|  |  |  |
| --- | --- | --- |
| **AsynchronousFileChannel** | Liên quan | |
| An asynchronous channel for reading, writing, and manipulating a file.  An asynchronous file channel is created when a file is opened by invoking one of the [open](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BAsynchronousFileChannel.java%E2%98%83AsynchronousFileChannel%E2%98%82%E2%98%82open) methods defined by this class. The file contains a variable-length sequence of bytes that can be read and written and whose current size can be [queried](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BAsynchronousFileChannel.java%E2%98%83AsynchronousFileChannel%E2%98%82%E2%98%82size%E2%98%82). The size of the file increases when bytes are written beyond its current size; the size of the file decreases when it is [truncated](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BAsynchronousFileChannel.java%E2%98%83AsynchronousFileChannel%E2%98%82%E2%98%82truncate).  An asynchronous file channel does not have a *current position* within the file. Instead, the file position is specified to each read and write operation.  … | | 🄌**AsyncronousChannel**  ➊**Path[2]**  ➊**ExecutorService**  ➊**FileAttribute[1]**  ➊**OpenOption[2]**  ➊**FileSystemProvider[3]**  ➊**Set**  ➊**HashSet**  ➊**Collections**  ➊**FileLock** ➊**CompletionHandler**  ➊**Future**  ➊**ByteBuffer** |

|  |  |
| --- | --- |
| **FileLock** | Liên quan |
| A token representing a lock on a region of a file.  A file-lock object is created each time a lock is acquired on a file via one of the [lock](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileLock.java%E2%98%83FileLock%E2%98%82FileChannel%E2%98%82lock%E2%98%82long%E2%98%82long%E2%98%82boolean) or [tryLock](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileLock.java%E2%98%83FileLock%E2%98%82FileChannel%E2%98%82tryLock%E2%98%82long%E2%98%82long%E2%98%82boolean) methods of the [FileChannel](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileLock.java%E2%98%83FileLock%E2%98%82FileChannel) class, or the [lock](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileLock.java%E2%98%83FileLock%E2%98%82AsynchronousFileChannel%E2%98%82lock%E2%98%82long%E2%98%82long%E2%98%82boolean%E2%98%82Object%E2%98%82CompletionHandler) or [tryLock](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileLock.java%E2%98%83FileLock%E2%98%82AsynchronousFileChannel%E2%98%82tryLock%E2%98%82long%E2%98%82long%E2%98%82boolean) methods of the [AsynchronousFileChannel](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileLock.java%E2%98%83FileLock%E2%98%82AsynchronousFileChannel) class.  A file-lock object is initially valid. It remains valid until the lock is released by invoking the [release](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileLock.java%E2%98%83FileLock%E2%98%82%E2%98%82release) method, by closing the channel that was used to acquire it, or by the termination of the Java virtual machine, whichever comes first. The validity of a lock may be tested by invoking its [isValid](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileLock.java%E2%98%83FileLock%E2%98%82%E2%98%82isValid) method.  … | ➊**Channel**  ➊**FileChannel**  ➊**AsynchronousFileChannel** |

|  |  |
| --- | --- |
| **AsynchronousChannel** | Liên quan |
| A channel that supports asynchronous I/O operations. Asynchronous I/O operations will usually take one of two forms:   1. [Future](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BAsynchronousChannel.java%E2%98%83AsynchronousChannel%E2%98%82Future)<V> *operation*(*...*) 2. Future<V> *operation*(*...* A attachment, [CompletionHandler](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BAsynchronousChannel.java%E2%98%83AsynchronousChannel%E2%98%82CompletionHandler)<V,? super A> handler)   where *operation* is the name of the I/O operation (read or write for example), *V* is the result type of the I/O operation, and *A* is the type of an object attached to the I/O operation to provide context when consuming the result. The attachment is important for cases where a *state-less* CompletionHandler is used to consume the result of many I/O operations.  … | ➋**Channel** |

|  |  |
| --- | --- |
| **CompletionHandler** | Liên quan |
| A handler for consuming the result of an asynchronous I/O operation.  The asynchronous channels defined in this package allow a completion handler to be specified to consume the result of an asynchronous operation. The [completed](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BCompletionHandler.java%E2%98%83CompletionHandler%E2%98%82%E2%98%82completed) method is invoked when the I/O operation completes successfully. The [failed](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BCompletionHandler.java%E2%98%83CompletionHandler%E2%98%82%E2%98%82failed) method is invoked if the I/O operations fails. The [cancelled](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BCompletionHandler.java%E2%98%83CompletionHandler%E2%98%82%E2%98%82cancelled) method is invoked when the I/O operation is cancelled by invoking the [cancel](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BCompletionHandler.java%E2%98%83CompletionHandler%E2%98%82java.util.concurrent.Future%E2%98%82cancel) method. The implementations of these methods should complete in a timely manner so as to avoid keeping the invoking thread from dispatching to other completion handlers. | ➋**Channel** |

|  |  |
| --- | --- |
| **FileChannel** | Liên quan |
| A channel for reading, writing, mapping, and manipulating a file.  {@note revised} A file channel is a [SeekableByteChannel](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileChannel.java%E2%98%83FileChannel%E2%98%82SeekableByteChannel) that is connected to a file. It has a current *position* within its file which can be both [*queried*](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileChannel.java%E2%98%83FileChannel%E2%98%82%E2%98%82position%E2%98%82) and [*modified*](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileChannel.java%E2%98%83FileChannel%E2%98%82%E2%98%82position%E2%98%82long). The file itself contains a variable-length sequence of bytes that can be read and written and whose current [*size*](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileChannel.java%E2%98%83FileChannel%E2%98%82%E2%98%82size) can be queried. The size of the file increases when bytes are written beyond its current size; the size of the file decreases when it is [*truncated*](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BFileChannel.java%E2%98%83FileChannel%E2%98%82%E2%98%82truncate). The file may also have some associated *metadata* such as access permissions, content type, and last-modification time; this class does not define methods for metadata access.  … | ➋**AbstractInterruptibleChannel**  🄌**SeekableByteChannel**  🄌**GatheringByteChannel**  🄌**ScatteringByteChannel**  ➊**Path[2]**  🄌**Set**  🄌**Collections**  🄌**FileAttribute[10]**  🄌**OpenOption[2]**  🄌**FileSystemProvider[3]**  🄌**HashSet**  🄌**ByteBuffer**  🄌**WriteByteChannel**  🄌**ReadByteChannel**  🄌**MapMode[…]**  🄌**FileLock** |

|  |  |
| --- | --- |
| **MapMode** | Liên quan |
| A typesafe enumeration for file-mapping modes. |  |

|  |  |
| --- | --- |
| **SeekableByteChannel** | Liên quan |
| A byte channel that maintains a current *position* and allows the position to be changed.  A seekable byte channel is connected to an entity, typically a file, that contains a variable-length sequence of bytes that can be read and written. The current position can be [*queried*](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BSeekableByteChannel.java%E2%98%83SeekableByteChannel%E2%98%82%E2%98%82position%E2%98%82) and [*modified*](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BSeekableByteChannel.java%E2%98%83SeekableByteChannel%E2%98%82%E2%98%82position%E2%98%82long). The channel also provides access to the current *size* of the entity to which the channel is connected. The size increases when bytes are written beyond its current size; the size decreases when it is [*truncated*](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BSeekableByteChannel.java%E2%98%83SeekableByteChannel%E2%98%82%E2%98%82truncate).  The [position](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BSeekableByteChannel.java%E2%98%83SeekableByteChannel%E2%98%82%E2%98%82position%E2%98%82long) and [truncate](eclipse-javadoc:%E2%98%82=gaevfs/src%3Ccom.newatlanta.repackaged.java.nio.channels%7BSeekableByteChannel.java%E2%98%83SeekableByteChannel%E2%98%82%E2%98%82truncate) methods which do not otherwise have a value to return are specified to return the channel upon which they are invoked. This allows method invocations to be chained. Implementations of this interface should specialize the return type so that method invocations on the implementation class can be chained. | ➋**ByteChannel** |