# Deduction Guides for packaged_task

Document #: D1038R0
Date: 2018-06-07
Project: Library Evolution Group
Reply-to: Marc Mutz <marc.mutz@kdab.com>

**Abstract**

`std::function` has deduction guides, but `std::packaged_task`, which is otherwise very similar, does not. This is surprising to users and we can think of no reason for the former to be treated differently from the latter. We therefore propose to add deduction guides for `packaged_task` with the same semantics as the existing ones for `function`.

## 1 Proposed Wording[1]

In `[futures.task]`, after the class definition and before the non-member `swap` function declaration, add:

```
template<class R, class... ArgTypes>
  packaged_task(R(*)(ArgTypes...)) -> packaged_task<R(ArgTypes...)>;

template<class F> packaged_task(F) -> packaged_task<see below>;
```

In `[futures.task.members]`, after the `packaged_task(F&&)` constructor description, add:

`template<class F> packaged_task(F) -> packaged_task<see below>;`

*Remarks:* This deduction guide participates in overload resolution only if `&F::operator()` is well-formed when treated as an unevaluated operand. In that case, if `decltype(&F::operator())` is of the form `R(G::*)(A...)  cv` $\&_{opt}$ `noexcept`$_{opt}$ for a class type `G`, then the deduced type is `packaged_task<R(A...)>`.

## 2 Acknowledgements

Barry Revzin found a bug in a draft version of this paper. All other errors are ours.

[1] NB: The intent is to re-use the wording that `std::function` has in the draft at the time of acceptance.