

# Lifting Restrictions on **requires**-Expressions

Document #: WG21 P0266R0  
Date: 2016-02-12  
Project: JTC1.22.32 Programming Language C++  
Audience: EWG  $\Rightarrow$  CWG  
Reply to: Walter E. Brown <[webrown.cpp@gmail.com](mailto:webrown.cpp@gmail.com)>

---

## Contents

1	Introduction . . . . .	<b>1</b>	4	Bibliography . . . . .	<b>2</b>
2	Proposal . . . . .	<b>1</b>	5	Document history . . . . .	<b>2</b>
3	Proposed wording . . . . .	<b>2</b>			

---

## Abstract

This paper proposes to lift restrictions, currently imposed by the Concepts-Lite Working Draft [N4553], on the contexts in which a *requires-expression* is allowed to appear.

## 1 Introduction

The Concepts-Lite Working Draft, [N4553], provides wording for several new C++ language features. In our opinion, chief in importance among them are the *requires-clause* and *requires-expression*, each introduced by the new **requires** keyword. This paper seeks to lift certain restrictions imposed by the current wording on the use of a *requires-expression*.

Our proposal is similar to one made in our earlier paper [N4434]. Among other “tweaks” to the then-current draft of Concepts-Lite, we had proposed “to allow a concept name plus appropriate arguments . . . in any context where a **bool** value may reasonably appear.” In the year since, we have continued to conduct very extensive experimentation with all the Concept-Lite features as implemented for the forthcoming gcc6. Based on our application of these language features, we now believe it appropriate (and possibly even more important) to allow the analogous relaxation for a *requires-expression*, too.

## 2 Proposal

According to [N4553], “A *requires-expression* provides a concise way to express requirements on template arguments” [expr.prim.req]/1. We agree, but also believe there is even greater utility to such an expression, which is currently limited (by [expr.prim.req]/4) as to the contexts in which it is allowed to appear:

A *requires-expression* shall appear only within a concept definition (7.1.7), or within the *requires-clause* of a *template-declaration* (Clause 14) or function declaration (8.3.5).

**We propose to lift this restriction** and thereby to allow such a construct to appear in any context that permits a **bool**-valued expression.

In particular, easing the above-cited limitations will avoid such boilerplate circumlocutions as:

```

1  template< class T, class U >
2  constexpr bool
3     is_assignable_v = false;

5  template< class T, class U >
6     requires requires1( T&& t, U&& u )
7         { std::forward<T>(t) = std::forward<U>(u); }
8  constexpr bool
9     is_assignable_v<T, U> = true;

```

in favor of the far more straightforward:

```

1  template< class T, class U >
2  constexpr bool
3     is_assignable_v = requires( T&& t, U&& u )
4         { std::forward<T>(t) = std::forward<U>(u); };

```

We could write similar Concepts-Lite code today, but would need to phrase it as a (variable or function) **concept**. to do so. We believe that's not good enough, for a concept can't (yet) be evaluated outside a *requires-clause* or equivalent environment. We urge the adoption of this proposal to allow a *requires-expression* the maximum possible utility.

### 3 Proposed wording

Modify subclause 5.1.4 [expr.prim.req] of WG21 draft [N4553] as indicated:

~~4 A *requires-expression* shall appear only within a concept definition (7.1.7), or within the *requires-clause* of a template declaration (Clause 14) or function declaration (8.3.5). [ Example: ... – end example ] [ Note: ... – end note ]~~

### 4 Bibliography

- [N4434] Walter E. Brown: "Tweaks to Streamline Concepts Lite Syntax." ISO/IEC JTC1/SC22/WG21 document N4434 (pre-Lenexa mailing), 2015-04-10.  
<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2015/n4434.pdf>.
- [N4553] Andrew Sutton: "Working Draft, C++ Extension for Concepts." ISO/IEC JTC1/SC22/WG21 document N4553 (post-Kona mailing), 2015-10-02.  
<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2015/n4553.pdf>.

### 5 Document history

Version	Date	Changes
1	2016-02-12	• Published as P0266R0.

<sup>1</sup>The apparent reduplication of this keyword is not an error: the first occurrence introduces a *requires-clause*, while the second introduces a *requires-expression*. This proposal will likely reduce the need for such stuttering.