liboqs-cpp

0.1

Generated by Doxygen 1.8.14

Contents

Chapter 1

liboqs-cpp

[work in progress] C++ bindings for liboqs

Header-only C++ wrapper for liboqs

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 impl_details Namespace Reference

Implementation details.

6.1.1 Detailed Description

Implementation details.

6.2 oqs Namespace Reference

Main namespace for the liboqs C++ wrapper.

Namespaces

- impl_details_

Classes

- class KEMs

  Singleton class, contains details about supported/enabled key exchange mechanisms (KEMs)
- class KeyEncapsulation

  Key encapsulation mechanisms.
- class MechanismNotEnabledError

  Cryptographic scheme not enabled.
- class MechanismNotSupportedError

  Cryptographic scheme not supported.
- class Signature

  Signature mechanisms.
- class Sigs

  Singleton class, contains details about supported/enabled signatures.

Typedefs

- using byte = std::uint8_t

  byte (unsigned)
- using bytes = std::vector< byte >

  vector of bytes (unsigned)

6.2.1 Detailed Description

Main namespace for the liboqs C++ wrapper.

6.2.2 Typedef Documentation

6.2.2.1 byte

```
using oqs::byte = typedef std::uint8_t
```

byte (unsigned)

6.2.2.2 bytes

```
using oqs::bytes = typedef std::vector<byte>
```

vector of bytes (unsigned)

6.3 oqs::impl_details_ Namespace Reference

Classes

- class Singleton

  Singleton class using CRTP pattern.

6.4 oqs_literals Namespace Reference

Functions

- oqs::bytes operator""_bytes (const char *c_str, std::size_t length)

  User-defined literal operator for converting C-style strings to oqs::bytes.

6.4.1 Function Documentation

6.4.1.1 operator""""_bytes()

```
oqs::bytes oqs_literals::operator""_bytes (
              const char * c_str,
              std::size_t length )
```

User-defined literal operator for converting C-style strings to oqs::bytes.

Note

The null terminator is not included

Parameters

c_str	C-style string
length	C-style string length (deduced automatically by the compiler)

Returns

The byte representation of the input C-style string

Chapter 7

Class Documentation

7.1 oqs::KeyEncapsulation::alg_details_ Struct Reference

KEM algorithm details.

Public Attributes

- std::string name
- std::string version
- std::size_t claimed_nist_level
- bool is_ind_cca
- std::size_t length_public_key
- std::size_t length_secret_key
- std::size_t length_ciphertext
- std::size_t length_shared_secret

7.1.1 Detailed Description

KEM algorithm details.

7.1.2 Member Data Documentation

7.1.2.1 claimed_nist_level

```
std::size_t oqs::KeyEncapsulation::alg_details_::claimed_nist_level
```

7.1.2.2 is_ind_cca

```
bool oqs::KeyEncapsulation::alg_details_::is_ind_cca
```

7.1.2.3 length_ciphertext

```
std::size_t oqs::KeyEncapsulation::alg_details_::length_ciphertext
```

7.1.2.4 length_public_key

```
std::size_t oqs::KeyEncapsulation::alg_details_::length_public_key
```

7.1.2.5 length_secret_key

```
std::size_t oqs::KeyEncapsulation::alg_details_::length_secret_key
```

7.1.2.6 length_shared_secret

```
std::size_t oqs::KeyEncapsulation::alg_details_::length_shared_secret
```

7.1.2.7 name

```
std::string oqs::KeyEncapsulation::alg_details_::name
```

7.1.2.8 version

```
std::string oqs::KeyEncapsulation::alg_details_::version
```

The documentation for this struct was generated from the following file:

- oqs_cpp.h

7.2 oqs::Signature::alg_details_ Struct Reference

Signature algorithm details.

Public Attributes

- std::string name
- std::string version
- std::size_t claimed_nist_level
- bool is_euf_cma
- std::size_t length_public_key
- std::size_t length_secret_key
- std::size_t length_signature

7.2.1 Detailed Description

Signature algorithm details.

7.2.2 Member Data Documentation

7.2.2.1 claimed_nist_level

```
std::size_t oqs::Signature::alg_details_::claimed_nist_level
```

7.2.2.2 is_euf_cma

```
bool oqs::Signature::alg_details_::is_euf_cma
```

7.2.2.3 length_public_key

```
std::size_t oqs::Signature::alg_details_::length_public_key
```

7.2.2.4 length_secret_key

```
std::size_t oqs::Signature::alg_details_::length_secret_key
```

7.2.2.5 length_signature

```
std::size_t oqs::Signature::alg_details_::length_signature
```

7.2.2.6 name

```
std::string oqs::Signature::alg_details_::name
```

7.2.2.7 version

```
std::string oqs::Signature::alg_details_::version
```

The documentation for this struct was generated from the following file:

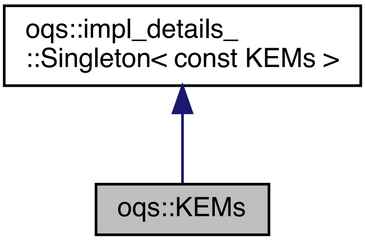
- oqs_cpp.h

7.3 oqs::KEMs Class Reference

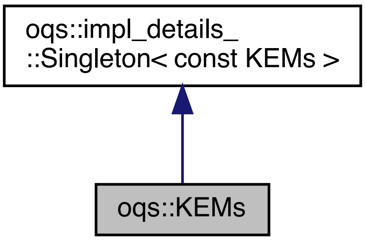
Singleton class, contains details about supported/enabled key exchange mechanisms (KEMs)

```
#include <oqs_cpp.h>
```

Inheritance diagram for oqs::KEMs:



Collaboration diagram for oqs::KEMs:



Static Public Member Functions

- static std::string get_KEM_name (std::size_t alg_id)

  KEM algorithm name.
- static bool is_KEM_enabled (const std::string &alg_name)

  Checks whether the KEM algorithm alg_name is enabled.
- static bool is_KEM_supported (const std::string &alg_name)

  Checks whether the KEM algorithm alg_name is supported.
- static const std::vector< std::string > & get_enabled_KEMs ()

  List of enabled KEM algorithms.
- static const std::vector< std::string > & get_supported_KEMs ()

  List of supported KEM algorithms.

Private Member Functions

- KEMs ()

  Private default constructor, initialization.

Static Private Attributes

- static std::size_t max_number_KEMs_ = ::OQS_KEM_alg_count()

  maximum number of supported KEMs
- static std::vector< std::string > supported_KEMs_

  list of supported KEMs
- static std::vector< std::string > enabled_KEMs_

  list of enabled KEMs

Friends

- class impl_details_::Singleton< const KEMs >

Additional Inherited Members

7.3.1 Detailed Description

Singleton class, contains details about supported/enabled key exchange mechanisms (KEMs)

7.3.2 Constructor & Destructor Documentation

7.3.2.1 KEMs()

```
oqs::KEMs::KEMs ( )  [inline], [private]
```

Private default constructor, initialization.

Note

Use oqs::KEMs::get_instance() to create an instance

7.3.3 Member Function Documentation

7.3.3.1 get_enabled_KEMs()

```
static const std::vector<std::string>& oqs::KEMs::get_enabled_KEMs ( )  [inline], [static]
```

List of enabled KEM algorithms.

Returns

List of enabled KEM algorithms

7.3.3.2 get_KEM_name()

```
static std::string oqs::KEMs::get_KEM_name (
            std::size_t alg_id )  [inline], [static]
```

KEM algorithm name.

Parameters

alg←↩ _id	Cryptographic algorithm numerical id

Returns

KEM algorithm name

7.3.3.3 get_supported_KEMs()

```
static const std::vector<std::string>& oqs::KEMs::get_supported_KEMs ( )  [inline], [static]
```

List of supported KEM algorithms.

Returns

List of supported KEM algorithms

7.3.3.4 is_KEM_enabled()

```
static bool oqs::KEMs::is_KEM_enabled (
             const std::string & alg_name )  [inline], [static]
```

Checks whether the KEM algorithm *alg_name* is enabled.

Parameters

alg_name	Cryptographic algorithm name

Returns

True if the KEM algorithm is enabled, false otherwise

7.3.3.5 is_KEM_supported()

```
static bool oqs::KEMs::is_KEM_supported (
             const std::string & alg_name )  [inline], [static]
```

Checks whether the KEM algorithm *alg_name* is supported.

Parameters

alg_name	Cryptographic algorithm name

Returns

True if the KEM algorithm is supported, false otherwise

7.3.4 Friends And Related Function Documentation

7.3.4.1 impl_details_::Singleton< const KEMs >

```
friend class impl_details_::Singleton< const KEMs > [friend]
```

7.3.5 Member Data Documentation

7.3.5.1 enabled_KEMs_

```
std::vector< std::string > oqs::KEMs::enabled_KEMs_ [static], [private]
```

list of enabled KEMs

7.3.5.2 max_number_KEMs_

```
std::size_t oqs::KEMs::max_number_KEMs_ = ::OQS_KEM_alg_count() [static], [private]
```

maximum number of supported KEMs

7.3.5.3 supported_KEMs_

```
std::vector< std::string > oqs::KEMs::supported_KEMs_ [static], [private]
```

list of supported KEMs

The documentation for this class was generated from the following file:

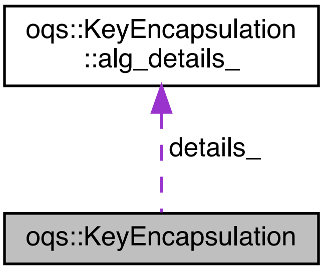
- oqs_cpp.h

7.4 oqs::KeyEncapsulation Class Reference

Key encapsulation mechanisms.

```
#include <oqs_cpp.h>
```

Collaboration diagram for oqs::KeyEncapsulation:



Classes

- struct alg_details_

  KEM algorithm details.

Public Member Functions

- KeyEncapsulation (const std::string &alg_name, const bytes &secret_key={})

  Constructs an instance of oqs::KeyEncapsulation.
- virtual ∼KeyEncapsulation ()

  Virtual default destructor.
- const alg_details_ & get_details () const

  KEM algorithm details.
- bytes generate_keypair ()

  Generate public key.
- bytes export_secret_key () const

  Export secret key.
- std::pair< bytes, bytes > encap_secret (const bytes &public_key) const

  Encapsulate secret.
- bytes decap_secret (const bytes &ciphertext) const

  Decapsulate secret.

Private Attributes

- const std::string alg_name_

    cryptographic algorithm name
- std::shared_ptr< ::OQS_KEM > kem_

    liboqs smart pointer to ::OQS_KEM
- bytes secret_key_ {}

    secret key
- struct oqs::KeyEncapsulation::alg_details_ details_

Friends

- std::ostream & operator<< (std::ostream &os, const alg_details_ &rhs)

    std::ostream extraction operator for the KEM algorithm details
- std::ostream & operator<< (std::ostream &os, const KeyEncapsulation &rhs)

    std::ostream extraction operator for oqs::KeyEncapsulation

7.4.1 Detailed Description

Key encapsulation mechanisms.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 KeyEncapsulation()

```
oqs::KeyEncapsulation::KeyEncapsulation (
             const std::string & alg_name,
             const bytes & secret_key = {} )  [inline]
```

Constructs an instance of oqs::KeyEncapsulation.

Parameters

alg_name	Cryptographic algorithm name
secret_key	Secret key (optional)

7.4.2.2 ∼KeyEncapsulation()

```
virtual oqs::KeyEncapsulation::∼KeyEncapsulation ( )  [inline], [virtual]
```

Virtual default destructor.

7.4.3 Member Function Documentation

7.4.3.1 decap_secret()

```
bytes oqs::KeyEncapsulation::decap_secret (
             const bytes & ciphertext ) const  [inline]
```

Decapsulate secret.

Parameters

ciphertext	Ciphertext

Returns

Shared secret

7.4.3.2 encap_secret()

```
std::pair<bytes, bytes> oqs::KeyEncapsulation::encap_secret (
             const bytes & public_key ) const  [inline]
```

Encapsulate secret.

Parameters

public_key	Public key

Returns

Pair consisting of 1) ciphertext, and 2) shared secret

7.4.3.3 export_secret_key()

```
bytes oqs::KeyEncapsulation::export_secret_key ( ) const  [inline]
```

Export secret key.

Returns

Secret key

7.4.3.4 generate_keypair()

```
bytes oqs::KeyEncapsulation::generate_keypair ( )  [inline]
```

Generate public key.

Returns

Public key

7.4.3.5 get_details()

```
const alg_details_& oqs::KeyEncapsulation::get_details ( ) const  [inline]
```

KEM algorithm details.

Returns

KEM algorithm details

7.4.4 Friends And Related Function Documentation

7.4.4.1 operator<< [1/2]

```
std::ostream& operator<< (
            std::ostream & os,
            const alg_details_ & rhs )  [friend]
```

std::ostream extraction operator for the KEM algorithm details

Parameters

os	Output stream
rhs	Algorithm details instance

Returns

Reference to the output stream

7.4.4.2 operator<< [2/2]

```
std::ostream& operator<< (
            std::ostream & os,
            const KeyEncapsulation & rhs )  [friend]
```

std::ostream extraction operator for oqs::KeyEncapsulation

Parameters

os	Output stream
rhs	Key encapsulation instance

Returns

Reference to the output stream

7.4.5 Member Data Documentation

7.4.5.1 alg_name_

```
const std::string oqs::KeyEncapsulation::alg_name_  [private]
```

cryptographic algorithm name

7.4.5.2 details_

```
struct oqs::KeyEncapsulation::alg_details_ oqs::KeyEncapsulation::details_  [private]
```

7.4.5.3 kem_

```
std::shared_ptr<::OQS_KEM> oqs::KeyEncapsulation::kem_  [private]
```

Initial value:

```
{nullptr, [](::OQS_KEM* p) {
                                    ::OQS_KEM_free(p);
                                }}
```

liboqs smart pointer to ::OQS_KEM

7.4.5.4 secret_key_

```
bytes oqs::KeyEncapsulation::secret_key_ {}  [private]
```

secret key

The documentation for this class was generated from the following file:

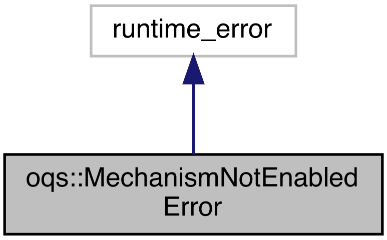
- oqs_cpp.h

7.5 oqs::MechanismNotEnabledError Class Reference

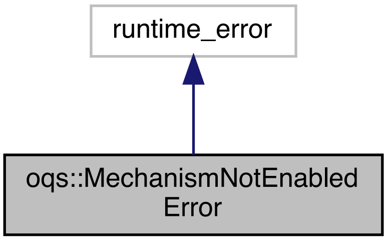
Cryptographic scheme not enabled.

```
#include <oqs_cpp.h>
```

Inheritance diagram for oqs::MechanismNotEnabledError:



Collaboration diagram for oqs::MechanismNotEnabledError:



Public Member Functions

- MechanismNotEnabledError (const std::string &alg_name)

  Constructor.

7.5.1 Detailed Description

Cryptographic scheme not enabled.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 MechanismNotEnabledError()

```
oqs::MechanismNotEnabledError::MechanismNotEnabledError (
            const std::string & alg_name )  [inline]
```

Constructor.

Parameters

alg_name	Cryptographic algorithm name

The documentation for this class was generated from the following file:

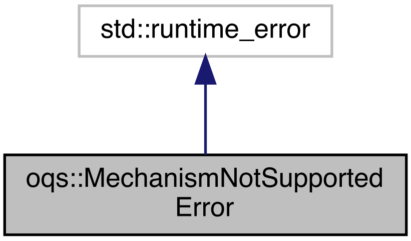
- oqs_cpp.h

7.6 oqs::MechanismNotSupportedError Class Reference

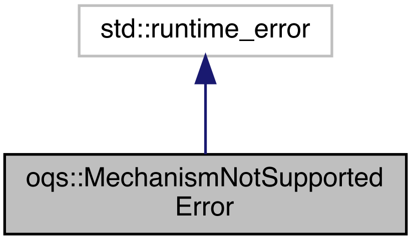
Cryptographic scheme not supported.

```
#include <oqs_cpp.h>
```

Inheritance diagram for oqs::MechanismNotSupportedError:



Collaboration diagram for oqs::MechanismNotSupportedError:



Public Member Functions

- MechanismNotSupportedError (const std::string &alg_name)

    Constructor.

7.6.1 Detailed Description

Cryptographic scheme not supported.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 MechanismNotSupportedError()

```
oqs::MechanismNotSupportedError::MechanismNotSupportedError (
            const std::string & alg_name )  [inline]
```

Constructor.

Parameters

alg_name	Cryptographic algorithm name

The documentation for this class was generated from the following file:

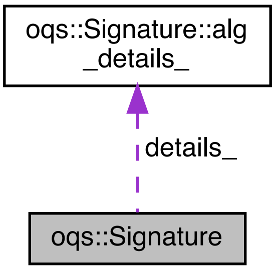
- oqs_cpp.h

7.7 oqs::Signature Class Reference

Signature mechanisms.

```
#include <oqs_cpp.h>
```

Collaboration diagram for oqs::Signature:



Classes

- struct alg_details_

  Signature algorithm details.

Public Member Functions

- Signature (const std::string &alg_name, const bytes &secret_key={})

  Constructs an instance of oqs::Signature.
- virtual ∼Signature ()

  Virtual default destructor.
- const alg_details_ & get_details () const

  Signature algorithm details.
- bytes generate_keypair ()

  Generate public key.
- bytes export_secret_key () const

  Export secret key.
- bytes sign (const bytes &message)

  Sign message.
- bool verify (const bytes &message, const bytes &signature, const bytes &public_key)

  Verify signature.

Private Attributes

- const std::string alg_name_

  cryptographic algorithm name
- std::shared_ptr< ::OQS_SIG > sig_

  liboqs smart pointer to ::OQS_SIG
- bytes secret_key_ {}

  secret key
- struct oqs::Signature::alg_details_ details_

Friends

- std::ostream & operator<< (std::ostream &os, const alg_details_ &rhs)

  std::ostream extraction operator for the signature algorithm details
- std::ostream & operator<< (std::ostream &os, const Signature &rhs)

  std::ostream extraction operator for oqs::Signature

7.7.1 Detailed Description

Signature mechanisms.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 Signature()

```
oqs::Signature::Signature (
            const std::string & alg_name,
            const bytes & secret_key = {} )  [inline]
```

Constructs an instance of oqs::Signature.

Parameters

alg_name	Cryptographic algorithm name
secret_key	Secret key (optional)

7.7.2.2 ∼Signature()

```
virtual oqs::Signature::∼Signature ( )  [inline], [virtual]
```

Virtual default destructor.

7.7.3 Member Function Documentation

7.7.3.1 export_secret_key()

```
bytes oqs::Signature::export_secret_key ( ) const  [inline]
```

Export secret key.

Returns

Secret key

7.7.3.2 generate_keypair()

```
bytes oqs::Signature::generate_keypair ( )  [inline]
```

Generate public key.

Returns

Public key

7.7.3.3 get_details()

```
const alg_details_& oqs::Signature::get_details ( ) const  [inline]
```

Signature algorithm details.

Returns

Signature algorithm details

7.7.3.4 sign()

```
bytes oqs::Signature::sign (
             const bytes & message )  [inline]
```

Sign message.

Parameters

message	Message

Returns

Message signature

7.7.3.5 verify()

```
bool oqs::Signature::verify (
             const bytes & message,
             const bytes & signature,
             const bytes & public_key )  [inline]
```

Verify signature.

Parameters

message	Message
signature	Signature
public_key	Public key

Returns

True if the signature is valid, false otherwise

7.7.4 Friends And Related Function Documentation

7.7.4.1 operator<< [1/2]

```
std::ostream& operator<< (
             std::ostream & os,
             const alg_details_ & rhs )  [friend]
```

std::ostream extraction operator for the signature algorithm details

Parameters

os	Output stream
rhs	Algorithm details

Returns

Reference to the output stream

7.7.4.2 operator<< [2/2]

```
std::ostream& operator<< (
            std::ostream & os,
            const Signature & rhs )  [friend]
```

std::ostream extraction operator for oqs::Signature

Parameters

os	Output stream
rhs	Signature instance

Returns

Reference to the output stream

7.7.5 Member Data Documentation

7.7.5.1 alg_name_

```
const std::string oqs::Signature::alg_name_  [private]
```

cryptographic algorithm name

7.7.5.2 details_

```
struct oqs::Signature::alg_details_ oqs::Signature::details_  [private]
```

7.7.5.3 secret_key_

```
bytes oqs::Signature::secret_key_ {}  [private]
```

secret key

7.7.5.4 sig_

```
std::shared_ptr<::OQS_SIG> oqs::Signature::sig_ [private]
```

Initial value:

```
{nullptr, [](::OQS_SIG* p) {
                                    ::OQS_SIG_free(p);
                                }}
```

liboqs smart pointer to ::OQS_SIG

The documentation for this class was generated from the following file:

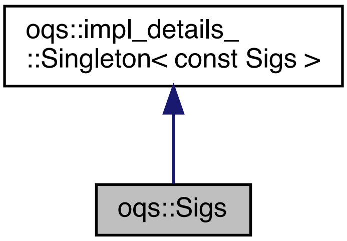
- oqs_cpp.h

7.8 oqs::Sigs Class Reference

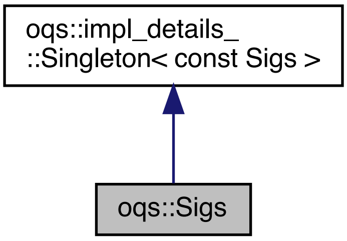
Singleton class, contains details about supported/enabled signatures.

```
#include <oqs_cpp.h>
```

Inheritance diagram for oqs::Sigs:



Collaboration diagram for oqs::Sigs:



Static Public Member Functions

- static std::string get_Sig_name (std::size_t alg_id)

  Signature algorithm name.
- static bool is_Sig_enabled (const std::string &alg_name)

  Checks whether the signature algorithm alg_name is enabled.
- static bool is_Sig_supported (const std::string &alg_name)

  Checks whether the signature algorithm alg_name is supported.
- static const std::vector< std::string > & get_enabled_Sigs ()

  List of enabled signature algorithms.
- static const std::vector< std::string > & get_supported_Sigs ()

  List of supported signature algorithms.

Private Member Functions

- Sigs ()

  Private default constructor, initialization.

Static Private Attributes

- static std::size_t max_number_Sigs_ = ::OQS_SIG_alg_count()

  maximum number of supported signatures
- static std::vector< std::string > supported_Sigs_

  list of supported signatures
- static std::vector< std::string > enabled_Sigs_

  list of enabled signatures

Friends

- class impl_details_::Singleton< const Sigs >

Additional Inherited Members

7.8.1 Detailed Description

Singleton class, contains details about supported/enabled signatures.

7.8.2 Constructor & Destructor Documentation

7.8.2.1 Sigs()

```
oqs::Sigs::Sigs ( )  [inline], [private]
```

Private default constructor, initialization.

Note

Use oqs::Sigs::get_instance() to create an instance

7.8.3 Member Function Documentation

7.8.3.1 get_enabled_Sigs()

```
static const std::vector<std::string>& oqs::Sigs::get_enabled_Sigs ( )  [inline], [static]
```

List of enabled signature algorithms.

Returns

List of enabled signature algorithms

7.8.3.2 get_Sig_name()

```
static std::string oqs::Sigs::get_Sig_name (
             std::size_t alg_id )  [inline], [static]
```

Signature algorithm name.

Parameters

alg←_id	Cryptographic algorithm numerical id

Returns

Signature algorithm name

7.8.3.3 get_supported_Sigs()

```
static const std::vector<std::string>& oqs::Sigs::get_supported_Sigs ( )  [inline], [static]
```

List of supported signature algorithms.

Returns

List of supported signature algorithms

7.8.3.4 is_Sig_enabled()

```
static bool oqs::Sigs::is_Sig_enabled (
             const std::string & alg_name )  [inline], [static]
```

Checks whether the signature algorithm *alg_name* is enabled.

Parameters

alg_name	Cryptographic algorithm name

Returns

True if the signature algorithm is enabled, false otherwise

7.8.3.5 is_Sig_supported()

```
static bool oqs::Sigs::is_Sig_supported (
             const std::string & alg_name )  [inline], [static]
```

Checks whether the signature algorithm *alg_name* is supported.

Parameters

alg_name	Cryptographic algorithm name

Returns

True if the signature algorithm is supported, false otherwise

7.8.4 Friends And Related Function Documentation

7.8.4.1 impl_details_::Singleton< const Sigs >

```
friend class impl_details_::Singleton< const Sigs >  [friend]
```

7.8.5 Member Data Documentation

7.8.5.1 enabled_Sigs_

```
std::vector< std::string > oqs::Sigs::enabled_Sigs_  [static], [private]
```

list of enabled signatures

7.8.5.2 max_number_Sigs_

```
std::size_t oqs::Sigs::max_number_Sigs_ = ::OQS_SIG_alg_count()  [static], [private]
```

maximum number of supported signatures

7.8.5.3 supported_Sigs_

```
std::vector< std::string > oqs::Sigs::supported_Sigs_  [static], [private]
```

list of supported signatures

The documentation for this class was generated from the following file:

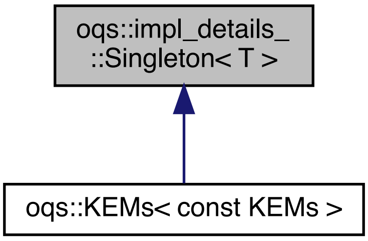
- oqs_cpp.h

7.9 oqs::impl_details_::Singleton< T > Class Template Reference

Singleton class using CRTP pattern.

```
#include <oqs_cpp.h>
```

Inheritance diagram for oqs::impl_details_::Singleton< T >:



Static Public Member Functions

- static T & get_instance () noexcept(std::is_nothrow_constructible< T >::value)

  Singleton instance (thread-safe) via CRTP pattern.

Protected Member Functions

- Singleton () noexcept=default
- Singleton (const Singleton &)=delete
- Singleton & operator= (const Singleton &)=delete
- virtual ∼Singleton ()=default

7.9.1 Detailed Description

template<typename T>
class oqs::impl_details_::Singleton< T >

Singleton class using CRTP pattern.

Template Parameters

T	Class type of which instance will become a Singleton

7.9.2 Constructor & Destructor Documentation

7.9.2.1 Singleton() [1/2]

```
template<typename T>
oqs::impl_details_::Singleton< T >::Singleton ( )  [protected], [default], [noexcept]
```

7.9.2.2 Singleton() [2/2]

```
template<typename T>
oqs::impl_details_::Singleton< T >::Singleton (
             const Singleton< T > &  )  [protected], [delete]
```

7.9.2.3 ∼Singleton()

```
template<typename T>
virtual oqs::impl_details_::Singleton< T >::∼Singleton ( )  [protected], [virtual], [default]
```

7.9.3 Member Function Documentation

7.9.3.1 get_instance()

```
template<typename T>
static T& oqs::impl_details_::Singleton< T >::get_instance ( )  [inline], [static], [noexcept]
```

Singleton instance (thread-safe) via CRTP pattern.

Note

Code from `https://github.com/vsoftco/qpp/blob/master/include/internal/classes/singleton h`

Returns

Singleton instance

7.9.3.2 operator=()

```
template<typename T>
Singleton& oqs::impl_details_::Singleton< T >::operator= (
            const Singleton< T > &  )  [protected], [delete]
```

The documentation for this class was generated from the following file:

- oqs_cpp.h

Chapter 8

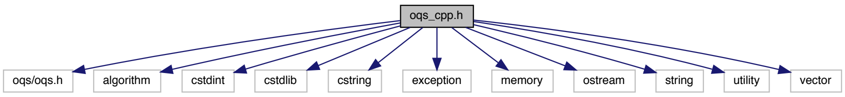
File Documentation

8.1 oqs_cpp.h File Reference

Main header file for the liboqs C++ wrapper.

```
#include <oqs/oqs.h>
#include <algorithm>
#include <cstdint>
#include <cstdlib>
#include <cstring>
#include <exception>
#include <memory>
#include <ostream>
#include <string>
#include <utility>
#include <vector>
```

Include dependency graph for oqs_cpp.h:



Classes

- class oqs::impl_details_::Singleton< T >

  Singleton class using CRTP pattern.
- class oqs::MechanismNotSupportedError

  Cryptographic scheme not supported.
- class oqs::MechanismNotEnabledError

  Cryptographic scheme not enabled.
- class oqs::KEMs

  Singleton class, contains details about supported/enabled key exchange mechanisms (KEMs)
- class oqs::KeyEncapsulation

  Key encapsulation mechanisms.

- struct oqs::KeyEncapsulation::alg_details_

  KEM algorithm details.
- class oqs::Sigs

  Singleton class, contains details about supported/enabled signatures.
- class oqs::Signature

  Signature mechanisms.
- struct oqs::Signature::alg_details_

  Signature algorithm details.

Namespaces

- oqs

  Main namespace for the liboqs C++ wrapper.
- impl_details

  Implementation details.
- oqs::impl_details_
- oqs_literals

Typedefs

- using oqs::byte = std::uint8_t

  byte (unsigned)
- using oqs::bytes = std::vector< byte >

  vector of bytes (unsigned)

Functions

- std::ostream & operator<< (std::ostream &os, const oqs::bytes &rhs)
- std::ostream & operator<< (std::ostream &os, const std::vector< std::string > &rhs)
- oqs::bytes oqs_literals::operator""_bytes (const char ∗c_str, std::size_t length)

  User-defined literal operator for converting C-style strings to oqs::bytes.

8.1.1 Detailed Description

Main header file for the liboqs C++ wrapper.

8.1.2 Function Documentation

8.1.2.1 operator<<() [1/2]

```
std::ostream& operator<< (
             std::ostream & os,
             const oqs::bytes & rhs )
```

::ostream extraction operator for oqs::bytes

Parameters

os	Output stream
rhs	Signature instance

Returns

Reference to the output stream

8.1.2.2 operator<<() [2/2]

```
std::ostream& operator<< (
            std::ostream & os,
            const std::vector< std::string > & rhs )
```

::ostream extraction operator for vectors of strings

Parameters

os	Output stream
rhs	Signature instance

Returns

Reference to the output stream

Index