Lambda Functions

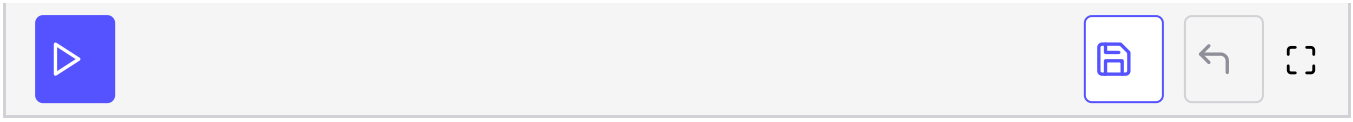
Lambda functions provide us all the functionality we need, on the go. They are faster than usual functions.

Lambda functions provide in-place functionality. The compiler gets a lot of insight and has therefore great optimization potential. Lambda functions can receive their arguments by value or by reference. They can capture their environment by value, by reference, and with C++14 by move.

```
#include <iostream>
#include <vector>
#include <algorithm>

int main(){
  std::vector<int> myVec{1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
  std::for_each(myVec.begin(), myVec.end(), [](int& i){
    i= i*i;
    std::cout << i << " ";
  }); // 1 4 9 16 25 36 49 64 81 100

  return 0;
}
```



> **Lambda functions should be your first choice**
> If the functionality of your callable is short and self-explanatory, use a lambda function. A lambda function is, in general, faster and easier to understand.