Deduction Guides for packaged_task

Document #:	PxxxxR0
Date:	2018-04-19
Project:	Programming Language C++ Library Evolution Group

Reply-to:	Marc Mutz <marc.mutz@kdab.com>

1 Introduction

`std::function` has deduction guides, but `std::packaged_task`, which is otherwise very similar to `std::function`, does not. We propose to add deduction guides for `packaged_task` that are compatible with the ones for `function`.

2 Impact on the Standard

Minimal. We propose to add deduction guides to `packaged_task`, which lacked them. This enables code that did not previously compile, but does not change the meaning of existing code.

3 Proposed Wording

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

In `[futures.task]`, 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...)>`.

4 Acknowledgements

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