



STRINGBUILDER

SUMMARY OF TOPICS

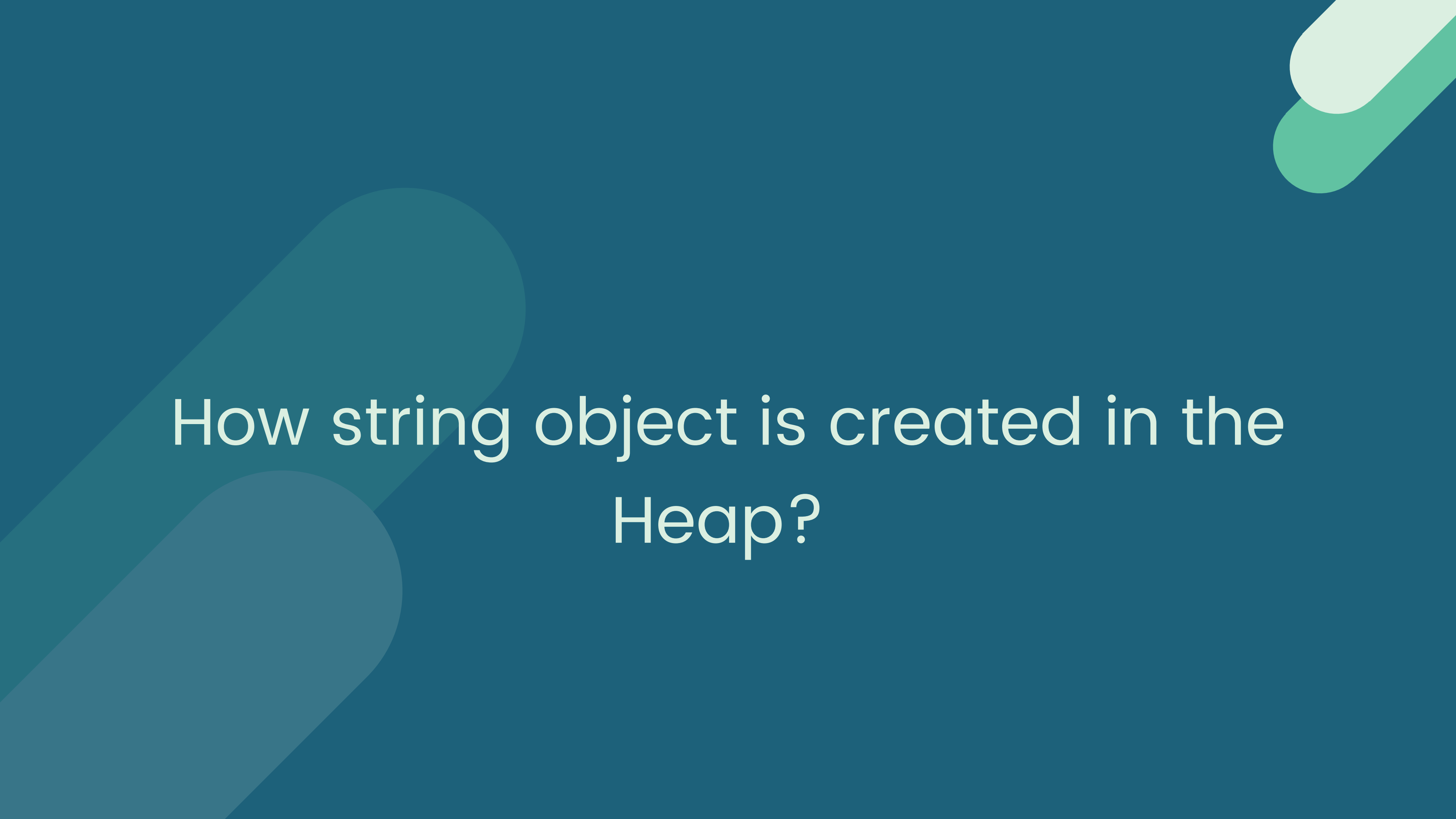
Why we need StringBuilder?

- ❖ How to create StringBuilder?
 - ❖ StringBuilder Methods
 - ❖ Mutability and chaining
 - ❖ Equality of object

WHY WE NEED STRINGBUILDER?

To avoid creating a lot of String objects we need
StringBuilder.

- ❖ Unlike String it is mutable.



How string object is created in the
Heap?

Creating to the StringBuilder Object

```
StringBuilder builder=new StringBuilder();
```

- ❖

```
StringBuilder builder1=new StringBuilder("Techtorial");
```
- ❖

```
StringBuilder builder2=new StringBuilder(10);
```

This method adds the parameter to the `StringBuilder` which we gave, and returns a reference to the current `StringBuilder`.

- ❖ **This method takes around 10 different data types.**
 - ❖ **You can do method chaining.**

EXAMPLES

```
StringBuilder builder=new StringBuilder();  
builder.append("Techtorial").append(2019);  
System.out.println(builder);
```

EXAMPLE

```
StringBuilder success=new  
StringBuilder().append("Techtorial").append(2020);  
success.append("Best Year Ever ").append(true);  
System.out.println(success);
```

DIFFERENCE BETWEEN STRING AND STRINGBUILDER

StringBuilder instead of creating a lot of object it creates one time and use same object every time. However in String we are creating every time new object. If previous object has no reference to point then it is eligible for garbage collection



MUTABILITY OF THE STRINGBUILDER

Unlike to the String, StringBuilder is not immutable. So StringBuilder is mutable. It means every time when you call the StringBuilder method the value for the StringBuilder will change.

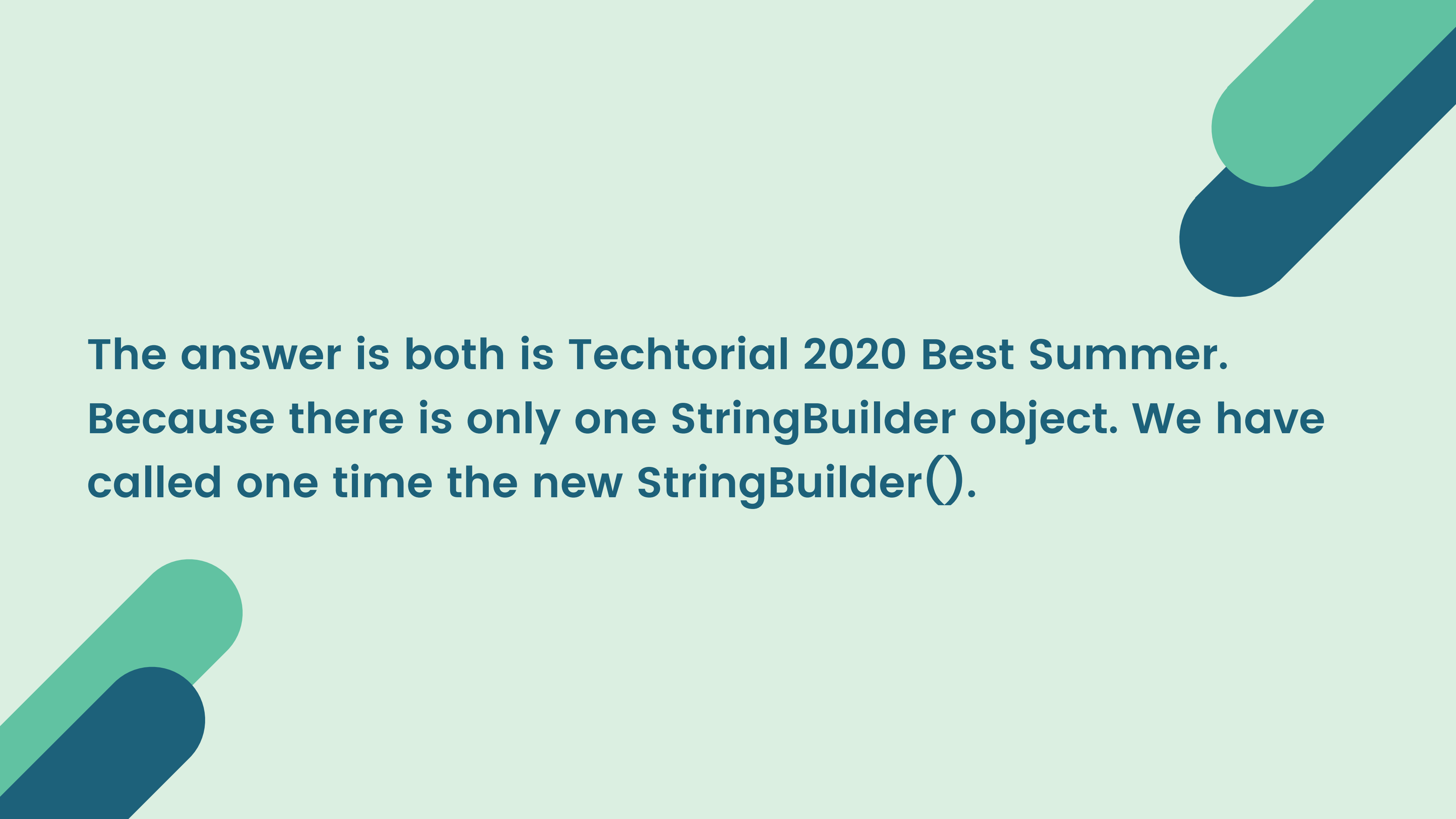
EXAMPLE

```
StringBuilder success=new StringBuilder("Working  
Hard");
```

```
❖ success.append(" does not make you successful.");  
❖ StringBuilder truth=success.append("Thinking hard makes  
you successful");  
System.out.println(success);  
System.out.println(truth);
```

Example

```
StringBuilder success=new StringBuilder("Tech");  
    success.append("torial");  
    StringBuilder truth=success.append(" 2020");  
truth=truth.append(" Best").append(" Summer");  
    System.out.println(success);  
    System.out.println(truth);
```



**The answer is both is Tectorial 2020 Best Summer.
Because there is only one StringBuilder object. We have
called one time the new StringBuilder().**

charAt() – indexOf() – length() – substring()

charAt() = takes the int and return the char which is pointing to that int.

❖ indexOf() = takes the char and return the int which is pointing to that char.

length() – substring()

- ❖ length() = returns the length of the StringBuilder
- ❖ substring() = returns the value from starting point and right before ending point.
- ❖ NOTE: substring() returns a String rather than StringBuilder. That is why the value will not change.

insert()

insert() method adds characters to the **StringBuilder** at the requested index and returns a reference to the current **StringBuilder**.

❖ method signature is :

```
StringBuilder insert(int offset, String str);
```

EXAMPLES

```
StringBuilder success=new StringBuilder();
```

```
❖ success.append("preparation");
```

```
❖ success.insert(0,"$");
```

```
❖ success.insert(11,"$");
```

```
System.out.println(success);
```

```
❖ success.insert(3,"-");
```


DELETE() – DELETECHARAT()

delete method is removes the characters from the sequence and return the reference to the current **StringBuilder**.

- ❖ **StringBuilder delete(int start, int end)**
- ❖ **StringBuilder deleteCharAt(int index)**

EXAMPLES

```
StringBuilder sName=new StringBuilder("Microsoft");
```

- ❖ sName.delete(1,4);
- ❖ System.out.println(sName);
- ❖ sName.deleteCharAt(1);
- ❖ System.out.println(sName);
- ❖ sName.deleteCharAt(4);
- ❖ System.out.println(sName);

REVERSE()

It reverse the characters in the sequences and returns a reference to the current `StringBuilder`.

`StringBuilder reverse()`

TOSTRING()

If we want to convert StringBuilder object to String.

We need to use the toString() method.

Without convert we cannot assign StringBuilder to String



UNDERSTANDING THE EQUALITY OF THE STRINGBUILDER

Example

```
StringBuilder sName=new StringBuilder();  
StringBuilder sName1=new StringBuilder();  
StringBuilder sName2=sName.append("Techtorial");  
System.out.println(sName==sName1);  
System.out.println(sName==sName2);
```