String

String Constructors

String s = new String();

String(char chars[])

Char chars[]={ ‘a’,’b’,’c’}
String s = new String(chars);

Constructor initializes s with the string abc

int length( )

System.out.println(s.length()); 3

String(String strObj)

String s1 = new String(s);
System.out.println(s); abc
System.out.println(s1); abc

String(char chars[], int startIndex, int numchars)

Char chars[] = {‘a’,’b’,’c’,’d’,’e’,’f’};
String s = new String(chars, 2,3); cde

String(byte asciiChars[ ])

String(byte asciiChars[ ], int startIndex, int numchars)

byte ascii[ ] = {65,66,67,68,69,70}
String s = new String(ascii); ABCDEF
String s = new String(ascii, 2,3); CDE

String Literals

String s = “Java”;

String conversion and toString( )

String toString( ) – is automatically invoked when a object is used in a concatenation expression or in a call to println( )

Character Extraction

char charAt (int where)

```
char ch;
ch = "abc".charAt(1);                    b
```

void getChars(int sourceStart, int sourceEnd,char target[ ], int targetStart)

```
String s = "This is a demo of the getchars method";
int start = 10;
int end = 14;
char buf[ ]=new char[end-start];
s.getChars(start,end,buf,0);
System.out.println(buf);                 demo
```

byte[ ] getBytes( ) – uses the default character to byte conversion
- is useful when exporting a String value into an environment that does not support 16-bit Unicode characters

char[ ] toCharArray( ) – converts all the characters in a String object into a character array

equals( ) and equalsIgnoreCase( )

boolean equals(Object str)
boolean equalsIgnoreCase(String str)

```
String s1 = "Hello";
String s2 = "HELLO";
s1.equals(s2);                   true
s1.equalsIgnoreCase(s2);         false
```

boolean startsWith(String str)
boolean endsWith(String str)

```
"Foobar".endsWith("bar")         true
"Foobar".startsWith("Foo")       true
```

equals( ) versus = =

```
s1 = "Hello";
s2 = "Hello";
Hello equals Hello               True
Hello  = =  Hello                false
```

int compareTo(String str)

Value	Meaning
Less than zero	The invoking string is less than str
Greater than zero	The invoking string is greater than str
Zero	The two strings are equal

To search a string for a specified character or substring

String s = " Now is the time for all good men" + "to come to the aid of their country.";

int indexOf(int ch)
int lastIndexOf(int ch)

s.indexOf('t') - 7
s.lastIndexOf('t') - 65

int indexOf(String str)
int lastIndexOf(String str)

s.indexOf("the") - 7
s.lastIndexOf("the") - 55

int indexOf(int ch, int startIndex)
int lastIndexOf(int ch, int startIndex)

s.indexOf('t',10) - 11
s.lastIndexOf('t',60) - 55

int indexOf(String str, , int startIndex)
int lastIndexOf(String str , int startIndex)

s.indexOf("the",10) - 44
s.lastIndexOf("the",60) - 55

Modifying a String

String substring(int startIndex)
String substring(int startIndex, int endIndex)

concat( )

String s1 = "one";
String s2 = s1.concat("two"); onetwo

replace( )

String s = “Hello”.replace(‘l’,’w’); Hewwo

trim( ) – remove leading or trailing whitespaces

String s = “ Hello World “.trim( ); Hello world

Data conversion using valueOf( )

static String valueOf(double num)
static String valueOf(long num)
static String valueOf(Object ob)
static String valueOf(char chars[ ])

Changing the case of characters

String toLowerCase( )
String toUpperCase( )

String s = “This is a test”;

s.toUpperCase( ); - THIS IS A TEST
s.toLowerCase( ); - this is a test

StringBuffer – is a peer class of String

StringBuffer Constructors

StringBuffer( )
StringBuffer(int size)
StringBuffer(String str)

length( ) and capacity( )

StringBuffer Sb = new StringBuffer(“Hello”);

Sb - Hello
Sb.length( ) - 5
Sb.capacity( ) - 21

void ensureCapacity(int capacity) – to preallocate room for a certain number of characters after a StringBuffer has been constructed

- to set the size of the buffer

setLength( ), charAt( ) and setCharAt( )

void setLength(int len) – increase in the size of the buffer will include null characters to the end of the existing buffer
- decrease in the value than the existing buffer value, the characters beyond the new length will be lost

char charAt(int where)

void setCharAt(int where, char ch)

StringBuffer Sb = new StringBuffer("Hello");

Sb	Hello
Sb.charAt(1);	e
Sb.setCharAt(1, 'i');	
Sb.setLength(2);	
Sb	Hi
Sb.charAt(1)	i

getChars( ) – To copy a substring of a stringBuffer into an array

void getChars(int sourceStart, int sourceEnd, char target[ ], int targetStart)

sourceStart - specifies the index of the beginning of the substring
sourceEnd - specifies an index that is one past the end of the desired substring

i.e., sourceStart through sourceEnd – 1

target[ ] - the array that will receive the characters
targetStart - the index within target at which the substrinj will be copied is passed here

append( )

StringBuffer append(String str)
StringBuffer append(int num)
StringBuffer append(Object obj)

String s;
int a = 42;

StringBuffer Sb = new StringBuffer(40);

s = Sb.append("a=").append(a).append(!).toString( ); a = 42!

insert( )

StringBuffer insert(int index, String str)
StringBuffer insert(int index, int num)
StringBuffer insert(int index, Object obj)

StringBuffer Sb = new StringBuffer("I Java !");
Sb.insert(2, "like"); I like Java !

reverse( )

StringBuffer reverse( )

StringBuffer Sb = new StringBuffer("Java");
Sb.reverse( ); avaJ

delete( ) and deleteCharAt( )

StringBuffer delete(int startIndex, int endIndex) i.e., startIndex to endIndex-1
StringBuffer deleteCharAt(int loc)

StringBuffer Sb = new StringBuffer("This is a test.");
Sb.delete(4,7);
Sb This a test
Sb.deleteCharAt(0);
Sb his a test

replace( )

StringBuffer replace(int startIndex, int endIndex, String str)

StringBuffer Sb = new StringBuffer("This is a test.");

Sb.replace(5,7,"was"); This was a test

Substring( ) – which returns a returns a portion of a StringBuffer

StringBuffer substring(int startIndex)
StringBuffer substring(int startIndex, int endIndex)