

MyModules

Generated by Doxygen 1.7.6.1

Mon Jun 10 2013 01:14:24

Contents

1	Module Index	1
1.1	Modules	1
2	Namespace Index	3
2.1	Namespace List	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Module Documentation	9
5.1	Queue on list	9
5.2	Stack on list	10
5.3	Ring on list	11
6	Namespace Documentation	13
6.1	hzw Namespace Reference	13
6.1.1	Detailed Description	13
6.1.2	Typedef Documentation	14
6.1.2.1	FuncCompare	14
7	Class Documentation	15
7.1	hzw::Queue< Data > Class Template Reference	15
7.2	hzw::QueueException Class Reference	16
7.3	hzw::QueueVoid::QueueImplementation Class Reference	16
7.4	hzw::QueueVoid Class Reference	16

7.5	hzw::Ring< Data > Class Template Reference	17
7.5.1	Detailed Description	18
7.5.2	Constructor & Destructor Documentation	19
7.5.2.1	Ring	19
7.5.2.2	Ring	19
7.5.3	Member Function Documentation	19
7.5.3.1	contain	19
7.5.3.2	current	19
7.5.3.3	goForward	19
7.5.3.4	operator*	20
7.5.3.5	operator*= operator+	20
7.5.3.6	operator+	20
7.5.3.7	operator+=	20
7.5.3.8	operator-	20
7.5.3.9	operator-=	21
7.6	hzw::RingException Class Reference	21
7.6.1	Detailed Description	21
7.7	hzw::Stack< Data > Class Template Reference	21
7.8	hzw::StackException Class Reference	22
7.9	hzw::StackVoid::StackImplementation Class Reference	22
7.10	hzw::StackVoid Class Reference	22
8	File Documentation	25
8.1	/home/hz/MyProgramms/Gcc/MyModules/QueueOnList/queue.cpp File Reference	25
8.1.1	Detailed Description	25
8.2	/home/hz/MyProgramms/Gcc/MyModules/RingOnList/ring.cpp File Reference	26
8.2.1	Detailed Description	26
8.3	/home/hz/MyProgramms/Gcc/MyModules/RingOnList/ring.h File Reference	27
8.3.1	Detailed Description	27

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Queue on list	9
Stack on list	10
Ring on list	11

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

[hzw](#)

Harald zealot's werke [13](#)

Chapter 3

Class Index

3.1 Class List

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

hzw::Queue< Data >	15
hzw::QueueException	16
hzw::QueueVoid::QueueImplementation	16
hzw::QueueVoid	16
hzw::Ring< Data >	
Container class with current element and operations as on sets	17
hzw::RingException	
Exception that will be thrown while trying to read from empty Ring	21
hzw::Stack< Data >	21
hzw::StackException	22
hzw::StackVoid::StackImplementation	22
hzw::StackVoid	22

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

/home/hz/MyProgramms/Gcc/MyModules/include/hzw/ queue.h	??
/home/hz/MyProgramms/Gcc/MyModules/include/hzw/ stack.h	??
/home/hz/MyProgramms/Gcc/MyModules/QueueOnList/ queue.cpp	25
/home/hz/MyProgramms/Gcc/MyModules/QueueOnList/ queue.h	??
/home/hz/MyProgramms/Gcc/MyModules/RingOnList/ ring.cpp Implementation of classes RingVoid and RingImplementation	26
/home/hz/MyProgramms/Gcc/MyModules/RingOnList/ ring.h Declaration of classes Ring and RingVoid, implementation of Ring .	27
/home/hz/MyProgramms/Gcc/MyModules/StackOnList/ stack.h	??

Chapter 5

Module Documentation

5.1 Queue on list

Namespaces

- namespace [hzw](#)
harald zealot's werke

5.2 Stack on list

Namespaces

- namespace [hzw](#)
harald zealot's werke

5.3 Ring on list

Namespaces

- namespace [hzw](#)
harald zealot's werke

Chapter 6

Namespace Documentation

6.1 hzw Namespace Reference

harald zealot's werke

Classes

- class [QueueException](#)
- class [Queue](#)
- class [QueueVoid](#)
- class [StackException](#)
- class [Stack](#)
- class [StackVoid](#)
- class [RingException](#)
Exception that will be thrown while trying to read from empty [Ring](#).
- class [Ring](#)
Container class with current element and operations as on sets.
- class **RingVoid**

Typedefs

- typedef int(* [FuncCompare](#))(const void *, const void *)

6.1.1 Detailed Description

harald zealot's werke Namespace for all classes and function in the MyModules project.

6.1.2 Typedef Documentation

6.1.2.1 `typedef int(* hzw::FuncCompare)(const void *, const void *)`

Pointer to function with typical for compare function signature

Parameters

<i>in</i>	<i>a</i>	The memory area contains object a.
<i>in</i>	<i>b</i>	The memory area contains object b.

Returns

Integer value, that corresponds to some comparability.

Chapter 7

Class Documentation

7.1 hzw::Queue< Data > Class Template Reference

Public Member Functions

- **Queue** (const [Queue](#) &original)
- [Queue](#) & **operator=** (const [Queue](#) &roperand)
- void **clear** ()
- void **enqueue** (Data dt)
- Data **onFront** () const
- Data **onBack** () const
- void **dequeue** ()
- bool **isEmpty** () const
- **Queue** (const [Queue](#) &original)
- [Queue](#) & **operator=** (const [Queue](#) &roperand)
- void **clear** ()
- void **enqueue** (Data dt)
- Data **onFront** () const
- Data **onBack** () const
- void **dequeue** ()
- bool **isEmpty** () const

```
template<typename Data> class hzw::Queue< Data >
```

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

- /home/hz/MyProgramms/Gcc/MyModules/include/hzw/queue.h
- /home/hz/MyProgramms/Gcc/MyModules/QueueOnList/queue.h

7.2 hzw::QueueException Class Reference

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

- /home/hz/MyProgramms/Gcc/MyModules/include/hzw/queue.h
- /home/hz/MyProgramms/Gcc/MyModules/QueueOnList/queue.h

7.3 hzw::QueueVoid::QueueImplementation Class Reference

Classes

- struct **Node**

Public Member Functions

- **QueueImplementation** (const [QueueImplementation](#) &original)
- [QueueImplementation](#) & **operator=** (const [QueueImplementation](#) &roperand)
- void **clear** ()
- void **enqueue** (const void *dtAdress, int dtSize)
- void **onFront** (void *dtAdress) const
- void **onBack** (void *dtAdress) const
- void **dequeue** ()
- bool **isEmpty** () const

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

- /home/hz/MyProgramms/Gcc/MyModules/QueueOnList/[queue.cpp](#)

7.4 hzw::QueueVoid Class Reference

Classes

- class [QueueImplementation](#)

Public Member Functions

- **QueueVoid** (const [QueueVoid](#) &original)
- [QueueVoid](#) & **operator=** (const [QueueVoid](#) &roperand)
- void **clear** ()
- void **enqueue** (const void *dtAdress, int dtSize)
- void **onFront** (void *dtAdress) const
- void **onBack** (void *dtAdress) const
- void **dequeue** ()

- bool **isEmpty** () const
- **QueueVoid** (const [QueueVoid](#) &original)
- [QueueVoid](#) & **operator=** (const [QueueVoid](#) &roperand)
- void **clear** ()
- void **enqueue** (const void *dtAdress, int dtSize)
- void **onFront** (void *dtAdress) const
- void **onBack** (void *dtAdress) const
- void **dequeue** ()
- bool **isEmpty** () const

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

- /home/hz/MyProgramms/Gcc/MyModules/include/hzw/queue.h
- /home/hz/MyProgramms/Gcc/MyModules/QueueOnList/queue.h
- /home/hz/MyProgramms/Gcc/MyModules/QueueOnList/[queue.cpp](#)

7.5 hzw::Ring< Data > Class Template Reference

Container class with current element and operations as on sets.

```
#include <hzw/ring.h>
```

Public Member Functions

- [Ring](#) ()
Construct an empty [Ring](#).
- [Ring](#) (const Data &element)
Construct a [Ring](#) with the single element.
- [Ring](#) (const Data elements[], int count)
Construct a [Ring](#) by range of elements.
- [Ring](#) (const [Ring](#) &original)
Copy constructor.
- [Ring](#) & **operator=** (const [Ring](#) &rightOperand)
Assign operator.
- [~Ring](#) ()
Destructor.
- void [goForward](#) (int turn)
Move up pointer to the current element.
- Data [current](#) () const
Value of the current element.
- void [excludeCurrent](#) ()
Exclude the current element from the [Ring](#).

- `Ring< Data > operator+ (const Ring< Data > &rightOperand) const`
Union operator.
- `Ring< Data > operator- (const Ring< Data > &rightOperand) const`
Substract operator.
- `Ring< Data > operator* (const Ring< Data > &rightOperand) const`
Intersect operator.
- `Ring< Data > & operator+= (const Ring< Data > &rightOperand)`
Union assign operatot.
- `Ring< Data > & operator-= (const Ring< Data > &rightOperand)`
Substract assign operatot.
- `Ring< Data > & operator*= (const Ring< Data > &rightOperand)`
Intersect assign operatot.
- `bool isEmpty () const`
Predicate that is true when the Ring is an empty.
- `bool hasSingle () const`
Predicate that is true when the Ring has the one element only.
- `bool contain (Data sample) const`
Predicate that is true when the Ring contain the sample element.

7.5.1 Detailed Description

```
template<typename Data>class hzw::Ring< Data >
```

Container class with current element and operations as on sets.

Object which may contain uniform and comparable data-objects, that have copy constructor. The data-objects are arranged in close chain order. In every nonempty `Ring` exists the special selected element named current. The current selection can be moved up to any nonegative integer number. If the number is greater than count of elements, movement will be continued *ab initio*. It looks like the addition in modular ring in the mathematics very much. They may obtain and exclude current element. For two `Rings` are also defined union, intersection and substruction with the same semantics as on mathematiacal sets.

Purpose:

The main purpose of the `Ring` is to generate nonrepeated random number from previously determined set. `Ring` can be used also as usual set.

Template Parameters

<code>Data</code>	is any data type with <code><</code> and <code>==</code> comparsion operators.
-------------------	---

7.5.2 Constructor & Destructor Documentation

7.5.2.1 `template<typename Data> hzw::Ring< Data >::Ring (const Data & element)`
`[inline, explicit]`

Construct a [Ring](#) with the single element.

Parameters

<code>in</code>	<code><i>element</i></code>	
-----------------	-----------------------------	--

7.5.2.2 `template<typename Data> hzw::Ring< Data >::Ring (const Data elements[], int count)` `[inline]`

Construct a [Ring](#) by range of elements.

The duplicates in the range will be eliminated if exist.

Parameters

<code>in</code>	<code><i>elements</i></code>	of the range.
<code>in</code>	<code><i>count</i></code>	of elements in the range.

7.5.3 Member Function Documentation

7.5.3.1 `template<typename Data> bool hzw::Ring< T >::contain (Data sample) const`
`[inline]`

Predicate that is true when the [Ring](#) contain the sample element.

Parameters

<code>in</code>	<code><i>sample</i></code>	element presence of which will be examined.
-----------------	----------------------------	---

7.5.3.2 `template<typename T> T hzw::Ring< T >::current () const` `[inline]`

Value of the current element.

Exceptions

RingException	is thrown while trying to read from empty Ring .
-------------------------------	--

7.5.3.3 `template<typename T> void hzw::Ring< T >::goForward (int turn)`
`[inline]`

Move up pointer to the current element.

$pointer \equiv turn \bmod count$

Parameters

<code>in</code>	<code>turn</code>	is count of step
-----------------	-------------------	------------------

7.5.3.4 `template<typename Data> Ring< T > hzw::Ring< T >::operator* (const Ring< Data > & rightOperand) const [inline]`

Intersect operator.

$result = left \cap right$

7.5.3.5 `template<typename Data> Ring< T > & hzw::Ring< T >::operator*= (const Ring< Data > & rightOperand) [inline]`

Intersect assign operator.

See also

[operator*\(\)](#)

7.5.3.6 `template<typename Data> Ring< T > hzw::Ring< T >::operator+ (const Ring< Data > & rightOperand) const [inline]`

Union operator.

$result = left \cup right$

7.5.3.7 `template<typename Data> Ring< T > & hzw::Ring< T >::operator+= (const Ring< Data > & rightOperand) [inline]`

Union assign operator.

See also

[operator+\(\)](#)

7.5.3.8 `template<typename Data> Ring< T > hzw::Ring< T >::operator- (const Ring< Data > & rightOperand) const [inline]`

Subtract operator.

$result = left \setminus right$

7.5.3.9 `template<typename Data> Ring< T > & hzw::Ring< T >::operator= (const
Ring< Data > & rightOperand) [inline]`

Substract assign operatot.

See also

[operator-\(\)](#)

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

- `/home/hz/MyProgramms/Gcc/MyModules/RingOnList/ring.h`

7.6 hzw::RingException Class Reference

Exception that will be thrown while trying to read from empty [Ring](#).

```
#include <hzw/ring.h>
```

7.6.1 Detailed Description

Exception that will be thrown while trying to read from empty [Ring](#).

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

- `/home/hz/MyProgramms/Gcc/MyModules/RingOnList/ring.h`

7.7 hzw::Stack< Data > Class Template Reference

Public Member Functions

- **Stack** (const [Stack](#)< Data > &original)
- [Stack](#)< Data > & **operator=** (const [Stack](#)< Data > &roperand)
- void **clear** ()
- void **push** (Data dt)
- Data **onTop** () const
- void **pop** ()
- bool **isEmpty** () const
- **Stack** (const [Stack](#)< Data > &original)
- [Stack](#)< Data > & **operator=** (const [Stack](#)< Data > &roperand)
- void **clear** ()
- void **push** (Data dt)
- Data **onTop** () const
- void **pop** ()
- bool **isEmpty** () const

```
template<typename Data> class hzw::Stack< Data >
```

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

- /home/hz/MyProgramms/Gcc/MyModules/include/hzw/stack.h
- /home/hz/MyProgramms/Gcc/MyModules/StackOnList/stack.h

7.8 hzw::StackException Class Reference

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

- /home/hz/MyProgramms/Gcc/MyModules/include/hzw/stack.h
- /home/hz/MyProgramms/Gcc/MyModules/StackOnList/stack.h

7.9 hzw::StackVoid::StackImplementation Class Reference

Classes

- struct **Node**

Public Member Functions

- **StackImplementation** (const [StackImplementation](#) &original)
- [StackImplementation](#) & **operator=** (const [StackImplementation](#) &roperand)
- void **push** (const void *dtAdress, int dtSize)
- void **pop** ()
- void **onTop** (void *dtAdress) const
- bool **isEmpty** () const
- void **clear** ()

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

- /home/hz/MyProgramms/Gcc/MyModules/StackOnList/stack.cpp

7.10 hzw::StackVoid Class Reference

Classes

- class [StackImplementation](#)

Public Member Functions

- **StackVoid** (const [StackVoid](#) &original)
- [StackVoid](#) & **operator=** (const [StackVoid](#) &operand)
- void **push** (const void *dtAdress, int dtSize)
- void **pop** ()
- void **onTop** (void *dtAdress) const
- bool **isEmpty** () const
- void **clear** ()
- **StackVoid** (const [StackVoid](#) &original)
- [StackVoid](#) & **operator=** (const [StackVoid](#) &operand)
- void **push** (const void *dtAdress, int dtSize)
- void **pop** ()
- void **onTop** (void *dtAdress) const
- bool **isEmpty** () const
- void **clear** ()

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

- /home/hz/MyProgramms/Gcc/MyModules/include/hzw/stack.h
- /home/hz/MyProgramms/Gcc/MyModules/StackOnList/stack.h
- /home/hz/MyProgramms/Gcc/MyModules/StackOnList/stack.cpp

Chapter 8

File Documentation

8.1 /home/hz/MyProgramms/Gcc/MyModules/QueueOnList/queue.cpp File Reference

```
#include "queue.h"
```

Classes

- class `hzw::QueueVoid::QueueImplementation`
- struct `hzw::QueueVoid::QueueImplementation::Node`

Namespaces

- namespace `hzw`
harald zealot's werke

8.1.1 Detailed Description

Copyright (c) 2013 Alaksiej Piotr Stankievič (Alaksei Stankevich) All rights reserved.

Author: Alaksiej Stankievič

module: QueueOnList project: MyModules

implementation of classes QueueVoid and QueueImplementation

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Its copy is also available at the root directory of the project.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

8.2 /home/hz/MyProgramms/Gcc/MyModules/RingOnList/ring.cpp File Reference

implementation of classes RingVoid and RingImplementation

```
#include <cstring> #include "ring.h"
```

Classes

- class **hzw::RingVoid::RingImplementation**
- struct **hzw::RingVoid::RingImplementation::Node**

Namespaces

- namespace **hzw**
harald zealot's werke

8.2.1 Detailed Description

implementation of classes RingVoid and RingImplementation

Author

Alaksiej Stankievič aka Harald Zealot

Copyright

(c) 2013 Alaksiej Piotr Stankievič (Alaksei Stankevich). All rights reserved.
Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Its copy is also available at the root directory of the project.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR - CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

8.3 /home/hz/MyProgramms/Gcc/MyModules/RingOnList/ring.h - File Reference

declaration of classes Ring and RingVoid, implementation of Ring

```
#include <exception>
```

Classes

- class [hzw::RingException](#)
Exception that will be thrown while trying to read from empty [Ring](#).
- class [hzw::Ring< Data >](#)
Container class with current element and operations as on sets.
- class **[hzw::RingVoid](#)**

Namespaces

- namespace [hzw](#)
harald zealot's werke

Typedefs

- typedef int(* [hzw::FuncCompare](#))(const void *, const void *)

8.3.1 Detailed Description

declaration of classes Ring and RingVoid, implementation of Ring

Author

Alaksiej Stankievič aka Harald Zealot

Copyright

(c) 2013 Alaksiej Piotr Stankievič (Alaksei Stankevich). All rights reserved.
Licensed under the Apache License, Version 2.0 (the "License"); you may not use
this file except in compliance with the License. You may obtain a copy of the License
at

<http://www.apache.org/licenses/LICENSE-2.0>

Its copy is also available at the root directory of the project.

Unless required by applicable law or agreed to in writing, software distributed under
the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR -
CONDITIONS OF ANY KIND, either express or implied. See the License for the
specific language governing permissions and limitations under the License.