std::packaged_task

In this lesson, we will introduce std::packaged_task, which is used to parallelize big computer jobs.

`std::packaged_task` `pack` is a wrapper for a ***callable***, in order for it to be invoked asynchronously. By calling `pack.get_future()`, we get the associated future. Invoking the call operator on a pack (`pack()`) executes the `std::packaged_task` and, therefore, executes the callable.

Dealing with `std::packaged_task` usually consists of four steps:

I. Wrap the work:

```
std::packaged_task<int(int, int)> sumTask([](int a, int b){ return a + b; });
```

II. Create a future:

```
std::future<int> sumResult= sumTask.get_future();
```

III. Perform the calculation:

```
sumTask(2000, 11);
```

IV. Query the result:

```
sumResult.get();
```

In the next lesson, we will take a look at an example that goes through these four steps.