Iterating Over Strings

This lesson teaches us how to iterate over strings.

We'll cover the following

The following methods describe three different ways of traversing a String:

Tokenizing a String Object

A String object can be tokenized on whitespace or a character token.

Tokenizing to Separate on Whitespaces

`split_whitespace` is used to split a String on the occurrence of whitespace. Loop through the String to split on whitespaces using a `for` loop.

Syntax

The general syntax is:

```
for found in  str.split_whitespace(){
    println!("{}", found);
}
```

Here `str` is the original String which is to be traversed, `split_whitespace()` is a built-in keyword to split a string on whitespaces, `for` is used to traverse over the

String and print it as soon as the whitespace is found and `found` is an iterator over the String.

Rust Programming

Output:

Rust Programming

Output:

Rust Programming

Output:

Output:

whitespace found

Output: **Rust**

Output: **Rust**

Output: Rust

Output: Rust

Output: Rust

Output: Rust

Output: Rust

Output: Rust

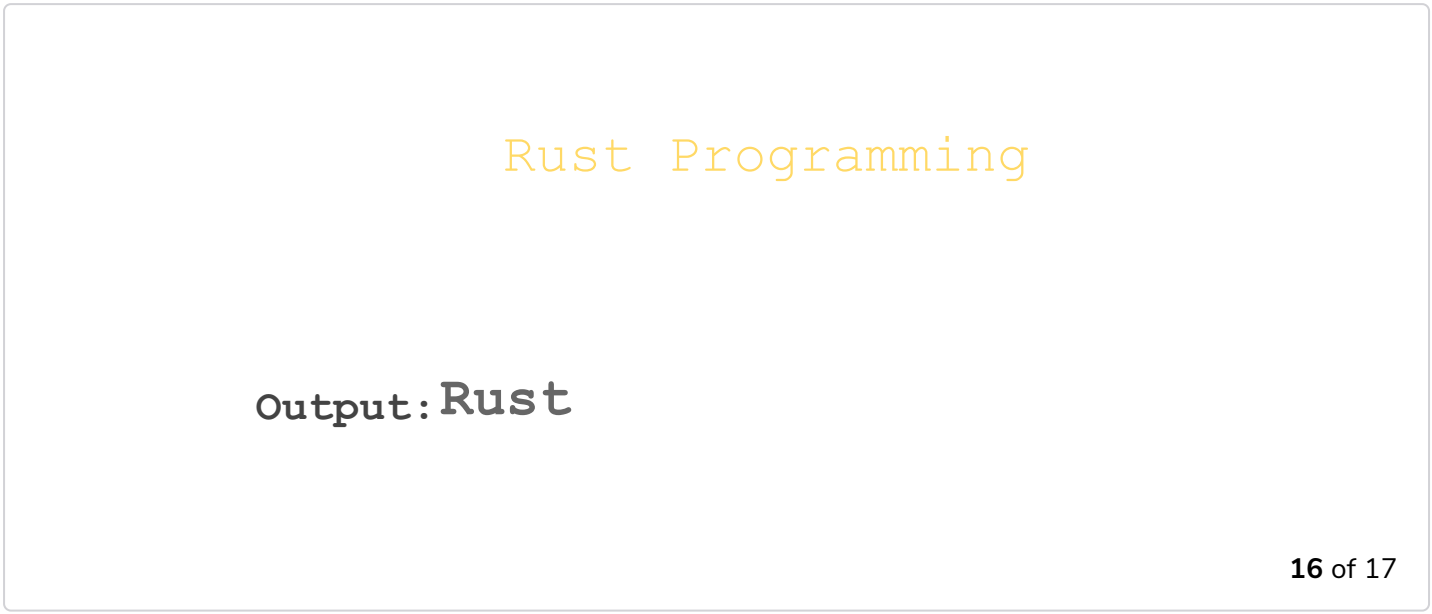
Output: Rust

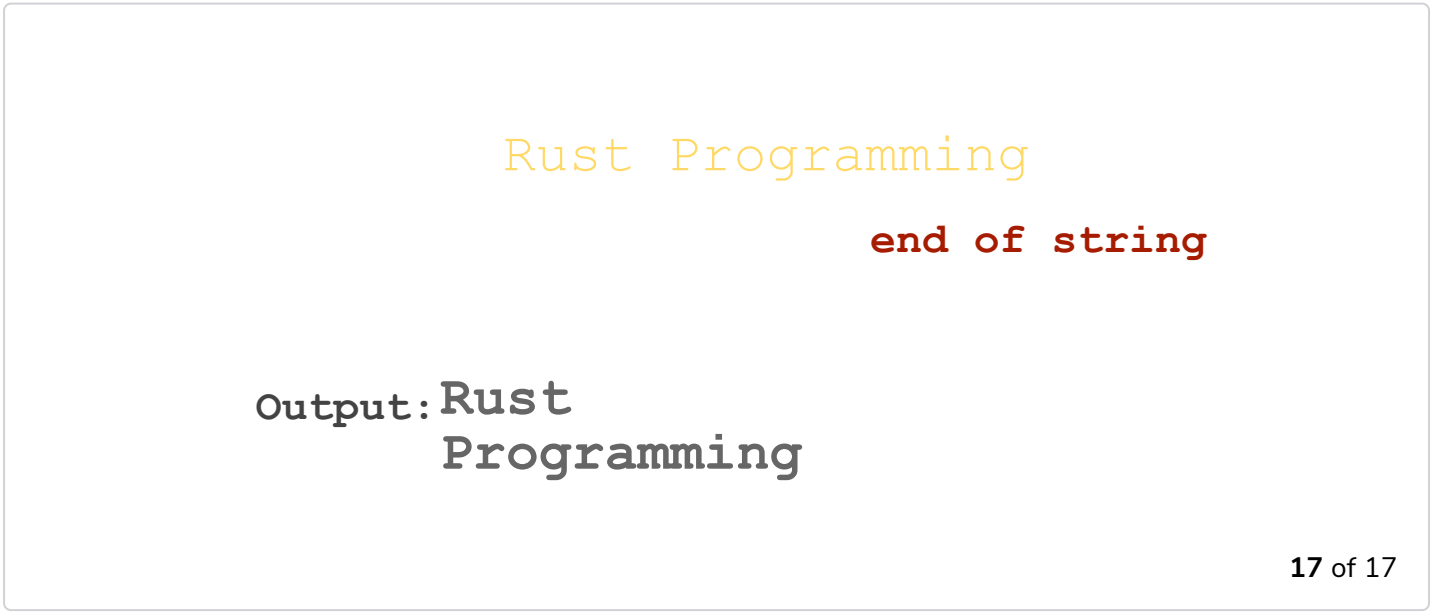
Rust Programming

Output: Rust

Rust Programming

Output: Rust






```
fn main() {
  // define a String object
  let str = String::from("Rust Programming");
  // split on whitespace
  for token in str.split_whitespace(){
      println!("{}", token);
  }
}
```

Tokenizing to Split on a Custom Character

`split` method is used to split a sentence on some token. The token is specified in the split method. This would be useful to process comma-separated data, which is

a common programming task.

Syntax

The general syntax is:

```
for found in str.split(","){
    println!("{}", found);
}
```

Here `str` is the original String which is to be traversed, `str.split()` is a built-in method which takes a parameter, i.e., any delimiter and split the sentence on that parameter, `for` is used to traverse over the String and print a word before the token.

Rust,Programming

Output:

Rust,Programming

Output:

Rust,Programming

Output:

```
Rust,Programming
```

Output:


```
Rust,Programming
```

```
token found
```

Output: Rust


```
Rust,Programming
```

Output: Rust

Rust,Programming

Output:Rust

Rust,Programming

Output:Rust

Rust,Programming

Output:Rust

Rust,Programming

Output:Rust

Rust,Programming

Output:Rust

Rust,Programming

Output:Rust

Rust,Programming

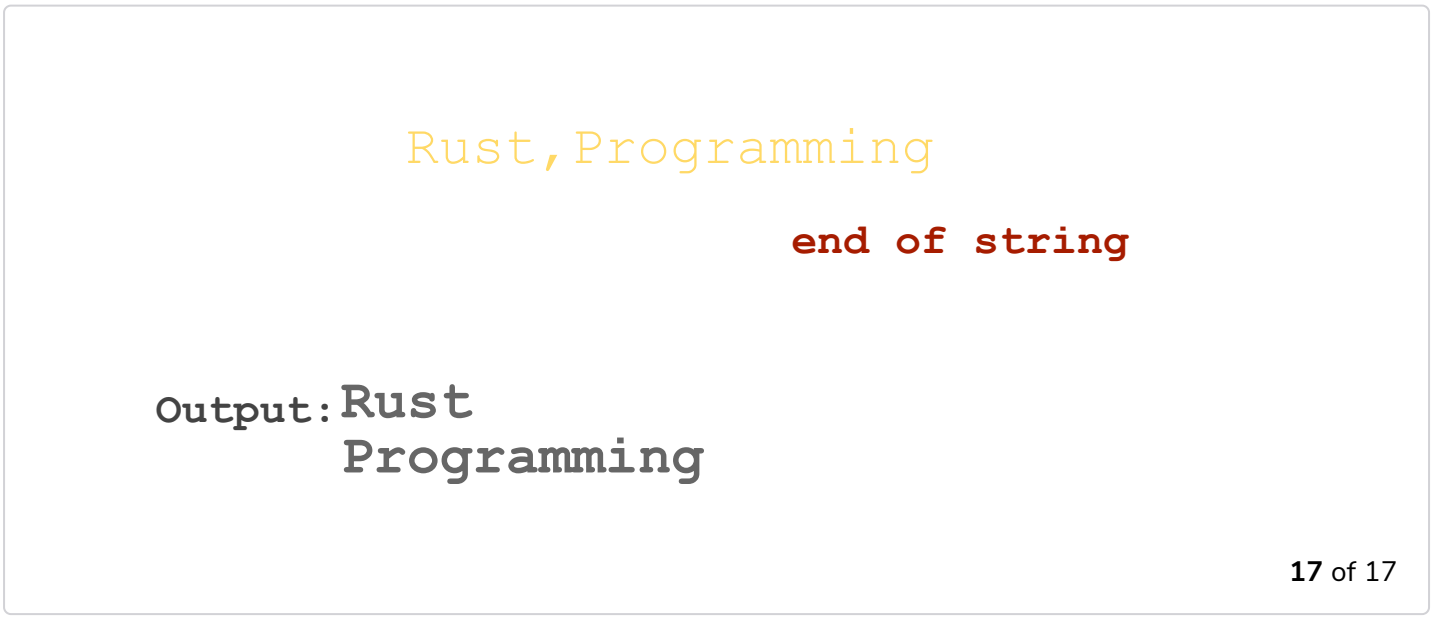
Output:Rust

Rust,Programming

Output:Rust

Rust,Programming

Output:Rust




```
fn main() {
  // define a String object
  let str = String::from("Educative, course on, Rust, Programming");
  // split on token
  for token in str.split(","){
      println!("{}", token);
  }
}
```

Iterating Over the String Object

`chars` method allows iterating over each element in a String using a `for` loop.

Syntax #

Syntax

The general syntax is:

```
for found in  str.chars(){
    println!("{}", found);
}
```

Here `str` is the original String which is to be traversed, `str.chars()` is a built-in keyword to denote letters in a String, `for` is used to traverse over the String and print every literal, and `found` is an iterator over the String.

Rust Programming

Output: Rust

Rust Programming

Output: Rust

Rust Programming

Output:Rust P

Rust Programming

Output:Rust Pr

Rust Programming

Output:**Rust Pro**

Rust Programming

Output:**Rust Prog**

Rust Programming

Output:**Rust Progr**

Rust Programming

Output:**Rust Progra**

Rust Programming

Output:**Rust Program**

Rust Programming

Output:**Rust Programm**

Rust Programming

Output:**Rust Programmi**

Rust Programming

Output:**Rust Programmin**

Rust Programming

Output: **Rust Programming**


```
fn main() {
  // define a String object
  let str = String::from("Rust Programming");
  // split on literal
  for token in str.chars(){
      println!("{}", token);
  }
}
```

Quiz

Test your understanding on looping through the String object.

Quick Quiz on Iterating Over Strings!

What is the output of the following code?

```
fn main() {
    // define a String object
    let str = String::from("Educative, course on, Rust; Programmi
ng");
    // split on token
    for token in str.split(";") {
        println!("{}", token);
```

```
    }
}
```

Retake Quiz

Now that you have learned to iterate over a string, let's learn to update a string in the next lesson.