Create and Delete

Let's start things off by learning how to create and destroy strings.

C++ offers many methods to create strings from C or C++ strings. Under the hood, there is always a C string involved for creating a C++string. That changes with C++14, because the new C++ standard supports C++ string literals: `std::string str{"string"s}`. With the suffix s,the C string literal `"string literal"` becomes a C++string literal: `"string literal"s`.

The table gives us an overview of the methods to create and delete a C++string.

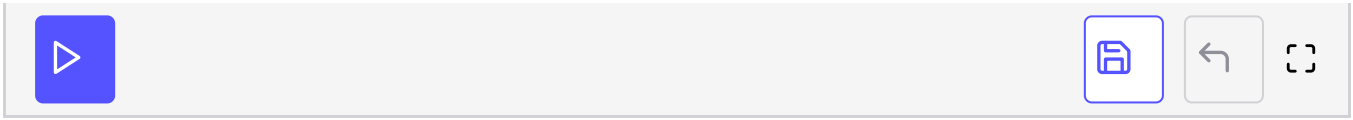
Methods	Example
Default	`std::string str`
Copies from a C++ string	`std::string str(oth)`
Moves from a C++ string	`std::string str(std::move(oth))`
From the range of a C++ string	`std::string(oth.begin(), oth.end())`
From a substring of a C++ string	`std::string(oth, otherIndex)`
From a substring of a C++ string	`std::string(oth, otherIndex, strlen)`
From a C string	`std::string str("c-string")`
From a C array	`std::string str("c-array", len)`

From characters	`std::string str(num, 'c')`
From a initializer list	`std::string str({'a', 'b', 'c', 'd'})`
From a substring	`str= other.substring(3, 10)`
Destructor	`str.~string()`

Methods to create and delete a string

```
#include <iostream>
#include <string>
#include <utility>

int main(){

  std::cout << std::endl;

  std::string defaultString;

  std::cout << "From C-String" << std::endl;

  std::string other{"123456789"};
  std::cout << "other: " << other << std::endl;

  std::cout << std::endl;

  std::cout << "From C++-string" << std::endl;

  std::string str1(other);
  std::string tmp(other);
  std::string str2(std::move(tmp));
  std::string str3(other.begin(), other.end());
  std::string str4(other, 2);
  std::string str5(other, 2, 5);

  std::cout << "str1: " << str1 << std::endl;
  std::cout << "str2: " << str2 << std::endl;
  std::cout << "str3: " << str3 << std::endl;
  std::cout << "str4: " << str4 << std::endl;
  std::cout << "str5: " << str5 << std::endl;

  std::cout << std::endl;

  std::cout << "From C-String" << std::endl;

  std::string str6("123456789", 5);
  std::string str7(5, '1');
  std::string str8({'1', '2', '3', '4', '5', '6', '7', '8', '9'});

  std::cout << "str6: " << str6 << std::endl;
```

```
  std::cout << "str7: " << str7 << std::endl;
  std::cout << "str8: " << str8 << std::endl;

  std::cout << std::endl;

  std::cout << "As Part of a C++-String" << std::endl;
  std::cout << "str6.substr(): "  << str6.substr() << std::endl;
  std::cout << "str6.substr(1): "  << str6.substr(1) << std::endl;
  std::cout << "str6.substr(1, 2): "  << str6.substr(1, 2) << std::endl;

  std::cout << std::endl;
}
```



Creation of a string

In the next lesson, we'll discuss different ways to convert C++ strings to C strings.