Core Methods of String Objects

This lesson gets you acquainted with some of the most common built-in functions of strings.

We'll cover the following

Some of the core methods are discussed in this lesson. You can find a list of all the String methods in Rust documentation of Strings.

Capacity in Bytes

The `capacity` gives the number of bytes allocated to the String, unlike `len` which gives the number of bytes taken by the String object. To get the capacity of a variable in **bytes**, use the built-in function `capacity()`.

Syntax

The general syntax is:

`str.capacity()`

Here `str` is the string whose capacity is to be found.

> **Note:** The length of String will always be less than or equal to the capacity.

```
fn main() {
  // define a growable string variable
  let course = String::from("Rust");
  println!("This is a beginner course in {}.", course);
  //capacity in bytes
  println!("Capacity: {}.", course.capacity());
}
```

Finding a Substring

To find if one string contains another string, use the `contains()` built-in function.

Syntax

The general syntax is :

`str.contains("sub_str")`

Here `str` is the original string and `"sub_str"` is a substring which is to be found in a string.

```
fn main() {
  // define a growable string variable
  let str = String::from("Rust Programming");
  let sub_str = String::from("Rust");
  println!("This is a beginner course in {}.", str);
  // find if string contains a substring
  println!("{} is a substring of {}: {}.", sub_str, str, str.contains("Rust"));
}
```

Replace a Substring

To replace all occurrences of one substring within a String object with another String, use the `replace()` built-in function.

Syntax

The general syntax is :

`str.replace(replace_from, replace_to)`

Here str is the original string, `replace_from` is the value which is to be replaced in the string `str` and `replace_to` is the value the string is converted to.

the string `str` and `replace_to` is the value the string is converted to.

```
fn main() {
  // define a growable string variable
  let str = String::from("Rust Programming");
  let replace_from = "Programming";
  let replace_to = "Language";
  // find if string contains a substring
  let result = str.replace(replace_from, replace_to);
  println!("{} now becomes {}.", str, result);
}
```

Trim a String

To trim a string use the function `trim()`. It is used to remove leading and trailing whitespaces in a string.

Syntax

The general syntax is :

`string.trim()`

> **Note:** The trim function does not remove the space between the string.

```
fn main() {
   let string = "   Rust     Programming     ".to_string();
   let trim_string = string.trim();
   // get characters at 5,6,7,8,9,10 and 11 indexes
   println!("Trimmed_string : {}", trim_string);
}
```

Quiz

Test your understanding of core methods of Strings in Rust!

Quick Quiz on Strings Methods!

1

Common method of string object and string literal are:

2

Trim method is used to remove inline spaces.

Retake Quiz

Now that you have learned the functions of strings, let’s learn to iterate them in the next lesson.