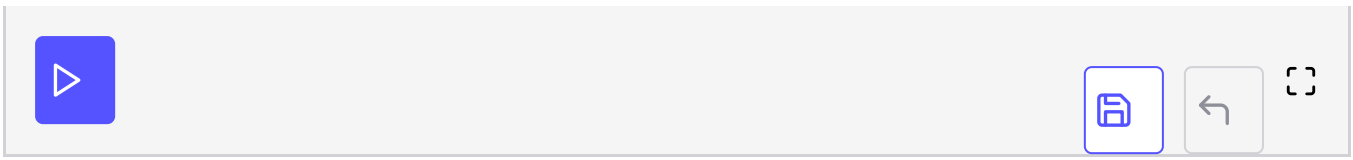
Format

Regular expressions can also specify the format of the target text. Find the implementation below.

`std::regex_replace` and `std::match_results.format`, in combination with capture groups enable us to format text. We can use a format string together with a placeholder to insert the value.

Here are both possibilities:

```
#include <regex>

#include <iomanip>
#include <iostream>
#include <string>

int main(){

  std::cout << std::endl;

  std::string future{"Future"};
  int len= sizeof(future);

  const std::string unofficial{"unofficial, C++0x"};
  const std::string official{"official, C++11"};

  std::regex regValues{"(.*), (.*)"};


  std::string standardText{"The $1 name of the new C++ standard is $2."};

  // using std::regex_replace
  std::string textNow= std::regex_replace(unofficial, regValues, standardText );

  std::cout <<  std::setw(len) << std::left << "Now: " << textNow << std::endl;

  // using std::match_results
  // typedef match_results<string::const_iterator> smatch;
  std::smatch smatch;
  if ( std::regex_match(official, smatch, regValues)){

    std::string textFuture= smatch.format(standardText);
    std::cout <<  std::setw(len) << std::left << "Future: " << textFuture << std::endl;

  }

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



Formatting with regex

In the function call `std::regex_replace(unofficial, regValues, standardText)` the text that matches the first and second capture groups of the regular expression `regValues` is extracted from the string `unofficial`. The placeholders `$1` and `$2` in the text `standardText` are then replaced by the extracted values. The strategy of `smatch.format(standardTest)` is similar but there is a difference:

The creation of the search results `smatch` are separated from their usage when formatting the string.

In addition to capture groups, C++ supports additional formatted escape sequences. We can use them in formatted strings:

Format escape sequence	Description
`$&`	Returns the total match (0th capture group).
`$$`	Returns $.
`$`` (backward tic)	Returns the text before the total match.
`$´` (forward tic)	Returns the text after the total match.
`$i`	Returns the ith capture group.

Format escape sequences

In the next lesson, we'll discuss the concept of iterators in regular expressions.