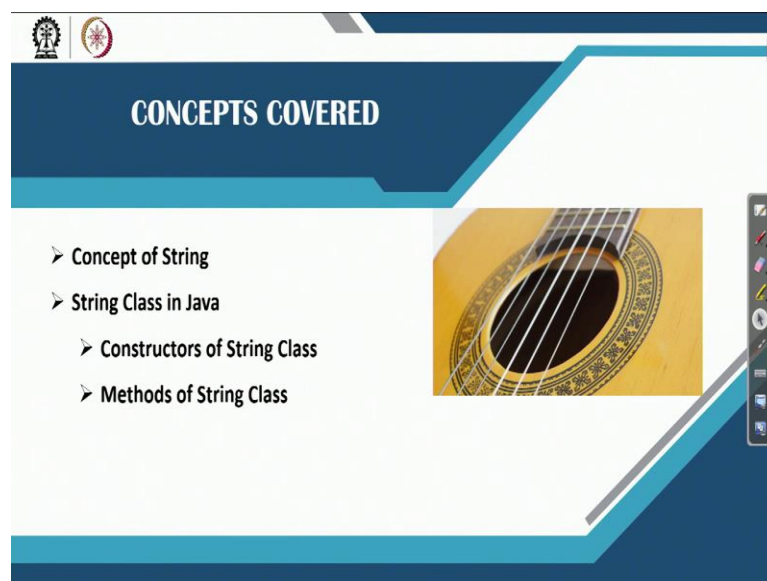


Data Structures and Algorithms Using Java
Professor. Debasis Samanta
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Lecture 56
String Class

So, this is the last week of this course. In this last week, we will cover one important module, this module is very important for many application developer. So, we will cover the String and utilities, these are the two things are very important in Java programming, but the topics is very past.

So, it is not so adequate to cover everything, whatever it is there in string and utilities, but for the sake of basic knowledge and understanding, I will try to cover most of the important things which are there in studying and utilities.

(Refer Slide Time: 01:07)



So, in today's lecture we will cover the concept of string and the string class in Java. So, in order to learn the string class in Java, we have to cover what are the different constructors, so that the string object can be instantiated and the different methods in String class by which the sting objects can be manipulated. So, this is the topic that we have planned in this video lecture.

(Refer Slide Time: 1:41)

The slide features a central title "Concept of String" in blue text. To the left is an icon of a coffee cup with steam. The background is decorated with various icons: gears, a tree with circular nodes, an atom, a hard hat, and a circuit board. A small video inset of a presenter is in the bottom right corner. The footer includes the NPTEL logo and the text "NPTEL Online Certification Courses IIT Kharagpur".

So, let us have the first concept of string. Now, this string is an important concept in Java programming language. This is because, whatever with a data that Java internally process it basically see as a string possibly you know, what exactly a string is. In C and C++ programming, this concept of string is not there, it is a very unique concept which is only known in Java programming language. Now, so, let us learn what is basically a string is in Java.

(Refer Slide Time: 02:20)

The slide is titled "What is string?". It features an icon of a rope. Below the icon is a bulleted list defining a string and its handling in C/C++ and Java. An "Example:" section shows a Java code snippet for a character array. A small video inset of a presenter is in the bottom right corner. The footer includes the NPTEL logo and the text "NPTEL Online Certification Courses IIT Kharagpur".

- A sequence of characters (e.g., alphabetic, alpha-numeric, or any printable characters, such as !, @, #, \$ % &, +, ?, /, etc.) is called string.
- In C, C++ programming languages, a string is managed as an array of data type `char`.
- Java also allows the same.

Example:

```
char aSet[10] = {'a', 'e', 'i', 'o', 'u', ' ', ' ', 's', 't'};  
System.out.println(aSet); // It will print aciou s#
```

What is string?

- A sequence of characters (e.g., alphabetic, alpha-numeric, or any printable characters, such as !, @, #, \$ % &, +, ?, /, etc.) is called string.
- In C, C++ programming languages, a string is managed as an array of data type `char`.
- Java also allows the same.

Example:

```
char aSet[10] = {'a', 'e', 'i', 'o', 'u', '\n', '\t', '5', 'M'};
System.out.println(aSet); // It will print aeiou +
```

NPTEL Online Certification Courses
IIT Kharagpur

So, a string is basically a sequence of character that characters may be from the set of alphabets, set of numeric or any other sets of printable characters that we can see on the keyboard like the different symbols like punctuation symbols, the mathematical operator etc. So, a sequence of characters is basically is represented as a string in Java programming.

Now, in C or C++ programming language, we also handle a sequence of characters, we usually store all the sequence of characters in a array of type `char`, carries a primitive data type which is known in C, C++ programming language. Now, in Java also a primitive data type carries there and Java also sees the same way as C and C++ sees, but Java deals this sequence of characters in a more details way rather or more efficient way, which is in the form of string actually.

Anyway, so a character as you know, if we want to represent in programming, So, this basically enclosed within a single quote. For example, here as we see is a set of characters, a set of characters, which are here is stored in this array and this array is a character, array of characters. Now, so this basically a set, a set is also in that sense sequence of characters. So, we can say it is also a string, it also can be as a termed as a string.

Now, a string can be printed in Java using simple `println` statement as we see here. So, this is basically we are printing a string, so it will print this is the string. So, this basically is a string we can say in this example. So, in a basic sense, a string is nothing but a sequence of characters that characters from any set of alphabets or numerics or any printable characters, which we can think of.

(Refer Slide Time: 05:10)

What is string?

- As Java is a true object-oriented programming language, and hence, it treats a string from a more greater perspective.
 - It treats string as an object!

Example:

- All the following literals are actually objects in Java:

"NPTEL",
"Data Structures",
"Joy with Java"

NPTEL Online Certification Courses
IIT Kharagpur

Now, let us see what are the different way that Java can see the string, other than the area of sequence basically. Now Java mainly object oriented programming language and it basically treats everything as a string for the more general perspective. In fact, Java treats a string is not a primitive data like, a character array, it basically treat it as an object and you know, so object and data they are different.

So, data is only the data it contains certain value, on the other hand objects is basically encapsulation of data and what are the methods that we can operate on these data. So, this whole constitute objects. So, here in Java a string is in fact more than the array of characters it is basically an object and this is the object of the class stream. Now, here are a few popular familiar example of string that I have mentioned here.

For example, this is a string, this is also another string and this is also another string and possibly you will note that here a string is something is not any array of, sequence of what is called characters, but they are enclosed within a double quote and in fact, these are the basically two three strings that we have and they are treated as a constant, I will tell you, why they are called constant, you know possibly constant.

So, x equals 123. So, here x is a variable whereas 123 is a constant. So, in that sense, these are the string objects we can say, but this basically string objects are constant actually, that means you will not be able to change them that is important property that it has, we will not be able to change means, we will not be able to delete any characters from it or you will not be able to insert any characters into it or you will not be able to replace these things by any

other strings like this one in that sense, it is called constant. Now, let us come to the more details discussion about the concept of string.

(Refer Slide Time: 07:35)

String in Java

- Java defines a class called **String** in **java.lang** package. It allows to create and manipulate **String** objects.

Examples:

```
String myString = "Joy with Java";  
System.out.println(myString);  
System.out.println("Welcome");  
System.out.println("Welcome" + " " + myString);  
String aString = "An example of string is " + myString;
```

NPTEL Online Certification Courses
IIT Kharagpur

So, string as I told you, it is basically a class, these classes is defined in java.lang package, the java.lang is a default package, which basically to be imported for any programs that you want to run. So, is basically the class string is there, if you want to create an object of type string, then you have to take the help of this class string.

Now, here are a few examples that you can think for how we can create some string objects. So as I told you, so if we want to create a string object, let the name of the object, this is the reference to the object, my string and it is the, it is of type class string and this is the, this is the string object itself, that is the value of the string, my string in that case and we can print like this one.

So, this is also a string that can be directly although we did not create any what is called initialized an object for this, but we can do it also and here also you can see this is also a string, This thing is basically merging of the string and this string. So, this is also a string here, this plus operation is basically a concatenation of two string objects and this is also another example of concatenation, this is another string and this is also string.

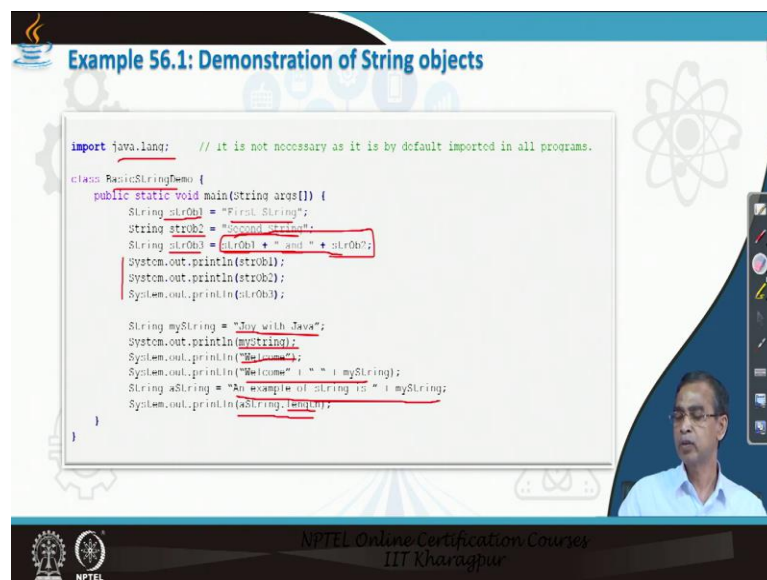
Now, so far the length of a string is concerned in Java, theoretically there is no limit, your string can be of any size, absolutely no problem that and after concatenation it will resolve a bigger string actually. So this is a new thing that can be created. For example, here astring is

a new string, which is basically combination of this string and this string, where my string is basically this string.

So this is these are the few examples that is easy to understand about how we can create an object string. It is basically in a usual manner, the way we create a variable or identifier or any primitive data type.

So we can say many people, many students confused about that string possibly is a data type, it is not a data type, it is basically a class with a class is defined in java.io package, but this class behaves like a primitive datatype, it is because it is so frequent it is so useful it is so visible in many programs, so that is why the Java developer made it as the usual primitive datatype like. Now, let us proceed for that, so, that we can understand few more things about the string.

(Refer Slide Time: 10:33)



The slide displays a Java code snippet within a presentation window. The code demonstrates string concatenation and printing. The title of the slide is "Example 56.1: Demonstration of String objects". The code is as follows:

```
import java.lang; // it is not necessary as it is by default imported in all programs.

class BasicStringDemo {
    public static void main(String args[]) {
        String strObj1 = "First String";
        String strObj2 = "Second String";
        String strObj3 = strObj1 + " and " + strObj2;
        System.out.println(strObj1);
        System.out.println(strObj2);
        System.out.println(strObj3);

        String myString = "My with Java";
        System.out.println(myString);
        System.out.println(myString);
        System.out.println(myString);
        System.out.println("Welcome" + " " + myString);
        String aString = "An example of string is " + myString;
        System.out.println(aString);
    }
}
```

Now, I want to give an illustration using the string objects and this is a simple program for you, you can follow this program. So, I am giving you the basic string demo in this program, my objective is to create some string and then do certain operations mainly the merging operation and print them.

Now, here is basically as you can see, we create the three string objects, these are the three string object, these basically initialize by this string, this is another string object initialized by string and this string object is basically a concatenation of these, these and these So, it is basically combination of three string is this one and therefore, these are the official usual statement to print all three string that we have created just now.

We created another string printing the string, this is the example already we have given in the last slide. So, it is like this and here you can see, if you want to know what is the size of the screen string, size of the string means the number of constituent character in it, including blank space.

So, this basically there is a method defined in the class string, which is defined here, the method is basically length method, we can call this method this way also, the `asString.length` basically returns the length of the characters that is present in this string, here it is like this. So, you can run this program and you can see how you can get output and you can play this program writing whatever the way you want to manipulate your string, you can see it is very interesting, in fact and we will cover a few more interesting property of the string in next few slides.

(Refer Slide Time: 12:17)

String in Java

Notes:

- String is probably the most commonly used class in Java's class library.
- Java String is a powerful concept because everything is treated as a string in Java.
- The obvious reason for this is that strings are a very important part of programming.
- Every string you create is actually an object of type String.
- Even string constants are actually String objects.
- In Java, objects of String are immutable which means a constant and cannot be changed once created.
- Strings in Java are always represented with double quotes.

string - x = x + 'A'
x

NPTEL Online Certification Courses
IIT Kharagpur

Now, let us come to consolidation about the string fast. So, string is possibly the most commonly used class which is defined in Java APIs the name of the API `java.lang`. Now, this is basically created as a very powerful one concept, because in Java, everything is treated as a string, I can tell you one example, you probably recall the command line input, in case of command line input, we can enter some data into your running program while you are invoking the program execution.

So, there you have to pass that input, now you have to type, so this is a command line input. Now, this command line input stored in an string array, so possibly you can define the main method where one argument is string args and an array. So, it basically arrays of string objects.

So, whatever the input you supply during running the program, I mean invoking the program that time all the input is basically goes to the array of strings and you know, all input if you type 12.34 which is basically according to your understanding, is a floating point value, but it will go to the program via a string objects and later on from the program, you have to parse the string object to get its actual value.

So, this basically the idea that Java programming follows is called everything is treated as a string. That means all data whether it is simple character, a Boolean or a integer number or even a some set of text everything this physically created this thing that is a very important what is called a formalism that is there in Java in fact.

Now, so that is why so, now there are benefits of doing these things, because string something is basic characters and which makes the platform independence in any programming. This is because in some architecture integer may be four bytes in another architecture integer may be two bytes.

So, platform what is called a heterogeneity is there, but in order to make it homogeneous, so that your program can run across any platform. That means it can take input in your machine which consider an input at two bytes or in another machine which consider as a four bytes, no issue because we can treat them as this string and later on, we can parse this thing into that target machine language.

So, this way platform independency can be ensured and this is why the string is very important one concept that Java programmer thought from very beginning. Now, every string that you can create is usually an object of type string that I have told you, because it is an object oriented programming language, if I say Java is a pure object oriented programming language, then in that sense also it is true, because it keeps everything does like an object.

In fact, we have also faced that any numbers say integer or float or long it also can be treated as an object and for which there are many wrapper methods are known, which are defined in `java.util`, where this wrapper method is basically wrapped an integer value into a string. And then later on this or you can say that wrapper object, wrapper methods rather or wrapper classes rather we can say they basically treats everything as an object.

Now, so they are, so seeing basically the constant are actually objects I told you why it is constant because it cannot be changed. So, that is it the next important property is called strings are immutable. So, this means that you cannot write a string like for example, it is also

invalid say suppose, if you write a string this kind of concept is not valid in Java programming that because of this so string, so you declare x is a string and this is equal to x plus x plus, so that means here adding A to x where x is string is not possible.

So, because it is basically not possible, which is just like the changing the x by setting something this is not possible in Java, that is why if this is because they are basically immutable it is like this if you declare x is 1234 is a constant and 123 is equals to 123 plus something which is impossible.

So, this in that sense it is called immutable means you cannot change it is a constant and here are strings in Java as we have already observed few example that they basically enclosed within double quotes, if you write something within double quotes it basically considered as a string which is in contrast to something if you write within a single quote, one character like okay then basically it returns you that ASCII value of that character or integer basically.

In Java, it is not exactly ASCII value, it basically the Unicode value of that character I will discuss about Unicode concept later on. So, this is these are the few important part is called the properties that the string holds.

(Refer Slide Time: 18:09)

The slide displays a Java class named `CommandLineDemo` with a `main` method that takes an array of strings as input. The code prints the number of arguments, checks if there are any, and then iterates through each argument to print its index and value. Below the code, a terminal window shows the program being run with four different sets of arguments: no arguments, numbers 1-4, the text "Joy with Java", and "Joy with Java 2020".

```
class CommandLineDemo {
    public static void main(String args[]) {
        System.out.println("Number of arguments" + " " + args.length);
        if(args.length == 0)
            return;
        for(int i=0; i<args.length; i++)
            System.out.println("args[" + i + "]: " + args[i]);
    }
}
```

Hint: Run the program with following:

```
java CommandLineDemo
java CommandLineDemo 1 2 3 4
java CommandLineDemo Joy with Java
java CommandLineDemo Joy with Java 2020
```

Now, here, I told you one example about command line arguments I am giving one demo, so that you can understand, how every data, any data basically can be clustered into the form of a string and from there you can retrieve your own data as usual. For example here, so this is a program simple program, you see command line demo and here you see main and we first give an argument, this argument is basically is an array of strings.

Now here, this is a `println` statement number of arguments, then how many arguments that you have passed this is basically `args.length` is basically, what is the total number of elements in this array that also can be printed in this way. So, this basically printing what is the total number of element that is stored in this array, this one.

Now, if `args.length` equals to 0, then return it will not do anything, otherwise it will print all the arguments that you have entered. So, this is a very simple program. Now, let us come to the execution of this program. So, this is the program if you successfully compile, so it will produce a class file, the name of the class file is `command line demo`, to run this program, this is a command that you have to use `Java` command and this is the name of the program that you want to invoke.

Now, here you see this is basically now so here `args`, array basically does not contains any elements, whereas this basically 0, 1, 2 and 3, so these are the four arguments that we have passed. So all arguments are basically in this case as an example, but it will print as all these 1, 2, 3, 4 in the sequence one line by line. Now here again, this is the argument, this is another argument this is another argument.

So, these are the three arguments separated by space. So, the command line interpreter `Java` virtual machine will consider this is one input, this is another input, this is another input, so three input passed. So, it will print like in one line your next line with next line `Java`. Now, here again you can see the same thing, but one more and you can see that here input can be of any type and this one, so it can print.

Now, here we can see the input can be passed of any type, maybe integer or any sequence of characters or anything like whatever it is not an issue, but there is a statement by which you can parse, if you want to parse one elements as an integer. So, usually you can use `Integer.parseInt` the `args[0]` or `args[1]` or whatever it is there and it will obviously give errors if you do not parse it properly, if it is an integer you should parse it as an integer.

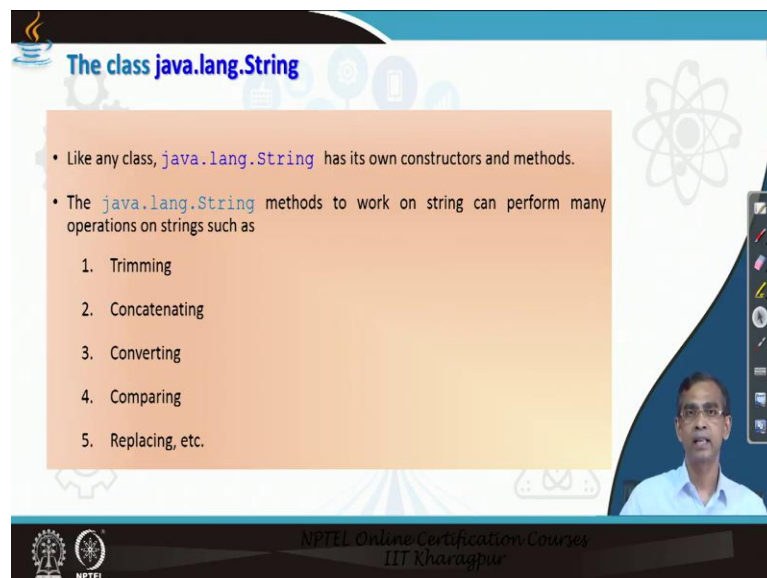
If the input is a floating point value, then you should parse it as a float value or double or Boolean accordingly. Otherwise, runtime exception will occur. Possibly you know, you can check it by writing parsing one that already you know, so you can follow this and it will basically. Now here I want to say everything that is passed here is basically is a string who is basically the programmer can handle it. So, this is one important concept basically how `Java` program treat everything as a string.

(Refer Slide Time: 21:36)



Now, we shall discuss about the composition of this class string the class string is in fact is heavy, heavy in the sense that it has a large number of constructors, it has a large number of methods by which a string object can be manipulated, large number of constructor because we can create a string object in many ways that is possible.

(Refer Slide Time: 22:04)



Now, let us have some quick familiarity about the string of string class which is defined in `java.lang` package. Now, so, I told you that there are many constructors I will come to the discussion of all the constructors and then many methods are there all methods are related to the usual string manipulation namely trimming, concatenation, converting comparing two

strings replacing and so many things are there. So, all these things, we will see exactly how the different methods can provide you and how we can do that.

(Refer Slide Time: 22:36)

Constructor	Description
<code>String()</code>	Initializes a newly created String object so that it represents an empty character sequence.
<code>String(byte[] bytes)</code>	Constructs a new String by decoding the specified array of bytes using the platform's default charset.
<code>String(byte[] bytes, Charset charset)</code>	Constructs a new String by decoding the specified array of bytes using the specified charset.
<code>String(byte[] ascii, int hibyte)</code>	This method does not properly convert bytes into characters. As of JDK 1.1, the preferred way to do this is via the String constructors that take a Charset, charset name, or that use the platform's default charset.
<code>String(byte[] bytes, int offset, int length)</code>	Constructs a new String by decoding the specified subarray of bytes using the platform's default charset.
<code>String(StringBuffer buffer)</code>	Allocates a new string that contains the sequence of characters currently contained in the string buffer argument.
<code>String(StringBuilder builder)</code>	Allocates a new string that contains the sequence of characters currently contained in the string builder argument.
<code>String(byte[] bytes, int offset, int length, Charset charset)</code>	Constructs a new String by decoding the specified subarray of bytes using the specified charset.

Now, let us first come to the discussion of constructors those are there in the string. So, there are many constructor as I told you, we are already familiar with some constructor that we have used possibly we have used one constructor like string passing a string and an object. Now, this is a default constructor that means, you can create a string object as a null without passing anything.

So, this is a default constructor which basically represents an empty string like. Now, some bytes array can be passed to a string, so that array bytes can be converted to a string. So, the array of bytes can be stored here as a string, here array of bytes with certain character sets according to some local characters also can be stored as a string.

Here basically bytes is an ASCII value that also can be stored in the high bits that means, how many number of bytes are there and there are many what is called other way this is basically portion of the bytes array can be converted as a string and here stringbuffer is basically one class that we will discuss, it is another concept of string basically, if you have that object then you can convert into a string object string builder like string buffer another object is also known, which can be converted to a string.

So, these are the two advanced concept of string actually and this is also similar thing a portion of string which can be converted to a portion of byte array which can be converted to string following a character sets. Now, all these constructor we can practice writing program

and then creating the methods these are the link I have given also those links basically to the Oracle document website, so you can find many things in details, what is the meaning of character sets and then how the different constructor can be called with the example everything is discussed there.

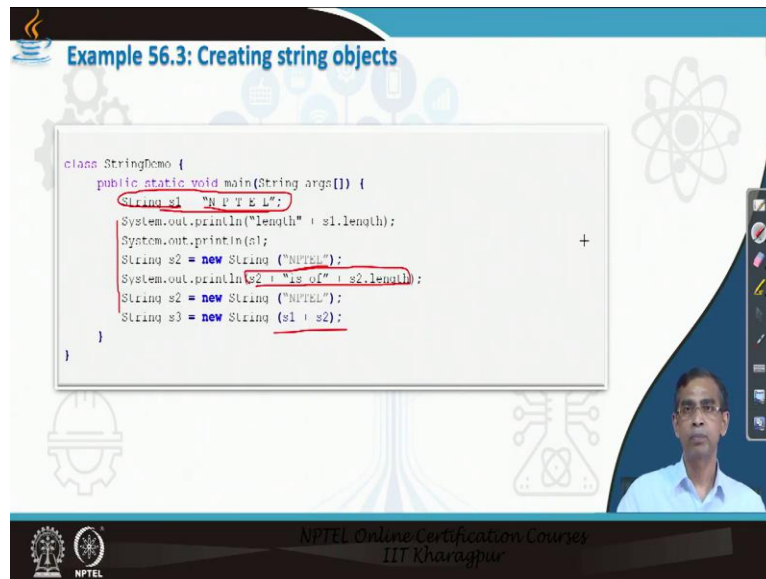
(Refer Slide Time: 24:27)

Constructor	Description
<code>String(byte[] ascii, int hibyte, int offset, int count)</code> <i>Deprecated.</i>	This method does not properly convert bytes into characters. As of JDK 1.1, the preferred way to do this is via the String constructors that take a Charset, charset name, or that use the platform's default charset.
<code>String(byte[] bytes, int offset, int length, String charsetName)</code>	Constructs a new String by decoding the specified subarray of bytes using the specified charset.
<code>String(byte[] bytes, String charsetName)</code>	Constructs a new String by decoding the specified array of bytes using the specified charset.
<code>String(char[] value)</code>	Allocates a new String so that it represents the sequence of characters currently contained in the character array argument.
<code>String(char[] value, int offset, int count)</code>	Allocates a new String that contains characters from a subarray of the character array argument.
<code>String(int[] codePoints, int offset, int count)</code>	Allocates a new String that contains characters from a subarray of the Unicode code point array argument.
<code>String(String original)</code>	Initializes a newly created String object so that it represents the same sequence of characters as the argument. In other words, the newly created string is a copy of the argument string.

Now, there are a few more constructor this is basically there are many ways a constructor can be, this is the one constructor who is currently being deprecated and there are many other this is basically is a array of characters it is a bytes and then string, character set name also we can pass and this is basically is the one example that we have already created. That means a string can be created by passing an argument as a string, these are usually example that we have already familiar to.

So, there are array characters also can be passed, while a portion of array of characters can be converted to the string and so many things are there. So, there are many all the details discussed and also it is given here. So, you may follow by reading all these things, but better the thing is that you follow each constructor, try to create objects that makes you more understanding about these string constructors. So, these are the string constructor and we will follow some example of course.

(Refer Slide Time: 25:24)



The slide displays a code editor window with the following Java code:

```
class StringDemo {  
    public static void main(String args[]) {  
        String s1 = "NPTEL";  
        System.out.println("length" + s1.length);  
        System.out.println(s1);  
        String s2 = new String ("NPTEL");  
        System.out.println(s2 + "is of" + s2.length);  
        String s2 = new String ("NPTEL");  
        String s3 = new String (s1 + s2);  
    }  
}
```

The slide also features a video feed of a presenter in the bottom right corner and the NPTEL logo in the bottom left corner.


Here is an example let us see. So, here basically we create a string here s1 by setting an array of character. So, this is the usual way the string can be created and here also you see we create a string by passing an arguments here are basically a set of characters or strings rather, we can pass as an argument and then concatenation of these things.

So, these are the few general way of creating string by calling the different constructors, there are other constructors, which is not possible to cover by illustration, I advise you to just write your program of your own and then use them. So, this is a few example of the constructor.

(Refer Slide Time: 26:10)




The slide features a central graphic of a coffee cup with steam rising from it, positioned to the left of a large, stylized tree diagram. The text "Methods of String" is displayed in blue font to the right of the coffee cup. The slide also includes a video feed of a presenter in the bottom right corner and the NPTEL logo in the bottom left corner.



Methods of String class

Method	Description
<code>contains(CharSequence cs)</code>	Returns true if and only if this string contains the specified sequence of char values.
<code>contentEquals(CharSequence cs)</code>	Compares this string to the specified CharSequence.
<code>contentEquals(StringBuffer sb)</code>	Compares this string to the specified StringBuffer.
<code>copyValueOf(char[] data)</code>	Returns a String that represents the character sequence in the array specified.
<code>copyValueOf(char[] data, int offset, int count)</code>	Returns a String that represents the character sequence in the array specified.
<code>endsWith(String suffix)</code>	Tests if this string ends with the specified suffix.
<code>equals(Object anObject)</code>	Compares this string to the specified object.

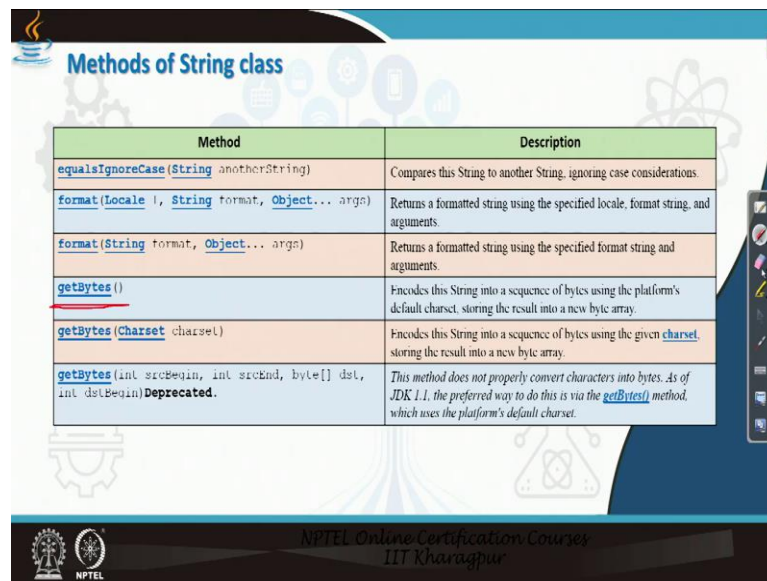


Now, let us come to the discussion of string methods, I told you there are many methods all these methods are basically the way you can manipulate string objects. So, here basically the name of the method is very simple, so that you can understand that discussion is also description also given there, you will be able to understand. So, contents means if you can call this method for a string object passing this as an argument to take that whether that string that means for which you call this method contains this set up character sequence or not.

So, these content equals whether these content equals or not, then content equals with string builder if you pass a string object objects of type string buffer and then check that they are equal or not, copy value of this basically array of characters can be converted to a string basically, if you can call the value, then all these values will be copied to the string object like and here is a portion of the array of this character array can be copied as a string.

endsWith string, if you want to find a suffix of the string on a given string and here any object can be given whether this object is equals to for which you are calling this method is equals or not that can be tested. Now, there are many such method, I do not want to discuss each method one by one I will take some examples so that you will be able to follow the different methods.

(Refer Slide Time: 27:35)

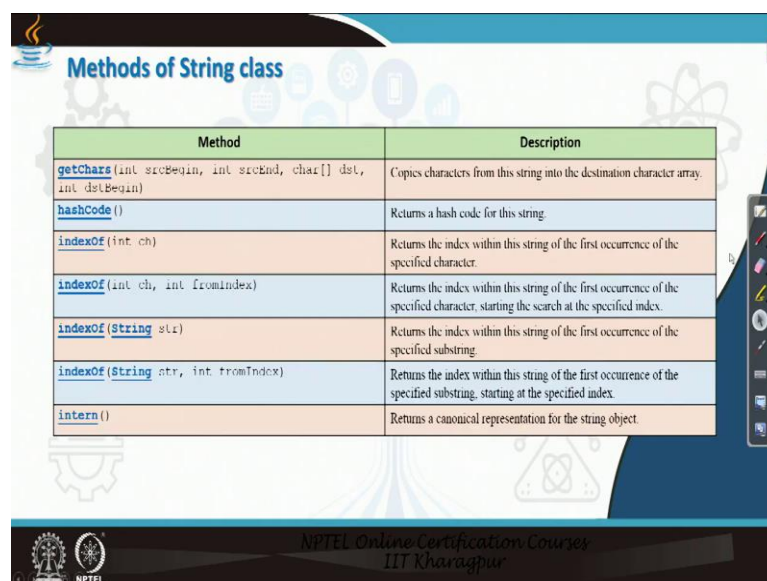


Method	Description
<code>equalsIgnoreCase(String anotherString)</code>	Compares this String to another String, ignoring case considerations.
<code>format(Locale l, String format, Object... args)</code>	Returns a formatted string using the specified locale, format string, and arguments.
<code>format(String format, Object... args)</code>	Returns a formatted string using the specified format string and arguments.
<code>getBytes()</code>	Encodes this String into a sequence of bytes using the platform's default charset, storing the result into a new byte array.
<code>getBytes(Charset charset)</code>	Encodes this String into a sequence of bytes using the given charset, storing the result into a new byte array.
<code>getBytes(int srcBegin, int srcEnd, byte[] dst, int dstBegin)</code> Deprecated.	<i>This method does not properly convert characters into bytes. As of JDK 1.1, the preferred way to do this is via the <code>getBytes()</code> method, which uses the platform's default charset.</i>

NPTEL Online Certification Courses
IIT Kharagpur

Possibly you are familiar about `getBytes`, this means that if you call this method for a string, it basically convert all the strings into an array of bytes. So, these basically to converting this one, now let us have some examples, so that we can understand the different methods in it.

(Refer Slide Time: 27:54)



Method	Description
<code>getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin)</code>	Copies characters from this string into the destination character array.
<code>hashCode()</code>	Returns a hash code for this string.
<code>indexOf(int ch)</code>	Returns the index within this string of the first occurrence of the specified character.
<code>indexOf(int ch, int fromIndex)</code>	Returns the index within this string of the first occurrence of the specified character, starting the search at the specified index.
<code>indexOf(String str)</code>	Returns the index within this string of the first occurrence of the specified substring.
<code>indexOf(String str, int fromIndex)</code>	Returns the index within this string of the first occurrence of the specified substring, starting at the specified index.
<code>intern()</code>	Returns a canonical representation for the string object.

NPTEL Online Certification Courses
IIT Kharagpur

Methods of String class

Method	Description
<code>isEmpty()</code>	Returns true if, and only if, <code>length()</code> is 0.
<code>lastIndexOf(int ch)</code>	Returns the index within this string of the last occurrence of the specified character.
<code>lastIndexOf(int ch, int fromIndex)</code>	Returns the index within this string of the last occurrence of the specified character, searching backward starting at the specified index.
<code>lastIndexOf(String str)</code>	Returns the index within this string of the last occurrence of the specified substring.
<code>lastIndexOf(String str, int fromIndex)</code>	Returns the index within this string of the last occurrence of the specified substring, searching backward starting at the specified index.
<code>length()</code>	Returns the length of this string.
<code>matches(String regex)</code>	Tells whether or not this string matches the given regular expression .

NPTEL Online Certification Courses
IIT Kharagpur

Methods of String class

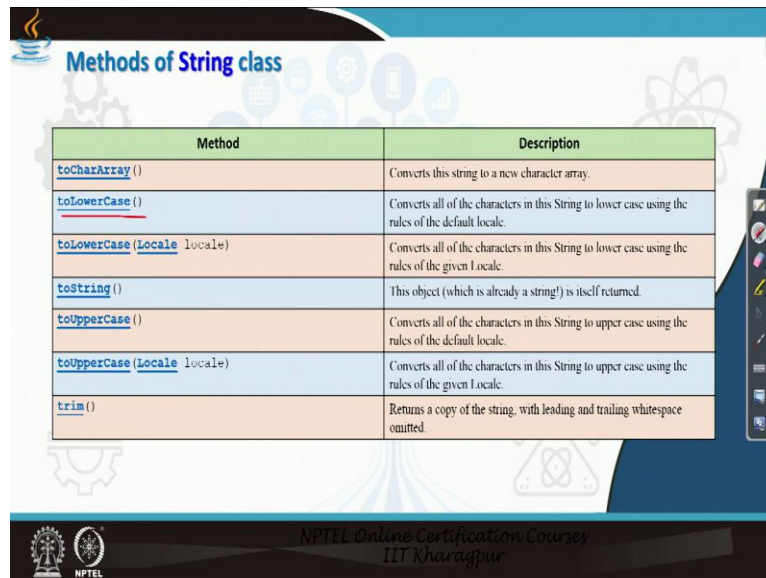
Method	Description
<code>offsetByCodePoints(int index, int codePointOffset)</code>	Returns the index within this String that is offset from the given index by <code>codePointOffset</code> code points.
<code>regionMatches(boolean ignoreCase, int toffset, String other, int ooffset, int len)</code>	Tests if two string regions are equal.
<code>regionMatches(int toffset, String other, int ooffset, int len)</code>	Tests if two string regions are equal.
<code>replace(char oldChar, char newChar)</code>	Returns a new string resulting from replacing all occurrences of <code>oldChar</code> in this string with <code>newChar</code> .
<code>replace(CharSequence target, CharSequence replacement)</code>	Replaces each substring of this string that matches the literal target sequence with the specified literal replacement sequence.
<code>replaceAll(String regex, String replacement)</code>	Replaces each substring of this string that matches the given regular expression with the given replacement.

NPTEL Online Certification Courses
IIT Kharagpur

Methods of String Class

Method	Description
<code>replaceFirst(String regex, String replacement)</code>	Replaces the first substring of this string that matches the given regular expression with the given replacement.
<code>split(String regex)</code>	Splits this string around matches of the given regular expression .
<code>split(String regex, int limit)</code>	Splits this string around matches of the given regular expression .
<code>startsWith(String prefix)</code>	Tests if this string starts with the specified prefix.
<code>startsWith(String prefix, int toffset)</code>	Tests if the substring of this string beginning at the specified index starts with the specified prefix.
<code>subSequence(int beginIndex, int endIndex)</code>	Returns a new character sequence that is a subsequence of this sequence.
<code>substring(int beginIndex)</code>	Returns a new string that is a substring of this string.
<code>substring(int beginIndex, int endIndex)</code>	Returns a new string that is a substring of this string.

NPTEL Online Certification Courses
IIT Kharagpur

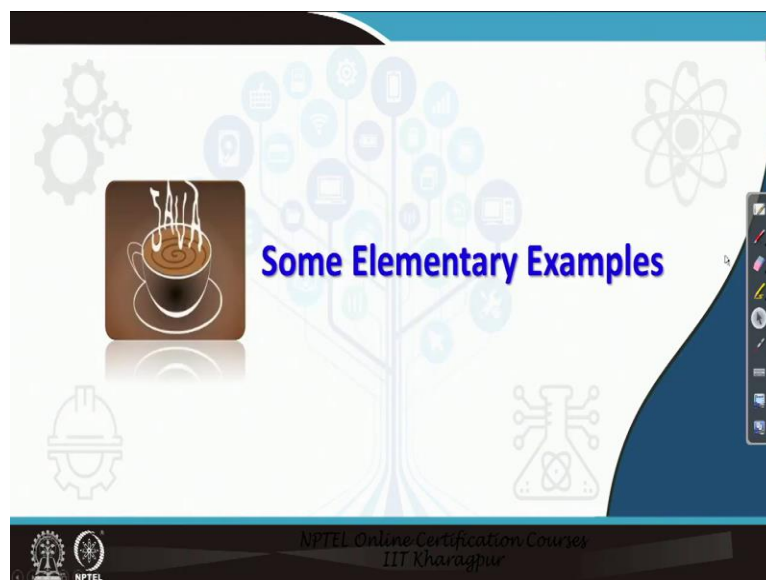


The slide is titled "Methods of String class" and features a table with two columns: "Method" and "Description". The table lists seven methods of the String class with their respective descriptions. The background of the slide includes decorative icons like gears, a tree, and a flask, along with the NPTEL logo at the bottom.

Method	Description
<code>toCharArray ()</code>	Converts this string to a new character array.
<code>toLowerCase ()</code>	Converts all of the characters in this String to lower case using the rules of the default locale.
<code>toLowerCase (Locale locale)</code>	Converts all of the characters in this String to lower case using the rules of the given Locale.
<code>toString ()</code>	This object (which is already a string!) is itself returned.
<code>toUpperCase ()</code>	Converts all of the characters in this String to upper case using the rules of the default locale.
<code>toUpperCase (Locale locale)</code>	Converts all of the characters in this String to upper case using the rules of the given Locale.
<code>trim ()</code>	Returns a copy of the string, with leading and trailing whitespace omitted.

Now, there are plenty of methods I have listed all the methods which are possible here and the meaning of all the methods are very easily understandable. As you know, what is the meaning of these toLowercase, that means if you call a, call this method for a string object, so it will convert the string object to a lowercase methods. Now, so these are the examples that fine, other examples are many easily understandable, I mean, the method description is easily understandable.

(Refer Slide Time: 28:24)



Example 56.4 : Create a String using literal and print

```
Code
public class createStringDemo{
    public static void main(String args[]){
        String text = "DATA STRUCTURES AND ALGORITHMS USING JAVA";
        System.out.print(text);
    }
}
```

Output
DATA STRUCTURES AND ALGORITHMS USING JAVA

NPTEL Online Certification Courses
IIT Kharagpur

So, let us have certain elementary example, so that we can understand at least few methods, and then how it works actually, rest of the methods you can practice on your own and then you can understand much more in details.

Now, here is a simple example. So, create string demo how we can create a string object that we are already familiar. So, we just, this is the name of the string objects and this is the initial value of the string and this is a print, so it will print like this. So, this way you can create string object, this is the usual method that we are already familiar to.

(Refer Slide Time: 28:55)

Converting Strings

NPTEL Online Certification Courses
IIT Kharagpur

Methods of converting one from to others

Method	Description
<code>length()</code>	Returns the length of this string.
<code>charAt(int index)</code>	Returns the char value at the specified index.
<code>concat(String str)</code>	Concatenates the specified string to the end of this string.
<code>isEmpty()</code>	Returns true if, and only if, <code>length()</code> is 0.
<code>toCharArray()</code>	Converts this string to a new character array.
<code>toLowerCase()</code>	Converts all of the characters in this String to lower case using the rules of the default locale.
<code>toLowerCase(Locale locale)</code>	Converts all of the characters in this String to lower case using the rules of the given Locale.
<code>toString()</code>	This object (which is already a string!) is itself returned.
<code>toUpperCase()</code>	Converts all of the characters in this String to upper case using the rules of the default locale.
<code>toUpperCase(Locale locale)</code>	Converts all of the characters in this String to upper case using the rules of the given Locale.
<code>trim()</code>	Returns a copy of the string, with leading and trailing whitespace omitted.

NPTEL Online Certification Courses
IIT Kharagpur

Now, let us consider another few methods. There are many methods I just categorize all these method regarding converting from one string to another string.

(Refer Slide Time: 29:05)

Example 56.5 : length() of String

`int length()` Returns the number of characters in the String.

```

public class StringlengthDemo{
    public static void main(String args[]){
        String text = "DATA STRUCTURE WITH JAVA";
        int stringLength = text.length();
        System.out.print(stringLength);
    }
}

```

Output: 24

NPTEL Online Certification Courses
IIT Kharagpur

And let us consider this example. So stringlength demo, as I told you, so if this is the string and the length method is there and if you print this basically return, what is the length of the string that is there.

This within brackets also, sometimes it is required, not required Java takes it at `(())`(29:23), so you can, these are not mandatory and then these basically print the length of the string. So, in this case, the length of the string as the program will print is 24.

(Refer Slide Time: 29:41)

Example 56.6 : charAt()

char charAt(int i) Returns the character at i-th index.

```
public class CharAccessDemo{
    public static void main(String args[]){
        String text = "DATA STRUCTURE WITH JAVA";
        char data = text.charAt(4);
        System.out.print(data);
    }
}
```

Output: A

NPTEL Online Certification Courses
IIT Kharagpur

Now, so there are many such methods, I will let us consider some more methods about, so this is the string object that we have created and character at four. So this basically 0, 1, 2, 3 so this is the character at 4. So, if you can just access any passing, any index as a characters, so if you say suppose here character is total say 29, you write a 92 then it basically exception will throw in this case.

So, if it is beyond the array of string like and this basically print the data that mean which is the character at. So, this basically return this character at return, what is the character of the ith index of this string. So, for example, this program will give output as 4 in this case.

(Refer Slide Time: 30:29)

Example 56.7 : concat()

String concat(String s) Concatenates specified string to the end of this string.

```
public class StringMergingDemo{
    public static void main(String args[]){
        String text1 = "DATA STRUCTURE WITH";
        String text2 = " JAVA";
        String text3 = text1.concat(text2);
        System.out.print(text3);
    }
}
```

Output: DATA STRUCTURE WITH JAVA

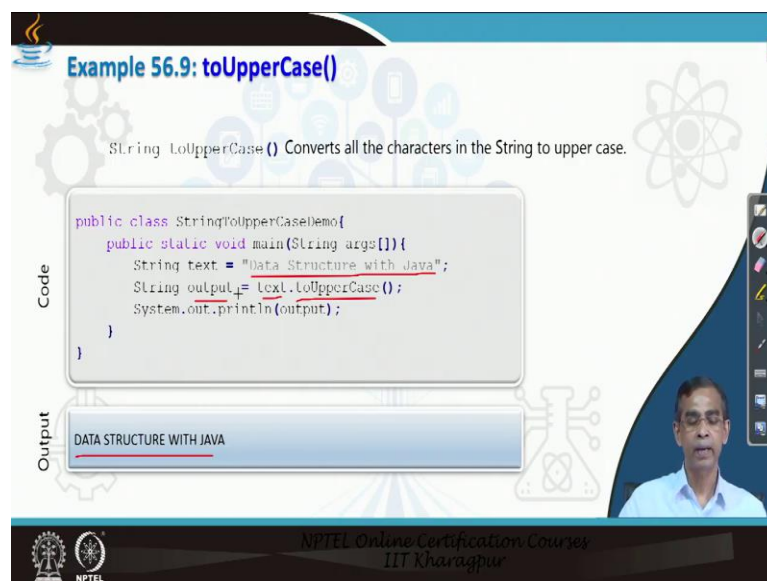
NPTEL Online Certification Courses
IIT Kharagpur

Now, this is another example where you can see how the concat method is there, concat method means it basically merging, it is equivalent to same as takes on plastics to also. So, this is on string and this is another steam, we call this concat method for the text one that mean this, this this takes to this one will be concatenated after this one, so this is the output that this program will give it to you.

So, these are concatenate method there are many other methods so class also like, so this is also equivalent to text on class text 2. So, they are same. So, this text 3 is basically there, now you see we are creating new object and if you write here say text 2 is equals to text 1 coordinate takes 2, then it basically a program.

So, if you can text the program writing this text two and you see that a compile time error it will throw, it is during the compilation it will check and then they will throw an exception, the program will not compile successfully. Because text 2 can is immutable that means you cannot convert merging the 2 string and storing into the string like it cannot be.

(Refer Slide Time: 31:58)



The screenshot shows a presentation slide with the following content:

- Title:** Example 56.9: toUpperCase()
- Description:** String toUpperCase () Converts all the characters in the String to upper case.
- Code:**

```
public class StringToUpperCaseDemo{
    public static void main(String args[]){
        String text = "Data Structure with Java";
        String output = text.toUpperCase ();
        System.out.println(output);
    }
}
```
- Output:** DATA STRUCTURE WITH JAVA

The slide also features a small video inset of a speaker in the bottom right corner and logos for NPTEL and IIT Kharagpur at the bottom.

So, this is another example where we can use two lowercase as the name implies that if we call this method for this object string, whatever the character it is there it will convert it into the lowercase. So, if this is the text objects and the output of this program will be like this.

(Refer Slide Time: 32:22)

The slide displays the following content:

Example 56.9: toUpperCase()

String toUpperCase () Converts all the characters in the String to upper case.

```
public class StringToUpperCaseDemo{
    public static void main(String args[]){
        String text = "Data Structure with Java";
        String output = text.toUpperCase();
        System.out.println(output);
    }
}
```

Output

DATA STRUCTURE WITH JAVA

NPTEL Online Certification Courses
IIT Kharagpur

So, these are the simple method is easily understandable and you can just play this writing the program and then you can understand about how the different methods are working for you and so, here is basically 2 uppercase, this is the string we call the method for this string and it will print this one and it basically store the result and string changing all characters converting the uppercase will be there.

(Refer Slide Time: 32:49)

The slide displays the following content:

Example 56.10: trim()

String trim () Returns the copy of the String, by removing whitespaces at both ends. It does not affect whitespaces in the middle.

```
public class TrimStringDemo{
    public static void main(String args[]){
        String text = " Data Structure with Java ";
        String output = text.trim();
        System.out.println(output);
    }
}
```

Output

Data Structure with Java

NPTEL Online Certification Courses
IIT Kharagpur

There is another example the trim method, this method basically if any whitespace before and after the string is there, it will basically clean and then it will print this on. So, this basically all the blank space those are there is basically trimmed. So, this is a method and it creates a new stream and then you can clean that stream.

(Refer Slide Time: 33:11)

The slide is titled "Example 56.11 : replace()". It features a description of the `String.replace(char old, char new)` method, which returns a new string by replacing all occurrences of the old character with the new one. Below this, a code block shows a Java class `StringReplaceDemo` with a `main` method. The code sets a string `text = "Data Structure with C++"`, replaces all occurrences of `'C++'` with `"Java"`, and prints the resulting `output`. The output section shows `Data Structure with Java`. The slide also includes the NPTEL logo and the text "NPTEL Online Certification Courses IIT Kharagpur".

```
String replace(char old, char new) Returns new string by replacing all occurrences of old with new.
```

```
public class StringReplaceDemo{
    public static void main(String args[]){
        String text = "Data Structure with C++";
        String output = text.replace("C++", "Java");
        System.out.println(output);
    }
}
```

Output: Data Structure with Java

And here is one example say replace. So, this is the one text the call the replacement of for this string that means in this string, if some character is with this one or some substring will be like this, it will be replaced by that newly given sub substring. So, this basically gives you a result and string, not that texts, you cannot write text equals to this one that is not possible instnct.

So, what is the new thing, so it basically creates a new string object, this is a new string object and then you can, now the output of this program as you can see this one. So, I hope you have understood that how the different methods are there, they are similar many methods are there which is not possible to cover.

(Refer Slide Time: 33:58)

The slide is titled "REFERENCES". It lists three references:

- [The Complete Reference, Herbert Schildt, 9th Edition, Oracle Press](#)
- <https://cse.iitkgp.ac.in/~dsamanta/javads/index.html>
- <https://docs.oracle.com/javase/tutorial/>

But I advise you to just check all the methods and then create different string objects in different way there are so many constructors and try to understand that how the different methods work. For details discussion, you can have many more concept that is covered here in this book that you can follow and in this link also I have given a very concentrated way at the total what is called the story of the string class and then the different programs whatever I have used and few more programs also additional programs it is included here.

So, you can take it and enjoy programming and this is the, these are super what is called the documentation who is basically gives everything in details about the string class, which is defined in java.lang. So, go to this link and then search to the java.lang and you can string class you can find many more information from there. This is few example that I have covered. There are a few more discussion, which is required in order to complete this string class that will be discussed in the next class. Thank you very much.