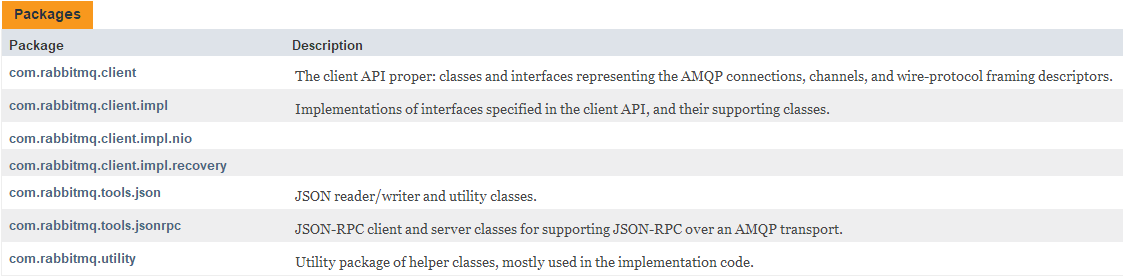
RabbitMQ--API介绍

<https://rabbitmq.github.io/rabbitmq-java-client/api/current/overview-summary.html>

# Package介绍

RabbitMQ Java Client 5.3.0 API ---- **Overview**



从package名可以看出，主要技术有NIO。

# 重点类介绍

## ConnectionFactory

java.lang.Object==> com.rabbitmq.client.ConnectionFactory

public class **ConnectionFactory** extends **Object** implements **Cloneable**

**Convenience factory class** to facilitate opening a **Connection** to a **RabbitMQ node**. Most connection and socket settings are configured using this factory. Some settings that apply to connections can also be configured here and will apply to all connections produced by this factory.

## 构造函数：创建ConnectionFactory实例，并配置参数方法

**ConnectionFactory​()**

只有这一个无参的构造函数，因此创建ConnectionFactory实例，只能是

ConnectionFactory factory = new ConnectionFactory();

关于连接RabbitMQ服务器的所有配置信息，都通过ConnectionFactory配置。

配置方式：

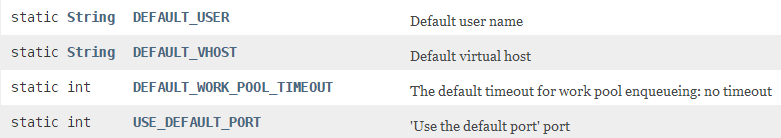
### setXxx()方法：设置单个参数

### setUri()方法：参数URI、String

### load()：参数Properties、Map、String

## 属性：ConnectionFactory的配置的默认值

默认的host、Vhost、port、User、Password等。



## 方法分类

### setXxx()

设置属性信息。如setHost()、setVirtualHost()、setPort()、setUsername等方法；

### setUri()

**void setUri​(String uriString)**

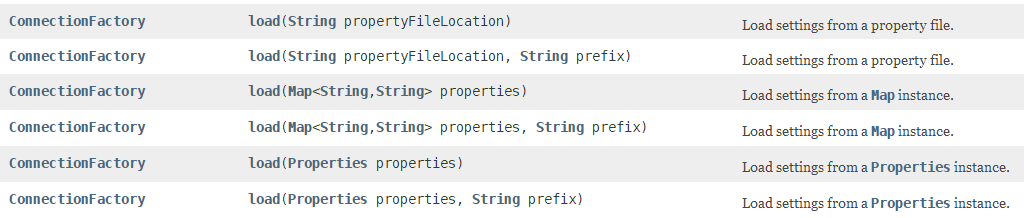
Convenience method for setting the fields in an AMQP URI: host, port, username, password and virtual host.

**void setUri​(URI uri)**

Convenience method for setting the fields in an AMQP URI: host, port, username, password and virtual host.

### getXxx()：与setXxx()对应，获取配置信息。

### load()：设置配置信息



### newConnection()：创建Connection对象。

Connection newConnection​()

**Create a new broker connection**.

Connection newConnection​(**Address[] addrs**)

Create a new broker connection, picking the first available address from the list.

Connection newConnection​(**Address[] addrs, String clientProvidedName**)

Create a new broker connection with a client-provided name, picking the first available address from the list.

Connection newConnection​(**AddressResolver addressResolver**)

Create a new broker connection, picking the first available address from the list provided by the AddressResolver.

Connection newConnection​(**String connectionName**)

Create a new broker connection.

Connection newConnection​(**ExecutorService executor**)

Create a new broker connection.

Connection newConnection​(**ExecutorService executor, Address[] addrs**)

Create a new broker connection, picking the first available address from the list.

Connection newConnection​(**ExecutorService executor, Address[] addrs, String clientProvidedName**)

Create a new broker connection with a client-provided name, picking the first available address from the list.

Connection newConnection​(**ExecutorService executor, AddressResolver addressResolver**) Create a new broker connection, picking the first available address from the list provided by the AddressResolver.

Connection newConnection​(**ExecutorService executor, AddressResolver addressResolver, String clientProvidedName**)

Create a new broker connection with a client-provided name, picking the first available address from the list provided by the AddressResolver.

Connection newConnection​(**ExecutorService executor, String connectionName**)

Create a new broker connection.

Connection newConnection​(ExecutorService executor, List<Address> addrs)

Create a new broker connection, picking the first available address from the list.

Connection newConnection​(ExecutorService executor, List<Address> addrs, String clientProvidedName)

Create a new broker connection with a client-provided name, picking the first available address from the list.

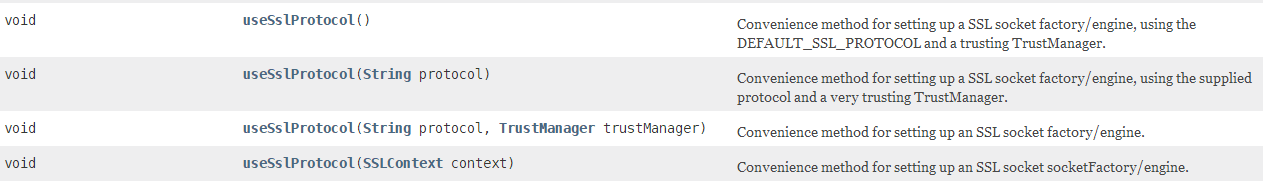
**Connection** newConnection​(List<Address> addrs)

Create a new broker connection, picking the first available address from the list.

**Connection** newConnection​(List<Address> addrs, String clientProvidedName)

Create a new broker connection with a client-provided name, picking the first available address from the list.

### useSslProtocol():使用**ssl协议**。



## 示例

ConnectionFactory factory = new ConnectionFactory();

factory.setHost("localhost");

**Connection connection = factory.newConnection();**

# ConnectionFactoryConfigurator配置器

public class **ConnectionFactoryConfigurator** extends Object

java.lang.Object ==> **com.rabbitmq.client.ConnectionFactoryConfigurator**

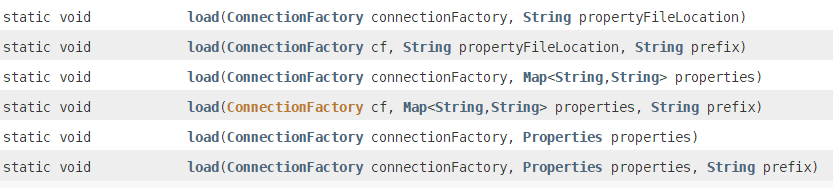
ConnectionFactoryConfigurator是一个辅助类，为ConnectionFactory的load()方法作为辅助。在ConnectionFactory的load方法中就是通过该ConnectionFactoryConfigurator类实现的。

**Helper class** to load ConnectionFactory settings from a property file. The authorised keys are the constants values in this class (e.g. USERNAME). **The property file/properties instance/map instance keys can have a prefix, the default being rabbitmq..** Property files can be loaded from the file system (the default), but also from the classpath, by using the classpath: prefix in the location. Client properties can be set by using the **client.properties.** prefix, e.g. client.properties.app.name. Default client properties and custom client properties are merged. To remove a default client property, set its key to an empty value.

既然是一个Helper class，那内部定义了一些静态属性和静态方法：

静态属性：就是一些配置信息。

静态方法：load方法。



# Connection接口

## 简单介绍

public interface **Connection** extends **ShutdownNotifier**, **Closeable**

**存在于包com.rabbitmq.client中。**

All Known Subinterfaces: **RecoverableConnection**

All Known Implementing Classes: **AMQConnection**, AutorecoveringConnection, RecoveryAwareAMQConnection

在**com.rabbitmq.client.impl**包中对Connection做了实现如**AMQConnection**。

## 创建Connection实例的示例：

To connect to a broker, fill in a **ConnectionFactory** and use a **ConnectionFactory** as follows:

ConnectionFactory factory = new ConnectionFactory();

factory.setHost(hostName);

factory.setPort(portNumber);

factory.setVirtualHost(virtualHost);

factory.setUsername(username);

factory.setPassword(password);

Connection conn = factory.newConnection();

// Then open a channel:

Channel channel = conn.createChannel();

Or, more **compactly**:(更简洁，更紧凑的使用)

ConnectionFactory factory = new ConnectionFactory();

factory.setUri("amqp://username:password@hostName:portNumber/virtualHost");

Connection conn = factory.newConnection();

Channel channel = conn.createChannel();

**Current implementations are thread-safe** for code at the client API level, and in fact thread-safe internally except for code within RPC calls.

## Connection接口方法

### createChannel方法

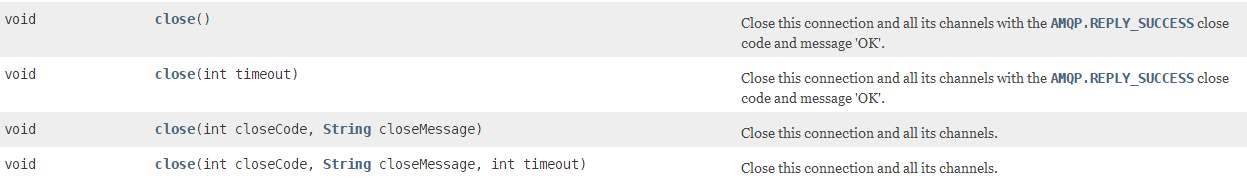
Channel **createChannel**​()

Create a new channel, using an internally allocated channel number.

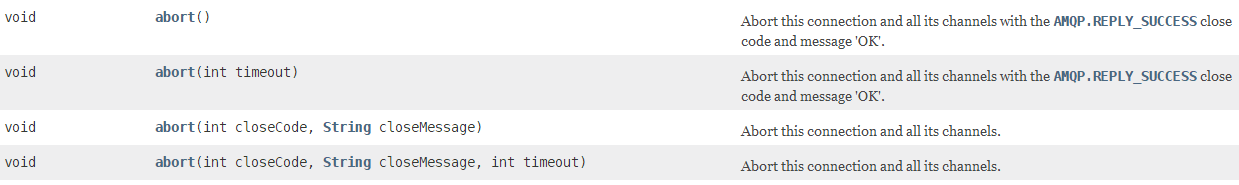
Channel **createChannel**​(int channelNumber)

Create a new channel, using the specified channel number if possible.

### close:继承了java.io. Closeable接口，所以具有close方法。



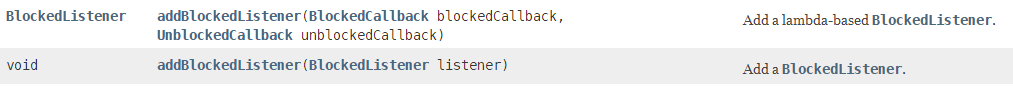
### abort



### getXxx

getAddress、getPort等。

### addBlockedListener和removeBlockedListener





# Channel接口

## 简单介绍

public interface **Channel** extends **ShutdownNotifier**, **AutoCloseable**

## Concurrency Considerations

**Channel instances** must not be shared between threads. Applications should prefer using a Channel per thread instead of sharing the same Channel across multiple threads. While some operations on channels are safe to invoke concurrently, some are not and will result in incorrect frame interleaving on the wire. **Sharing channels between threads will also interfere with Publisher Confirms.** As such, applications need to use a Channel per thread.

## 关键方法介绍

**RabbitMQ tutorials**, **RabbitMQ Java Client User Guide**

<http://www.rabbitmq.com/getstarted.html>

<http://www.rabbitmq.com/api-guide.html>

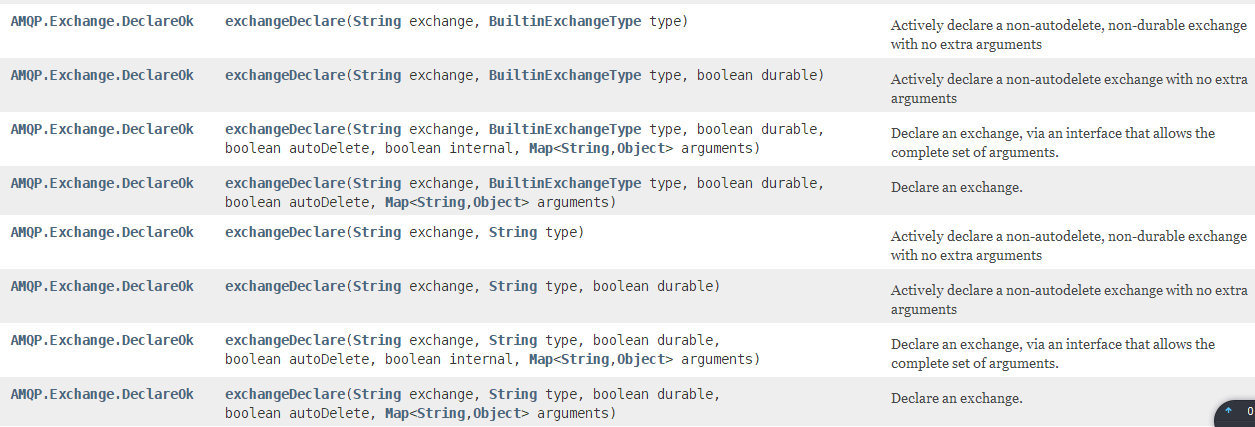
最重要的方法：就是对Exchange和Queue的操作；**basicXxx方法**

分别对应**exchangeDeclare、exchangeBind、exchangeUnbind、exchangeDelete**

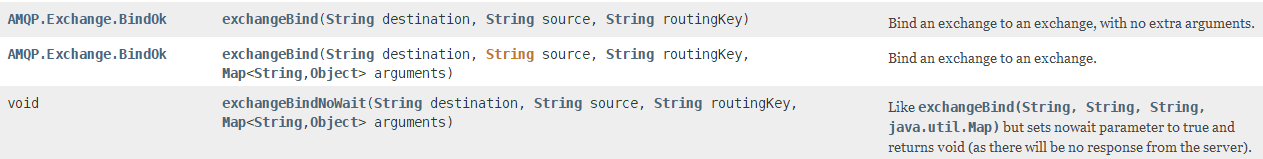
**queueDeclare、queueBind、queueUnbind、queueDelete**

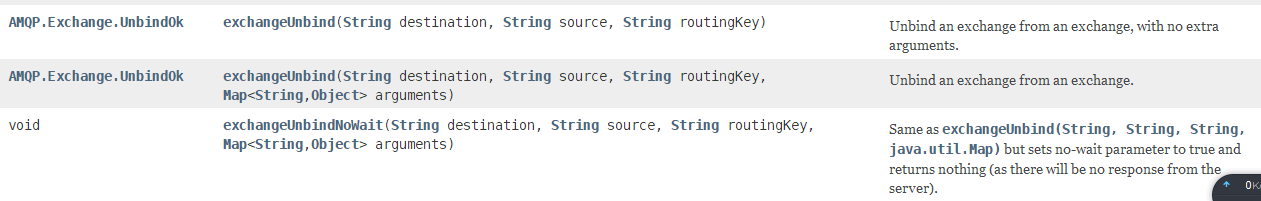
basicConsume、basicPublish、basicRecover、basicQos、basicReject。

### exchangeDeclare(): declare exchange

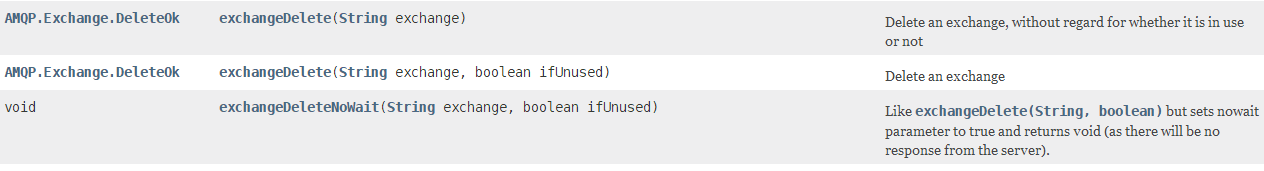


### exchangeBind与exchangeUnbind

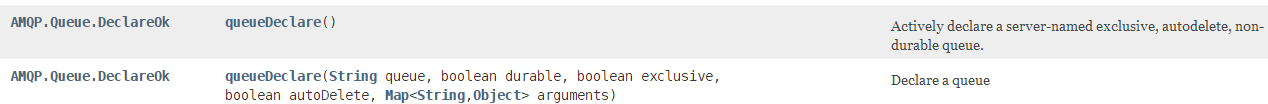




### exchangeDelete

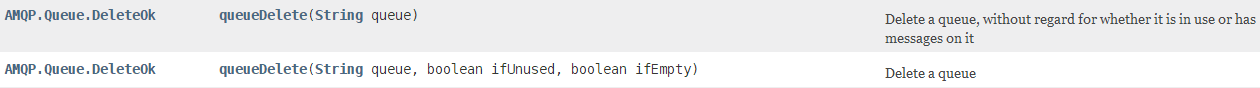


### queueDeclare与queueDeclarePassive

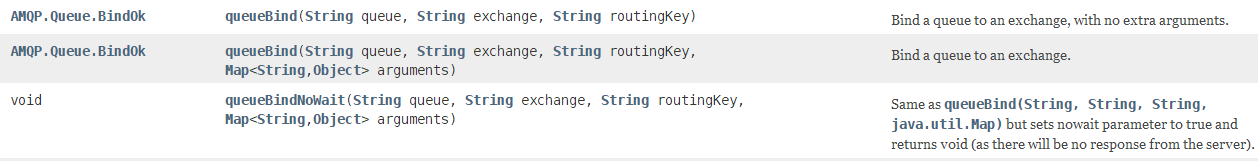


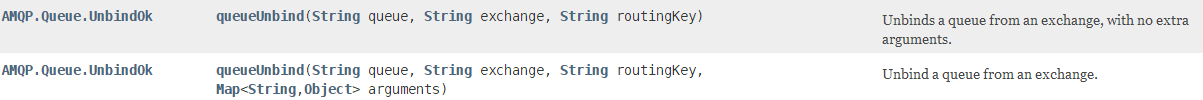


### queueDelete



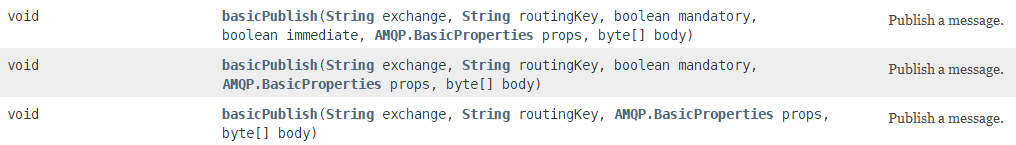
### queueBind与queueUnbind





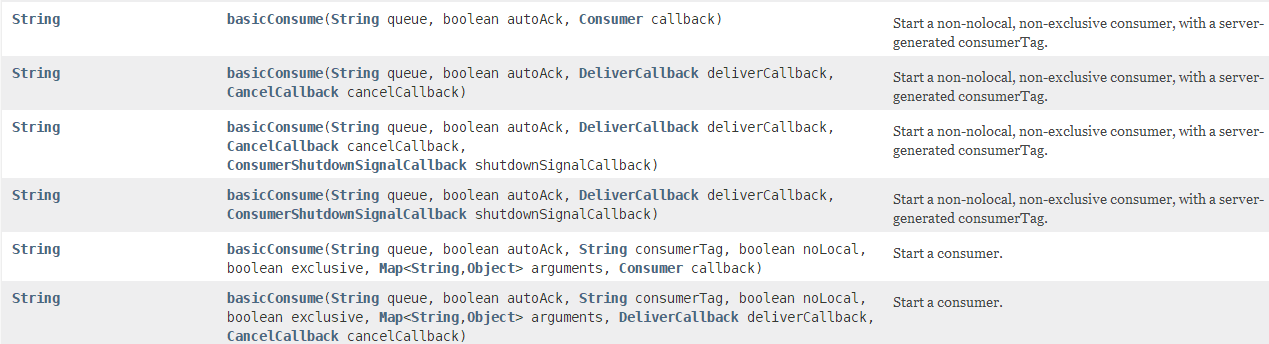
### basicPublish：对应在生产者发布消息

发布消息Publish a message.



### basicConsumer：对应在消费者Consumer处理消息

一般需要绑定一个Consumer对象。



# Consumer接口和DefaultConsumer实现类

## Consumer接口

Package **com.rabbitmq.client**

Interface Consumer

**Interface** for application callback objects **to receive notifications and messages** from a queue by subscription. Most implementations will subclass DefaultConsumer.

The methods of this interface are invoked in **a dispatch thread** which is separate from the Connection's thread. This allows Consumers to call Channel or Connection methods without causing a deadlock.

**The Consumers on a particular Channel are invoked serially on one or more dispatch threads.** Consumers should avoid executing long-running code because this will delay dispatch of messages to other Consumers on the same Channel. For a lambda-oriented syntax, use **DeliverCallback**, **CancelCallback**, and **ConsumerShutdownSignalCallback**.

方法介绍

### void **handleCancel**​(String consumerTag)

Called when the consumer is cancelled for reasons other than by a call to Channel.basicCancel(java.lang.String).

### void **handleCancelOk**​(String consumerTag)

Called when the consumer is cancelled by a call to Channel.basicCancel(java.lang.String).

### void **handleConsumeOk**​(String consumerTag)

Called when the consumer is registered by a call to any of the Channel.basicConsume(java.lang.String, com.rabbitmq.client.Consumer) methods.

### void handleDelivery​(String consumerTag, Envelope envelope, AMQP.BasicProperties properties, byte[] body)

Called when a basic.deliver is received for this consumer.

### void handleRecoverOk​(String consumerTag)

Called when a basic.recover-ok is received in reply to a basic.recover.

### void handleShutdownSignal​(String consumerTag, ShutdownSignalException sig)

Called when either the channel or the underlying connection has been shut down.

## DefaultConsumer

public class **DefaultConsumer** extends **Object** implements **Consumer**

Convenience class providing a default implementation of Consumer. We anticipate that most Consumer implementations will subclass this class.

构造方法：需要关联Channel。

DefaultConsumer​(**Channel channel**)

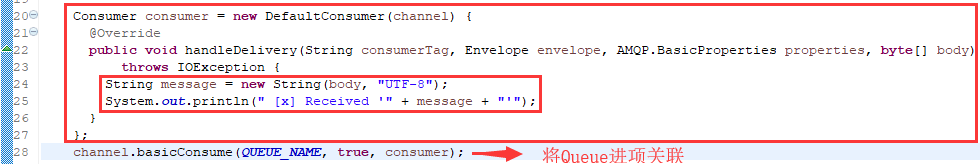
Constructs a new instance and records its association to the passed-in channel.



void handleDelivery​(String consumerTag, Envelope envelope, AMQP.BasicProperties properties, byte[] body)

No-op implementation of Consumer.handleDelivery(java.lang.String, com.rabbitmq.client.Envelope, com.rabbitmq.client.AMQP.BasicProperties, byte[]).

一般需要重写该方法：



# Address类

## 简单介绍

public class Addres extends Object



A representation of network addresses, i.e. **host/port pairs**, with some **utility functions** for **parsing address strings**.

host和port对；主机端口号对，提供些工具方法解析地址字符串。

## 构造方法

Address​(String host)

Construct an address from a host.

Address​(String host, int port)

Construct an address from a host name and port number.

## 一般方法

### equals​(Object obj)

### getHost或getPort

String getHost​()

Get the host name

int getPort​()

Get the port number

### 解析转换静态工具方法parseAddress：utility functions

static Address **parseAddress**​(String addressString)

Factory method: takes a formatted addressString string as construction parameter

static Address[] **parseAddresses**​(String addresses)

Array-based factory method: takes an array of formatted address strings as construction parameter

### int hashCode​()

# BuiltinExchangeType内置交换机类型

public enum **BuiltinExchangeType** extends **Enum**<BuiltinExchangeType>



Enum for built-in exchange types.

**BuiltinExchangeType**是一个枚举类Enum，内置交换机Exchange types。

对应四种类型：**DIRECT 、FANOUT 、HEADERS 、TOPIC**。

方法介绍

### String getType​()

### static **BuiltinExchangeType** valueOf​(String name)

Returns the enum constant of this type with the specified name.

### static **BuiltinExchangeType**[] values​()

Returns an array containing the constants of this enum type, in the order they are declared.