

Iterator Concepts

Author: David Abrahams, Jeremy Siek, Thomas Witt
Contact: dave@boost-consulting.com, jsiek@osl.iu.edu, witt@styleadvisor.com
Organization: Boost Consulting, Indiana University Open Systems Lab, Zephyr Associates, Inc.
Date: 2004-01-27
Copyright: Copyright David Abrahams, Jeremy Siek, and Thomas Witt 2004. All rights reserved

abstract: The iterator concept checking classes provide a mechanism for a template to report better error messages when a user instantiates the template with a type that does not meet the requirements of the template.

For an introduction to using concept checking classes, see the documentation for the [boost::concept_check](#) library.

Reference

Iterator Access Concepts

- [*Readable Iterator*](#)
- [*Writable Iterator*](#)
- [*Swappable Iterator*](#)
- [*Lvalue Iterator*](#)

Iterator Traversal Concepts

- [*Incrementable Iterator*](#)
- [*Single Pass Iterator*](#)
- [*Forward Traversal*](#)
- [*Bidirectional Traversal*](#)
- [*Random Access Traversal*](#)

iterator_concepts.hpp Synopsis

```
namespace boost_concepts {  
  
    // Iterator Access Concepts  
  
    template <typename Iterator>  
    class ReadableIteratorConcept;  
  
    template <  
        typename Iterator  
        , typename ValueType = std::iterator_traits<Iterator>::value_type  
    >  
    class WritableIteratorConcept;
```

```

template <typename Iterator>
class SwappableIteratorConcept;

template <typename Iterator>
class LvalueIteratorConcept;

// Iterator Traversal Concepts

template <typename Iterator>
class IncrementableIteratorConcept;

template <typename Iterator>
class SinglePassIteratorConcept;

template <typename Iterator>
class ForwardTraversalConcept;

template <typename Iterator>
class BidirectionalTraversalConcept;

template <typename Iterator>
class RandomAccessTraversalConcept;

// Interoperability

template <typename Iterator, typename ConstIterator>
class InteroperableIteratorConcept;
}

```