Note that, since C++14, all the standard type traits defined in the `<type_traits>` header provide a corresponding alias template with the goal of reducing boilerplate code. For instance, C++14 introduces the `std::remove_reference_t` alias template for the C++11 `std::remove_reference` type trait:

```
typename std::remove_reference<int&>::type i0 = 5; // OK in both C++11 and C++14
std::remove_reference_t<int&> i1 = 5;              // OK in C++14
```

See Also

- "Trailing Return" (§1.1, p. 124) offers an alternative syntax for function declaration, which can help improve readability in type aliases and alias templates involving function types.
- "Inheriting ~~Ctors~~" (§2.1, p. 535) provides another meaning for the **using** keyword to allow base-class constructors to be invoked as part of the derived class.