

---

# **Part I. Red5 Server Documentation**

# 1. Annotation Type Anonymous

Annotation for classes that should be serialized without their classname. This should only be used if you know what you're doing and currently is only added to Red5 internal classes.

## 1.1. Synopsis

```
@Target(value=java.lang.annotation.ElementType.TYPE) @Retention(value=java.lang.annotation.RetentionType.RUNTIME)
}
```

# 2. Annotation Type DeclarePrivate

Annotation for public methods that should not be callable through RTMP, RTMPT or Remoting.

## 2.1. Synopsis

```
@Target(value=java.lang.annotation.ElementType.METHOD) @Retention(value=java.lang.annotation.RetentionType.RUNTIME)
}
```

# 3. Annotation Type DeclareProtected

Annotation for public methods that should be protected by a named permission when called through RTMP, RTMPT or Remoting.

## 3.1. Synopsis

```
@Target(value=java.lang.annotation.ElementType.METHOD) @Retention(value=java.lang.annotation.RetentionType.RUNTIME)
    public String permission;
}
```

## 3.2. permission

Permission required to execute method.

# 4. Annotation Type DontSerialize

Annotation for fields that should not be serialized when sending objects to a client.

## 4.1. Synopsis

```
@Target(value=java.lang.annotation.ElementType.FIELD) @Retention(value=java.lang.annotation.RetentionType.RUNTIME)
}
```



# 1. Class DataMessage

Message containing data update requests.

## 1.1. Synopsis

```
public class DataMessage extends org.red5.compatibility.flex.messaging.messages.AsyncMessage {  
    // Public Fields  
  
    public Object identity;  
  
    public int operation;  
  
    // Public Constructors  
  
    public DataMessage();  
  
    // Protected Methods  
  
    protected void addParameters(StringBuilder result);  
}
```

**Methods inherited from**

**org.red5.compatibility.flex.messaging.messages.AsyncMessage:** addParameters

**Methods inherited from**

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** toString

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait

**Fields inherited from**

**org.red5.compatibility.flex.messaging.messages.AsyncMessage:** correlationId

**Fields inherited from**

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** body, clientId, destination, headers, messageId, timestamp, timeToLive

## 1.2. addParameters(StringBuilder)

```
protected void addParameters(StringBuilder result);
```

# 2. Class SequencedMessage

Response to DataMessage requests.

## 2.1. Synopsis

```
public class SequencedMessage extends org.red5.compatibility.flex.messaging.messages.AsyncMessage {  
    // Public Fields  
  
    public String dataMessage;
```

```

    public long sequenceId ;

    public Object sequenceProxies ;

    public long sequenceSize ;

// Public Constructors

    public SequencedMessage();

// Protected Methods

    protected void addParameters(StringBuilder result);

}

```

**Methods inherited from****org.red5.compatibility.flex.messaging.messages.AsyncMessage:** addParameters**Methods inherited from****org.red5.compatibility.flex.messaging.messages.AbstractMessage:** toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , wait

**Fields inherited from****org.red5.compatibility.flex.messaging.messages.AsyncMessage:** correlationId**Fields inherited from****org.red5.compatibility.flex.messaging.messages.AbstractMessage:** body , clientId  
, destination , headers , messageId , timestamp , timeToLive**See Also**[org.red5.compatibility.flex.data.messages.DataMessage](#)

## 2.2. addParameters(StringBuilder)

```
protected void addParameters(StringBuilder result);
```

# 1. Class ArrayCollection

Flex ArrayCollection compatibility class.

## 1.1. Synopsis

```
public class ArrayCollection<T> extends java.util.ArrayList
    implements java.util.Collection, org.red5.io.amf3.IExternalizable {
    // Public Constructors

    public ArrayCollection();

    // Public Methods

    public void readExternal(org.red5.io.amf3.IDataInput input);

    public void writeExternal(org.red5.io.amf3.IDataOutput output);

}
```

**Methods inherited from java.util.ArrayList:** add, addAll, clear, clone, contains, ensureCapacity, get, indexOf, isEmpty, lastIndexOf, remove, removeRange, set, size, toArray, trimToSize

**Methods inherited from java.util.AbstractList:** equals, hashCode, iterator, listIterator, subList

**Methods inherited from java.util.AbstractCollection:** containsAll, removeAll, retainAll, toString

**Methods inherited from java.lang.Object:** finalize, getClass, notify, notifyAll, wait

**Fields inherited from java.util.AbstractList:** modCount

**See Also**

Adobe Livedocs (external) [<http://livedocs.adobe.com/flex/2/langref/mx/collections/ArrayCollection.html>]

## 1.2. readExternal(IDataInput)

```
public void readExternal(org.red5.io.amf3.IDataInput input);
```

**Specified by:** Method readExternal in interface IExternalizable

Load custom object from stream.

## 1.3. writeExternal(IDataOutput)

```
public void writeExternal(org.red5.io.amf3.IDataOutput output);
```

**Specified by:** Method writeExternal in interface IExternalizable

Store custom object to stream.

## 2. Class ObjectProxy

Flex ObjectProxy compatibility class.

### 2.1. Synopsis

```
public class ObjectProxy<T,V> implements, java.util.Map, org.red5.io.amf3.IExternalizable {
// Public Constructors

    public ObjectProxy();
}

    public ObjectProxy(java.util.Map<T, V> item);

// Public Methods

    public void clear();

    public boolean containsKey(Object name);

    public boolean containsValue(Object value);

    public java.util.Set<java.util.Map.Entry<T, V>> entrySet();

    public V get(Object name);

    public boolean isEmpty();

    public java.util.Set<T> keySet();

    public V put(T name,
                V value);

    public void putAll(java.util.Map values);

    public void readExternal(org.red5.io.amf3.IDataInput input);

    public V remove(Object name);

    public int size();

    public String toString();

    public java.util.Collection<V> values();

    public void writeExternal(org.red5.io.amf3.IDataOutput output);
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

#### See Also

Adobe Livedocs (external) [<http://livedocs.adobe.com/flex/2/langref/mx/utils/ObjectProxy.html>]

## 2.2. ObjectProxy()

```
public ObjectProxy();
```

Create new empty proxy.

## 2.3. ObjectProxy(Map<T, V>)

```
public ObjectProxy(java.util.Map<T, V> item);
```

Create proxy for given object.

### Parameters

item	object to proxy
------	-----------------

## 2.4. containsKey(Object)

```
public boolean containsKey(Object name);
```

**Specified by:** Method `containsKey` in interface `Map`

Check if proxied object has a given property.

### Parameters

name	
<i>return</i>	

## 2.5. get(Object)

```
public V get(Object name);
```

**Specified by:** Method `get` in interface `Map`

Return the value of a property.

### Parameters

name	
<i>return</i>	

## 2.6. put(T, V)

```
public V put(T name,
            V value);
```

Change a property of the proxied object.

### Parameters

<i>name</i>	
<i>value</i>	
<i>return</i>	

## 2.7. readExternal(IDataInput)

```
public void readExternal(org.red5.io.amf3.IDataInput input);
```

**Specified by:** Method `readExternal` in interface `IExternalizable`

Load custom object from stream.

## 2.8. remove(Object)

```
public V remove(Object name);
```

**Specified by:** Method `remove` in interface `Map`

Remove a property from the proxied object.

Parameters	
<i>name</i>	
<i>return</i>	

## 2.9. toString()

```
public String toString();
```

Return string representation of the proxied object.

Parameters	
<i>return</i>	

## 2.10. writeExternal(IDataOutput)

```
public void writeExternal(org.red5.io.amf3.IDataOutput output);
```

**Specified by:** Method `writeExternal` in interface `IExternalizable`

Store custom object to stream.

# 1. Class AbstractMessage

Base class for all Flex compatibility messages.

## 1.1. Synopsis

```
public class AbstractMessage implements, java.io.Serializable {  
    // Public Fields  
  
    public Object body ;  
  
    public String clientId ;  
  
    public String destination ;  
  
    public java.util.Map headers ;  
  
    public String messageId ;  
  
    public long timeToLive ;  
  
    public long timestamp ;  
  
    // Public Constructors  
  
    public AbstractMessage();  
  
    // Public Methods  
  
    public String toString();  
  
    // Protected Methods  
  
    protected void addParameters(StringBuilder result);  
}
```

**Direct known subclasses:** org.red5.compatibility.flex.messaging.messages.AsyncMessage

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. AbstractMessage()

```
public AbstractMessage();
```

Initialize default message fields.

## 1.3. addParameters(StringBuilder)

```
protected void addParameters(StringBuilder result);
```

Add message properties to string.

**Parameters**

<b>result</b>	StringBuilder to add properties to
---------------	------------------------------------

## 1.4. **toString()**

```
public String toString();
```

Return string representation of the message.

**Parameters**

<b>return</b>
---------------

## 2. Class AcknowledgeMessage

Flex compatibility message that is returned to the client.

### 2.1. Synopsis

```
public class AcknowledgeMessage extends, org.?red5.?compatibility.?flex.?messaging.?messages.?AsyncMessage
// Public Constructors

    public AcknowledgeMessage( );

}
```

**Methods inherited from**

**org.red5.compatibility.flex.messaging.messages.AsyncMessage:** addParameters

**Methods inherited from**

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , wait

**Fields inherited from**

**org.red5.compatibility.flex.messaging.messages.AsyncMessage:** correlationId

**Fields inherited from**

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** body , clientId  
, destination , headers , messageId , timestamp , timeToLive

## 3. Class AsyncMessage

Base class for for asynchronous Flex compatibility messages.

### 3.1. Synopsis

```
public class AsyncMessage extends, org.?red5.?compatibility.?flex.?messaging.?messages.?AbstractMessage
// Public Fields
```

```

    public String correlationId ;

// Public Constructors

    public AsyncMessage();

// Protected Methods

    protected void addParameters(StringBuilder result);

}

```

**Direct known subclasses:** org.red5.compatibility.flex.data.messages.SequencedMessage , org.red5.compatibility.flex.messaging.messages.AcknowledgeMessage , org.red5.compatibility.flex.messaging.messages.CommandMessage , org.red5.compatibility.flex.messaging.messages.ErrorMessage , org.red5.compatibility.messaging.RemotingMessage

#### Methods inherited from

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** addParameters ,  
toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , wait

#### Fields inherited from

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** body , clientId  
, destination , headers , messageId , timestamp , timeToLive

## 3.2. correlationId

```
public String correlationId ;
```

Id of message this message belongs to.

## 3.3. addParameters(StringBuilder)

```
protected void addParameters(StringBuilder result);
```

## 4. Class CommandMessage

Command message as sent by the `mx:RemoteObject` tag.

### 4.1. Synopsis

```

public class CommandMessage extends org.red5.compatibility.flex.messaging.messages.AsyncMessage
// Public Fields

    public String messageRefType ;

    public int operation ;

// Public Constructors

```

```

public CommandMessage();

// Protected Methods

protected void addParameters(StringBuilder result);

}

```

**Methods inherited from****org.red5.compatibility.flex.messaging.messages.AsyncMessage:** addParameters**Methods inherited from****org.red5.compatibility.flex.messaging.messages.AbstractMessage:** toString

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait

**Fields inherited from****org.red5.compatibility.flex.messaging.messages.AsyncMessage:** correlationId**Fields inherited from****org.red5.compatibility.flex.messaging.messages.AbstractMessage:** body, clientId, destination, headers, messageId, timestamp, timeToLive**See Also**

osflash documentation (external) [<http://osflash.org/documentation/amf3>], Adobe Livedocs (external) [<http://livedocs.adobe.com/flex/2/langref/mx/rpc/remoting/mxml/RemoteObject.html>]

## 4.2. operation

```
public int operation;
```

Command id to execute.

## 4.3. addParameters(StringBuilder)

```
protected void addParameters(StringBuilder result);
```

## 5. Class Constants

Constants for the flex compatibility messages.

### 5.1. Synopsis

```

public class Constants {
// Public Static Fields

public static final int DATA_OPERATION_SET = 10;

public static final int DATA_OPERATION_UPDATE = 7;

public static final int DATA_OPERATION_UPDATE_ATTRIBUTES = 3;

```

```

    public static final String HEADER_ENDPOINT = "DSEndpoint";

    public static final int OPERATION_AUTHENTICATION = 8;

    public static final int OPERATION_PING = 5;

    public static final int OPERATION_POLL = 2;

    public static final int OPERATION_REGISTER = 0;

// Public Constructors

    public Constants();

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 5.2. DATA\_OPERATION\_SET

```
public static final int DATA_OPERATION_SET = 10;
```

Set all attributes from a data message.

## 5.3. DATA\_OPERATION\_UPDATE

```
public static final int DATA_OPERATION_UPDATE = 7;
```

Update destination based on nested DataMessage packet.

## 5.4. DATA\_OPERATION\_UPDATE\_ATTRIBUTES

```
public static final int DATA_OPERATION_UPDATE_ATTRIBUTES = 3;
```

Update given attributes from a data message.

## 5.5. HEADER\_ENDPOINT

```
public static final String HEADER_ENDPOINT = "DSEndpoint";
```

Header field that holds the name of the endpoint.

## 5.6. OPERATION\_AUTHENTICATION

```
public static final int OPERATION_AUTHENTICATION = 8;
```

Operation id of authentication commands.

## 5.7. OPERATION\_PING

```
public static final int OPERATION_PING = 5;
```

Operation id of ping commands.

## 5.8. OPERATION\_POLL

```
public static final int OPERATION_POLL = 2;
```

Operation id of poll command.

## 5.9. OPERATION\_REGISTER

```
public static final int OPERATION_REGISTER = 0;
```

Operation id of register command.

# 6. Class ErrorMessage

Compatibility flex error message to be returned to the client.

## 6.1. Synopsis

```
public class ErrorMessage extends org.red5.compatibility.flex.messaging.messages.AsyncMessage
// Public Fields

    public Object extendedData ;

    public String faultCode ;

    public String faultDetail ;

    public String faultString ;

    public Object rootCause ;

// Public Constructors

    public ErrorMessage();

}
```

**Methods inherited from**

**org.red5.compatibility.flex.messaging.messages.AsyncMessage:** addParameters

**Methods inherited from**

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** toString

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait

**Fields inherited from**

**org.red5.compatibility.flex.messaging.messages.AsyncMessage:** correlationId

**Fields inherited from**

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** body, clientId, destination, headers, messageId, timestamp, timeToLive

## 7. Class RemotingMessage

Flex compatibility message that is sent by the `mx:RemoteObject` mxxml tag.

### 7.1. Synopsis

```
public class RemotingMessage extends, org.?red5.?compatibility.?flex.?messaging.?messages.?AsyncMessage
// Public Fields

    public String operation ;

    public String source ;

// Public Constructors

    public RemotingMessage() ;

// Protected Methods

    protected void addParameters(StringBuilder result) ;

}
```

#### Methods inherited from

**org.red5.compatibility.flex.messaging.messages.AsyncMessage:** `addParameters`

#### Methods inherited from

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** `toString`

**Methods inherited from java.lang.Object:** `clone` , `equals` , `finalize` , `getClass` , `hashCode` , `notify` , `notifyAll` , `wait`

#### Fields inherited from

**org.red5.compatibility.flex.messaging.messages.AsyncMessage:** `correlationId`

#### Fields inherited from

**org.red5.compatibility.flex.messaging.messages.AbstractMessage:** `body` , `clientId` , `destination` , `headers` , `messageId` , `timestamp` , `timeToLive`

#### See Also

osflash documentation (external) [<http://osflash.org/documentation/amf3>]

### 7.2. operation

```
public String operation ;
```

Method to execute.

### 7.3. source

```
public String source ;
```

Value of the `source` attribute of `mx:RemoteObject` that sent the message.

#### 7.4. addParameters(StringBuilder)

```
protected void addParameters(StringBuilder result);
```

# 1. Class BaseStreamableFileService

Base class for streamable file services.

## 1.1. Synopsis

```
public abstract class BaseStreamableFileService implements org.red5.io.IStreamableFileService {  
    // Public Constructors  
  
    public BaseStreamableFileService();  
  
    // Public Methods  
  
    public boolean canHandle(java.io.File file);  
  
    public abstract String getExtension();  
  
    public abstract String getPrefix();  
  
    public abstract IStreamableFile getStreamableFile(java.io.File file)  
        throws IOException;  
  
    public String prepareFilename(String name);  
}
```

**Direct known subclasses:** org.red5.io.flv.impl.FLVService , org.red5.io.mp3.impl.MP3Service

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 1.2. canHandle(File)

```
public boolean canHandle(java.io.File file);
```

**Specified by:** Method canHandle in interface IStreamableFileService

Check whether file can be used by file service, that is, it does exist and have valid extension

## 1.3. getExtension()

```
public abstract String getExtension();
```

**Specified by:** Method getExtension in interface IStreamableFileService

Getter for extension of file

## 1.4. getPrefix()

```
public abstract String getPrefix();
```

**Specified by:** Method `getPrefix` in interface `IStreamableFileService`

Getter for prefix. Prefix is used in filename composition to fetch real file name.

## 1.5. `getStreamableFile(File)`

```
public abstract IStreamableFile getStreamableFile(java.io.File file)
throws IOException;
```

**Specified by:** Method `getStreamableFile` in interface `IStreamableFileService`

Return streamable file reference. For FLV files returned streamable file already has generated metadata injected.

## 1.6. `prepareFilename(String)`

```
public String prepareFilename(String name);
```

**Specified by:** Method `prepareFilename` in interface `IStreamableFileService`

Prepair given string to conform filename requirements, for example, add extension to the end if missing.

## 2. Class `BufferType`

Buffer types (auto, direct or heap).

### 2.1. Synopsis

```
public final class BufferType extends java.lang.Enum {
// Public Static Fields

    public static final BufferType AUTO ;

    public static final BufferType DIRECT ;

    public static final BufferType HEAP ;

// Public Static Methods

    public static BufferType valueOf(String name);

    public static BufferType[] values();

}
```

**Methods inherited from `java.lang.Enum`:** `clone`, `compareTo`, `equals`, `finalize`, `getDeclaringClass`, `hashCode`, `name`, `ordinal`, `toString`, `valueOf`

**Methods inherited from `java.lang.Object`:** `getClass`, `notify`, `notifyAll`, `wait`

## 3. Class `CachingFileKeyFrameMetaCache`

```

public class CachingFileKeyFrameMetaCache extends, org.?red5.?io.?FileKeyFrameMetaCache {
// Public Constructors

    public CachingFileKeyFrameMetaCache();

// Public Methods

    public org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta loadKeyFrameMeta(java.io.File file);

    public void saveKeyFrameMeta(java.io.File file,
                                org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta meta);

    public void setMaxCacheEntry(int maxCacheEntry);

}

```

**Methods inherited from org.red5.io.FileKeyFrameMetaCache:** loadKeyFrameMeta , saveKeyFrameMeta

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 4. Class FileKeyFrameMetaCache

File-based keyframe metadata cache.

### 4.1. Synopsis

```

public class FileKeyFrameMetaCache implements, org.?red5.?io.?IKeyFrameMetaCache {
// Public Constructors

    public FileKeyFrameMetaCache();

// Public Methods

    public org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta loadKeyFrameMeta(java.io.File file);

    public void saveKeyFrameMeta(java.io.File file,
                                org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta meta);

}

```

**Direct known subclasses:** org.?red5.?io.?CachingFileKeyFrameMetaCache

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

### 4.2. loadKeyFrameMeta(File)

```
public org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta loadKeyFrameMeta(java.io.File file);
```

**Specified by:** Method loadKeyFrameMeta in interface IKeyFrameMetaCache

Load keyframe informations for the given file.

### 4.3. saveKeyFrameMeta(File, IKeyFrameDataAnalyzer.KeyFrameMeta)

```
public void saveKeyFrameMeta(java.io.File file,
                            org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta meta);
```

**Specified by:** Method saveKeyFrameMeta in interface IKeyFrameMetaCache

Store keyframe informations for the given file.

## 5. Interface IKeyFrameMetaCache

Interface defining a cache for keyframe metadata informations.

### 5.1. Synopsis

```
public interface IKeyFrameMetaCache {
    // Public Methods

    public org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta loadKeyFrameMeta(java.io.File file);

    public void saveKeyFrameMeta(java.io.File file,
                                org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta meta);

}
```

### 5.2. loadKeyFrameMeta(File)

```
public org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta loadKeyFrameMeta(java.io.File file);
```

Load keyframe informations for the given file.

#### Parameters

file	File to load informations for.
return	The keyframe informations or <code>null</code> if none exist.

### 5.3. saveKeyFrameMeta(File, IKeyFrameDataAnalyzer.KeyFrameMeta)

```
public void saveKeyFrameMeta(java.io.File file,
                            org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta meta);
```

Store keyframe informations for the given file.

#### Parameters

file	File to save informations for.
meta	Keyframe informations for this file.

## 6. Interface IStreamableFile

Interface represents streamable file with tag reader and writers (one for plain mode and one for append)

## 6.1. Synopsis

```
public interface IStreamableFile {
// Public Methods

    public ITagWriter getAppendWriter()
        throws IOException;

    public ITagReader getReader()
        throws IOException;

    public ITagWriter getWriter()
        throws IOException;

}
```

## 6.2. getAppendWriter()

```
public ITagWriter getAppendWriter()
    throws IOException;
```

Returns a Writer which is setup to append to the file.

### Parameters

<i>return</i>	the writer Tag writer used for append mode
---------------	--

java.io.IOException

I/O exception

## 6.3. getReader()

```
public ITagReader getReader()
    throws IOException;
```

Returns a reader to parse and read the tags inside the file.

### Parameters

<i>return</i>	the reader Tag reader
---------------	-----------------------

java.io.IOException

I/O exception

## 6.4. getWriter()

```
public ITagWriter getWriter()
    throws IOException;
```

Returns a writer that creates a new file or truncates existing contents.

**Parameters**

<i>return</i>	the writer Tag writer
---------------	-----------------------

java.io.IOException  
I/O exception

## 7. Interface IStreamableFileFactory

Scope service extension that provides method to get streamable file services set

### 7.1. Synopsis

```
public interface IStreamableFileFactory extends, org.?red5.?server.?api.?IScopeService {
    // Public Static Fields

    public static final String BEAN_NAME = "streamableFileFactory";

    // Public Methods

    public IStreamableFileService getService(java.io.File fp);

    public java.util.Set<org.red5.io.IStreamableFileService> getServices();

}
```

### 7.2. getServices()

```
public java.util.Set<org.red5.io.IStreamableFileService> getServices();
```

Getter for services

**Parameters**

<i>return</i>	Set of streamable file services
---------------	---------------------------------

## 8. Interface IStreamableFileService

Provides access to files that can be streamed.

### 8.1. Synopsis

```
public interface IStreamableFileService {
    // Public Methods

    public boolean canHandle(java.io.File file);

    public String getExtension();

    public String getPrefix();

    public IStreamableFile getStreamableFile(java.io.File file)
```

```

    throws IOException;

    public String prepareFilename(String name);

}

```

## 8.2. canHandle(File)

```
public boolean canHandle(java.io.File file);
```

Check whether file can be used by file service, that is, it does exist and have valid extension

### Parameters

file	File object
<i>return</i>	<code>true</code> if file exist and has valid extension, <code>false</code> otherwise

## 8.3. getExtension()

```
public String getExtension();
```

Getter for extension of file

### Parameters

<i>return</i>	File extension that is used
---------------	-----------------------------

## 8.4. getPrefix()

```
public String getPrefix();
```

Getter for prefix. Prefix is used in filename composition to fetch real file name.

### Parameters

<i>return</i>	Prefix
---------------	--------

## 8.5. getStreamableFile(File)

```

public IStreamableFile getStreamableFile(java.io.File file)
    throws IOException;

```

Return streamable file reference. For FLV files returned streamable file already has generated metadata injected.

### Parameters

file	File resource
<i>return</i>	Streamable file resource

`IOException`

Thrown if there were problems accessing given file

## 8.6. `prepareFilename(String)`

```
public String prepareFilename(String name);
```

Prepair given string to conform filename requirements, for example, add extension to the end if missing.

### Parameters

<code>name</code>	String to format
<i>return</i>	Correct filename

## 9. Interface ITag

A Tag represents the contents or payload of a streamable file.

### 9.1. Synopsis

```
public interface ITag extends, org.?red5.?io.?IoConstants {
// Public Methods

    public org.apache.mina.common.ByteBuffer getBody();

    public int getBodySize();

    public org.apache.mina.common.ByteBuffer getData();

    public byte getDataType();

    public int getPreviousTagSize();

    public int getTimestamp();

    public void setBody(org.apache.mina.common.ByteBuffer body);

    public void setBodySize(int size);

    public void setDataType(byte datatype);

    public void setPreviousTagSize(int size);

    public void setTimestamp(int timestamp);

}
```

### 9.2. `getBody()`

```
public org.apache.mina.common.ByteBuffer getBody();
```

Return the body ByteBuffer

#### Parameters

<i>return</i>	ByteBuffer Body as byte buffer
---------------	--------------------------------

### 9.3. getBodySize()

```
public int getBodySize();
```

Return the size of the body

#### Parameters

<i>return</i>	int Body size
---------------	---------------

### 9.4. getData()

```
public org.apache.mina.common.ByteBuffer getData();
```

Returns the data as a ByteBuffer

#### Parameters

<i>return</i>	ByteBuffer Data as byte buffer
---------------	--------------------------------

### 9.5. getDataType()

```
public byte getDataType();
```

Get the data type

#### Parameters

<i>return</i>	byte Data type as byte
---------------	------------------------

### 9.6. getPreviousTagSize()

```
public int getPreviousTagSize();
```

Returns previous tag size

#### Parameters

<i>return</i>	int Previous tag size
---------------	-----------------------

### 9.7. getTimestamp()

```
public int getTimestamp();
```

Return the timestamp

#### Parameters

<i>return</i>	int Timestamp
---------------	---------------

## 9.8. setBody(ByteBuffer)

```
public void setBody(org.apache.mina.common.ByteBuffer body);
```

Set the body ByteBuffer.

### Parameters

body	Body as ByteBuffer
------	--------------------

## 9.9. setBodySize(int)

```
public void setBodySize(int size);
```

Set the size of the body.

### Parameters

size	Body size
------	-----------

## 9.10. setDataType(byte)

```
public void setDataType(byte datatype);
```

Set the data type.

### Parameters

datatype	Data type
----------	-----------

## 9.11. setPreviousTagSize(int)

```
public void setPreviousTagSize(int size);
```

Set the size of the previous tag.

### Parameters

size	Previous tag size
------	-------------------

## 9.12. setTimestamp(int)

```
public void setTimestamp(int timestamp);
```

Set the timestamp.

### Parameters

timestamp	Timestamp
-----------	-----------

# 10. Interface ITagReader

```

public interface ITagReader {
// Public Methods

    public void close();

    public void decodeHeader();

    public long getBytesRead();

    public long getDuration();

    public IStreamableFile getFile();

    public int getOffset();

    public long getTotalBytes();

    public boolean hasMoreTags();

    public boolean hasVideo();

    public void position(long pos);

    public ITag readTag();

}

```

## 10.1. close()

```
public void close();
```

Closes the reader and free any allocated memory.

## 10.2. decodeHeader()

```
public void decodeHeader();
```

Decode the header of the stream;

## 10.3. getBytesRead()

```
public long getBytesRead();
```

Returns the amount of bytes read

### Parameters

<i>return</i>	long
---------------	------

## 10.4. getDuration()

```
public long getDuration();
```

Return length in seconds

#### Parameters

<i>return</i>	
---------------	--

### 10.5. getFile()

<code>public IStreamableFile getFile();</code>	
--	--

Return the file that is loaded.

#### Parameters

<i>return</i>	the file to be loaded
---------------	-----------------------

### 10.6. getOffset()

<code>public int getOffset();</code>	
--------------------------------------	--

Returns the offset length

#### Parameters

<i>return</i>	int
---------------	-----

### 10.7. getTotalBytes()

<code>public long getTotalBytes();</code>	
---	--

Get the total readable bytes in a file or ByteBuffer

#### Parameters

<i>return</i>	Total readable bytes
---------------	----------------------

### 10.8. hasMoreTags()

<code>public boolean hasMoreTags();</code>	
--	--

Returns a boolean stating whether the FLV has more tags

#### Parameters

<i>return</i>	boolean
---------------	---------

### 10.9. hasVideo()

<code>public boolean hasVideo();</code>	
---	--

Check if the reader also has video tags.

#### Parameters

<i>return</i>	
---------------	--

## 10.10. position(long)

public void position(long pos);
---------------------------------

Move the reader pointer to given position in file.

### Parameters

pos	File position to move to
-----	--------------------------

## 10.11. readTag()

public ITag readTag();
------------------------

Returns a Tag object

### Parameters

<i>return</i>	Tag
---------------	-----

# 11. Interface ITagWriter

Writes tags to FLV file

## 11.1. Synopsis

```
public interface ITagWriter {
// Public Methods

    public void close();

    public long getBytesWritten();

    public IStreamableFile getFile();

    public int getOffset();

    public void writeHeader()
        throws IOException;

    public boolean writeStream(byte[] b);

    public boolean writeTag(byte type,
        org.apache.mina.common.ByteBuffer data)
        throws IOException;

    public boolean writeTag(ITag tag)
        throws IOException;

}
```

## 11.2. close()

```
public void close();
```

Closes a Writer

## 11.3. getBytesWritten()

```
public long getBytesWritten();
```

Return the bytes written

### Parameters

<i>return</i>	Number of bytes written
---------------	-------------------------

## 11.4. getFile()

```
public IStreamableFile getFile();
```

Return the file that is written.

### Parameters

<i>return</i>	the File to be written
---------------	------------------------

## 11.5. getOffset()

```
public int getOffset();
```

Return the offset

### Parameters

<i>return</i>	Offset value
---------------	--------------

## 11.6. writeHeader()

```
public void writeHeader()
    throws IOException;
```

Writes the header bytes

IOException  
I/O exception

## 11.7. writeStream(byte[])

```
public boolean writeStream(byte[] b);
```

Write a Stream to disk using bytes

### Parameters

b	Array of bytes to write
<i>return</i>	true on success, false otherwise

IOException  
I/O exception

## 11.8. writeTag(byte, ByteBuffer)

```
public boolean writeTag(byte type,
                       org.apache.mina.common.ByteBuffer data)
throws IOException;
```

Write a Tag using bytes

Parameters	
type	Tag type
data	Byte data
<i>return</i>	true on success, false otherwise

IOException  
I/O exception

## 11.9. writeTag(ITag)

```
public boolean writeTag(ITag tag)
throws IOException;
```

Writes a Tag object

Parameters	
tag	Tag to write
<i>return</i>	true on success, false otherwise

IOException  
I/O exception

# 12. Interface IoConstants

Constants found in FLV files / streams.

## 12.1. Synopsis

```
public interface IoConstants {
// Public Static Fields

    public static final byte FLAG_CODEC_H263 = 2;

    public static final byte FLAG_CODEC_SCREEN = 3;
```

```
public static final byte FLAG_CODEC_VP6 = 4;

public static final byte FLAG_FORMAT_ADPCM = 1;

public static final byte FLAG_FORMAT_MP3 = 2;

public static final byte FLAG_FORMAT_NELLYMOSER = 6;

public static final byte FLAG_FORMAT_NELLYMOSER_8_KHZ = 5;

public static final byte FLAG_FORMAT_RAW = 0;

public static final byte FLAG_FRAMETYPE_DISPOSABLE = 3;

public static final byte FLAG_FRAMETYPE_INTERFRAME = 2;

public static final byte FLAG_FRAMETYPE_KEYFRAME = 1;

public static final byte FLAG_RATE_11_KHZ = 1;

public static final byte FLAG_RATE_22_KHZ = 2;

public static final byte FLAG_RATE_44_KHZ = 3;

public static final byte FLAG_RATE_5_5_KHZ = 0;

public static final byte FLAG_SIZE_16_BIT = 1;

public static final byte FLAG_SIZE_8_BIT = 0;

public static final byte FLAG_TYPE_MONO = 0;

public static final byte FLAG_TYPE_STEREO = 1;

public static final byte MASK_SOUND_FORMAT = -15;

public static final byte MASK_SOUND_RATE = 12;

public static final byte MASK_SOUND_SIZE = 2;

public static final byte MASK_SOUND_TYPE = 1;

public static final byte MASK_VIDEO_CODEC = 15;

public static final byte MASK_VIDEO_FRAMETYPE = -15;

public static final byte TYPE_AUDIO = 8;

public static final byte TYPE_METADATA = 18;

public static final byte TYPE_VIDEO = 9;

}
```

## 12.2. FLAG\_CODEC\_H263

```
public static final byte FLAG_CODEC_H263 = 2;
```

H263 codec flag

## 12.3. FLAG\_CODEC\_SCREEN

```
public static final byte FLAG_CODEC_SCREEN = 3;
```

Screen codec flag

## 12.4. FLAG\_CODEC\_VP6

```
public static final byte FLAG_CODEC_VP6 = 4;
```

On2 VP6 codec flag

## 12.5. FLAG\_FORMAT\_ADPCM

```
public static final byte FLAG_FORMAT_ADPCM = 1;
```

ADPCM format flag

## 12.6. FLAG\_FORMAT\_MP3

```
public static final byte FLAG_FORMAT_MP3 = 2;
```

MP3 format flag

## 12.7. FLAG\_FORMAT\_NELLYMOSER

```
public static final byte FLAG_FORMAT_NELLYMOSER = 6;
```

NellyMoser-encoded audio format flag

## 12.8. FLAG\_FORMAT\_NELLYMOSER\_8\_KHZ

```
public static final byte FLAG_FORMAT_NELLYMOSER_8_KHZ = 5;
```

8 KHz NellyMoser audio format flag

## 12.9. FLAG\_FORMAT\_RAW

```
public static final byte FLAG_FORMAT_RAW = 0;
```

Raw data format flag

## 12.10. FLAG\_FRAMETYPE\_DISPOSABLE

```
public static final byte FLAG_FRAMETYPE_DISPOSABLE = 3;
```

Disposable frame type flag

## 12.11. FLAG\_FRAMETYPE\_INTERFRAME

```
public static final byte FLAG_FRAMETYPE_INTERFRAME = 2;
```

Interframe flag. Interframes are created from keyframes rather than independent image

## 12.12. FLAG\_FRAMETYPE\_KEYFRAME

```
public static final byte FLAG_FRAMETYPE_KEYFRAME = 1;
```

Keyframe type flag

## 12.13. FLAG\_RATE\_11\_KHZ

```
public static final byte FLAG_RATE_11_KHZ = 1;
```

11 KHz rate flag

## 12.14. FLAG\_RATE\_22\_KHZ

```
public static final byte FLAG_RATE_22_KHZ = 2;
```

22 KHz rate flag

## 12.15. FLAG\_RATE\_44\_KHZ

```
public static final byte FLAG_RATE_44_KHZ = 3;
```

44 KHz rate flag

## 12.16. FLAG\_RATE\_5\_5\_KHZ

```
public static final byte FLAG_RATE_5_5_KHZ = 0;
```

5.5 KHz rate flag

## 12.17. FLAG\_SIZE\_16\_BIT

```
public static final byte FLAG_SIZE_16_BIT = 1;
```

16 bit flag size

## 12.18. FLAG\_SIZE\_8\_BIT

```
public static final byte FLAG_SIZE_8_BIT = 0;
```

8 bit flag size

## 12.19. FLAG\_TYPE\_MONO

```
public static final byte FLAG_TYPE_MONO = 0;
```

Mono mode

## 12.20. FLAG\_TYPE\_STEREO

```
public static final byte FLAG_TYPE_STEREO = 1;
```

Stereo mode

## 12.21. MASK\_SOUND\_FORMAT

```
public static final byte MASK_SOUND_FORMAT = -15;
```

Mask sound format (unsigned)

## 12.22. MASK\_SOUND\_RATE

```
public static final byte MASK_SOUND_RATE = 12;
```

Mask sound rate

## 12.23. MASK\_SOUND\_SIZE

```
public static final byte MASK_SOUND_SIZE = 2;
```

Mask sound size

## 12.24. MASK\_SOUND\_TYPE

```
public static final byte MASK_SOUND_TYPE = 1;
```

Mask sound type

## 12.25. MASK\_VIDEO\_CODEC

```
public static final byte MASK_VIDEO_CODEC = 15;
```

Mask video codec

## 12.26. MASK\_VIDEO\_FRAMETYPE

```
public static final byte MASK_VIDEO_FRAMETYPE = -15;
```

Video frametype flag

## 12.27. TYPE\_AUDIO

```
public static final byte TYPE_AUDIO = 8;
```

Audio data

## 12.28. TYPE\_METADATA

```
public static final byte TYPE_METADATA = 18;
```

Metadata

## 12.29. TYPE\_VIDEO

```
public static final byte TYPE_VIDEO = 9;
```

Video data

# 13. Class StreamableFileFactory

Creates streamable file services

## 13.1. Synopsis

```
public class StreamableFileFactory implements org.?red5.?io.?IStreamableFileFactory {
    // Public Static Fields

    public static org.slf4j.Logger logger;

    // Public Constructors

    public StreamableFileFactory();

    // Public Methods

    public IStreamableFileService getService(java.io.File fp);

    public java.util.Set<org.red5.io.IStreamableFileService> getServices();

    public void setServices(java.util.Set<org.red5.io.IStreamableFileService> services);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 13.2. getService(File)

```
public IStreamableFileService getService(java.io.File fp);
```

**Specified by:** Method getService in interface IStreamableFileFactory

## 13.3. getServices()

```
public java.util.Set<org.red5.io.IStreamableFileService> getServices();
```

**Specified by:** Method getServices in interface IStreamableFileFactory

Getter for services

## 13.4. setServices(Set<IStreamableFileService>)

```
public void setServices(java.util.Set<org.red5.io.IStreamableFileService> services);
```

Setter for services

Parameters	
services	Set of streamable file services

# 1. Class AMF

These are the core AMF data types supported by Red5. For detailed specification please see the link below.

## 1.1. Synopsis

```
public class AMF {  
    // Public Static Fields  
  
    public static final java.nio.charset.Charset CHARSET ;  
  
    public static final int LONG_STRING_LENGTH = 65535;  
  
    public static final byte TYPE_AMF3_OBJECT = 17;  
  
    public static final byte TYPE_ARRAY = 10;  
  
    public static final byte TYPE_BOOLEAN = 1;  
  
    public static final byte TYPE_CLASS_OBJECT = 16;  
  
    public static final byte TYPE_DATE = 11;  
  
    public static final byte TYPE_END_OF_OBJECT = 9;  
  
    public static final byte TYPE_LONG_STRING = 12;  
  
    public static final byte TYPE_MIXED_ARRAY = 8;  
  
    public static final byte TYPE_MOVIECLIP = 4;  
  
    public static final byte TYPE_NULL = 5;  
  
    public static final byte TYPE_NUMBER = 0;  
  
    public static final byte TYPE_OBJECT = 3;  
  
    public static final byte TYPE_RECORDSET = 14;  
  
    public static final byte TYPE_REFERENCE = 7;  
  
    public static final byte TYPE_STRING = 2;  
  
    public static final byte TYPE_UNDEFINED = 6;  
  
    public static final byte TYPE_UNSUPPORTED = 13;  
  
    public static final byte TYPE_XML = 15;  
  
    public static final byte VALUE_FALSE = 0;  
  
    public static final byte VALUE_TRUE = 1;
```

```
// Public Constructors

    public AMF();
}

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

#### See Also

AMF specification (external) [<http://osflash.org/amf/astypes>]

## 1.2. CHARSET

```
public static final java.nio.charset.Charset CHARSET ;
```

UTF-8 is used

## 1.3. LONG\_STRING\_LENGTH

```
public static final int LONG_STRING_LENGTH = 65535;
```

Max string lenght constant

## 1.4. TYPE\_AMF3\_OBJECT

```
public static final byte TYPE_AMF3_OBJECT = 17;
```

Object marker constant (for AMF3)

## 1.5. TYPE\_ARRAY

```
public static final byte TYPE_ARRAY = 10;
```

Array marker constant

## 1.6. TYPE\_BOOLEAN

```
public static final byte TYPE_BOOLEAN = 1;
```

Boolean value marker constant

## 1.7. TYPE\_CLASS\_OBJECT

```
public static final byte TYPE_CLASS_OBJECT = 16;
```

Class marker constant

## 1.8. TYPE\_DATE

```
public static final byte TYPE_DATE = 11;
```

Date marker constant

## 1.9. TYPE\_END\_OF\_OBJECT

```
public static final byte TYPE_END_OF_OBJECT = 9;
```

End of object marker constant

## 1.10. TYPE\_LONG\_STRING

```
public static final byte TYPE_LONG_STRING = 12;
```

Long string marker constant

## 1.11. TYPE\_MIXED\_ARRAY

```
public static final byte TYPE_MIXED_ARRAY = 8;
```

Mixed array marker constant

## 1.12. TYPE\_MOVIECLIP

```
public static final byte TYPE_MOVIECLIP = 4;
```

MovieClip marker constant

## 1.13. TYPE\_NULL

```
public static final byte TYPE_NULL = 5;
```

Null marker constant

## 1.14. TYPE\_NUMBER

```
public static final byte TYPE_NUMBER = 0;
```

Number marker constant

## 1.15. TYPE\_OBJECT

```
public static final byte TYPE_OBJECT = 3;
```

Object marker constant

## 1.16. TYPE\_RECORDSET

```
public static final byte TYPE_RECORDSET = 14;
```

Recordset marker constant

## 1.17. TYPE\_REFERENCE

```
public static final byte TYPE_REFERENCE = 7;
```

Object reference marker constant

## 1.18. TYPE\_STRING

```
public static final byte TYPE_STRING = 2;
```

String marker constant

## 1.19. TYPE\_UNDEFINED

```
public static final byte TYPE_UNDEFINED = 6;
```

Undefined marker constant

## 1.20. TYPE\_UNSUPPORTED

```
public static final byte TYPE_UNSUPPORTED = 13;
```

Unsupported type marker constant

## 1.21. TYPE\_XML

```
public static final byte TYPE_XML = 15;
```

XML marker constant

## 1.22. VALUE\_FALSE

```
public static final byte VALUE_FALSE = 0;
```

false marker constant

## 1.23. VALUE\_TRUE

```
public static final byte VALUE_TRUE = 1;
```

true marker constant

## 2. Class Input

Input for Red5 data types

### 2.1. Synopsis

```
public class Input extends org.red5.io.object.BaseInput
    implements org.red5.io.object.Input {
// Protected Fields

    protected org.apache.mina.common.ByteBuffer buf;

    protected byte currentDataType;

    protected static org.slf4j.Logger log;

// Public Constructors

    public Input(org.apache.mina.common.ByteBuffer buf);
```

## Package org.red5.io.amf

```
// Public Static Methods

    public static String getString(org.apache.mina.common.ByteBuffer buf);

// Public Methods

    public String getString();

    public boolean hasMoreProperties();

    public Object readArray(org.red5.io.object.Deserializer deserializer);

    public Boolean readBoolean();

    public org.red5.io.amf.org.red5.io.amf3.ByteArray readByteArray();

    public Object readCustom();

    public byte readDataType();

    public java.util.Date readDate();

    public java.util.Map<java.lang.String, java.lang.Object> readKeyValues(org.red5.io.object.Deseriali

    public Object readMap(org.red5.io.object.Deserializer deserializer);

    public Object readNull();

    public Number readNumber();

    public Object readObject(org.red5.io.object.Deserializer deserializer);

    public String readPropertyName();

    public Object readReference();

    public String readString();

    public org.w3c.dom.Document readXML();

    public void reset();

    public void reset(org.red5.io.object.BaseInput.ReferenceMode mode);

    public void skipEndObject();

    public void skipPropertySeparator();

// Protected Methods

    protected Class getPropertyType(Object instance,
                                    String propertyName);

    protected Object newInstance(String className);

    protected Object readBean(org.red5.io.object.Deserializer deserializer,
                           Object bean);
```

```

protected byte readDataType(byte dataType);

protected void readKeyValues(java.util.Map<java.lang.String, java.lang.Object> result,
                            org.red5.io.object.Deserializer deserializer);

protected java.util.Map<java.lang.String, java.lang.Object> readSimpleObject(org.red5.io.object.De
}

}

```

**Direct known subclasses:** org.?red5.?io.?amf3.?Input

**Methods inherited from org.red5.io.object.BaseInput:** clearReferences , getReference , storeReference

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.io.object.BaseInput:** referenceMode , refId , refMap

## 2.2. Input(ByteBuffer)

```
public Input(org.apache.mina.common.ByteBuffer buf);
```

Creates Input object from byte buffer

### Parameters

buf	Byte buffer
-----	-------------

## 2.3. getString()

```
public String getString();
```

**Specified by:** Method getString in interface Input

Reads string from buffer

### Parameters

return	String
--------	--------

## 2.4. getString(ByteBuffer)

```
public static String getString(org.apache.mina.common.ByteBuffer buf);
```

Returns a string based on the buffer

### Parameters

buf	Byte buffer with data
-----	-----------------------

<i>return</i>	String Decoded string
---------------	-----------------------

## 2.5. hasMoreProperties()

```
public boolean hasMoreProperties();
```

Returns a boolean stating whether there are more properties

### Parameters

<i>return</i>	boolean <code>true</code> if there are more properties to read, <code>false</code> otherwise
---------------	--

## 2.6. newInstance(String)

```
protected Object newInstance(String className);
```

Creates a new instance of the className parameter and returns as an Object

### Parameters

className	Class name as String
-----------	----------------------

<i>return</i>	Object New object instance (for given class)
---------------	--

## 2.7. readArray(Deserializer)

```
public Object readArray(org.red5.io.object.Deserializer deserializer);
```

**Specified by:** Method `readArray` in interface `Input`

Read an array. This can result in a List or Map being deserialized depending on the array type found.

### Parameters

<i>return</i>	array
---------------	-------

**Description copied from interface: `readArray`**

## 2.8. readBean(Deserializer, Object)

```
protected Object readBean(org.red5.io.object.Deserializer deserializer,
                        Object bean);
```

Reads the input as a bean and returns an object

### Parameters

deserializer	Deserializer used
--------------	-------------------

bean	Input as bean
------	---------------

<i>return</i>	Decoded object
---------------	----------------

## 2.9. readBoolean()

```
public Boolean readBoolean();
```

**Specified by:** Method readBoolean in interface Input

Reads a boolean.

### Parameters

<i>return</i>	boolean
---------------	---------

## 2.10. readByteArray()

```
public org.red5.io.amf.org.red5.io.amf3.ByteArray readByteArray();
```

**Specified by:** Method readByteArray in interface Input

Read ByteArray object. This is not supported by the AMF0 deserializer.

### Parameters

<i>return</i>	ByteArray object
---------------	------------------

## 2.11. readCustom()

```
public Object readCustom();
```

**Specified by:** Method readCustom in interface Input

Reads Custom

### Parameters

<i>return</i>	Object Custom type object
---------------	---------------------------

## 2.12. readDataType()

```
public byte readDataType();
```

**Specified by:** Method readDataType in interface Input

Reads the data type.

### Parameters

<i>return</i>	byte Data type
---------------	----------------

## 2.13. readDataType(byte)

```
protected byte readDataType(byte dataType);
```

Reads the data type.

**Parameters**

<i>dataType</i>	Data type as byte
<i>return</i>	One of AMF class constants with type

**See Also**

org.red5.io.amf.AMF}

**2.14. readDate()**

```
public java.util.Date readDate();
```

**Specified by:** Method readDate in interface Input

Returns a date

**Parameters**

<i>return</i>	Date Decoded string object
---------------	----------------------------

**2.15. readKeyValues(Deserializer)**

```
public java.util.Map<java.lang.String, java.lang.Object> readKeyValues(org.red5.io.object.Deserializer);
```

**Specified by:** Method readKeyValues in interface Input

Read key - value pairs. This is required for the RecordSet deserializer.

**2.16. readKeyValues(Map<String, Object>, Deserializer)**

```
protected void readKeyValues(java.util.Map<java.lang.String, java.lang.Object> result,  
                           org.red5.io.object.Deserializer deserializer);
```

Read key - value pairs into Map object

**Parameters**

<i>result</i>	Map to put resulting pair to
<i>deserializer</i>	Deserializer used

**2.17. readMap(Deserializer)**

```
public Object readMap(org.red5.io.object.Deserializer deserializer);
```

**Specified by:** Method readMap in interface Input

Read a map containing key - value pairs. This can result in a List or Map being deserialized depending on the map type found.

**Parameters**

<i>return</i>	Map
---------------	-----

**Description copied from interface: readMap**

## 2.18. readNull()

<code>public Object readNull();</code>
--

**Specified by:** Method readNull in interface Input

Reads a null.

**Parameters**

<i>return</i>	Object
---------------	--------

## 2.19. readNumber()

<code>public Number readNumber();</code>
--

**Specified by:** Method readNumber in interface Input

Reads a Number. In ActionScript 1 and 2 Number type represents all numbers, both floats and integers.

**Parameters**

<i>return</i>	Number
---------------	--------

## 2.20. readObject(Deserializer)

<code>public Object readObject(org.red5.io.object.Deserializer deserializer);</code>
--

**Specified by:** Method readObject in interface Input

Reads start object

**Parameters**

deserializer	Deserializer to use
--------------	---------------------

<i>return</i>	Read object
---------------	-------------

## 2.21. readPropertyName()

<code>public String readPropertyName();</code>
--

Reads property name

**Parameters**

<i>return</i>	String Object property name
---------------	-----------------------------

## 2.22. readReference()

<code>public Object readReference();</code>
---

**Specified by:** Method readReference in interface Input

Reads Reference

## Parameters

<i>return</i>	Object Read reference to object
---------------	---------------------------------

**2.23. readSimpleObject(Deserializer)**

```
protected java.util.Map<java.lang.String, java.lang.Object> readSimpleObject(org.red5.io.object.Deser...
```

Reads the input as a map and returns a Map

## Parameters

deserializer	Deserializer to use
<i>return</i>	Read map

**2.24. readString()**

```
public String readString();
```

**Specified by:** Method readString in interface Input

Reads a string

## Parameters

<i>return</i>	String
---------------	--------

**2.25. readXML()**

```
public org.w3c.dom.Document readXML();
```

**Specified by:** Method readXML in interface Input

Reads XML

## Parameters

<i>return</i>	String XML as string
---------------	----------------------

**2.26. reset()**

```
public void reset();
```

Resets map

**2.27. reset(BaseInput.ReferenceMode)**

```
public void reset(org.red5.io.object.BaseInput.ReferenceMode mode);
```

Resets map and set mode to handle references

#### Parameters

mode	mode to handle references
------	---------------------------

## 2.28. skipEndObject()

```
public void skipEndObject();
```

Skips end object

## 2.29. skipPropertySeparator()

```
public void skipPropertySeparator();
```

Skips property seperator

## 3. Class Output

```
public class Output extends, org.?red5.?io.?object.?BaseOutput
    implements, org.?red5.?io.?object.?Output {
// Protected Fields

    protected org.apache.mina.common.ByteBuffer buf ;

    protected static org.slf4j.Logger log ;

    protected static final java.util.Map<java.lang.String, byte[]> stringCache ;

// Public Constructors

    public Output(org.apache.mina.common.ByteBuffer buf);

// Public Static Methods

    public static void putString(org.apache.mina.common.ByteBuffer buf,
                               String string);

// Public Methods

    public org.apache.mina.common.ByteBuffer buf();

    public boolean isCustom(Object custom);

    public void putString(String string);

    public void reset();

    public boolean supportsDataType(byte type);

    public void writeArray(Object array,
                          org.red5.io.object.Serializer serializer);

    public void writeArray(Object[] array,
                          org.red5.io.object.Serializer serializer);
```

## Package org.?red5.?io.?amf

```
public void writeArray(java.util.Collection<?> array,
                      org.red5.io.object.Serializer serializer);

public void writeBoolean(Boolean bol);

public void writeByteArray(org.red5.io.amf.org.red5.io.amf3.ByteArray array);

public void writeCustom(Object custom);

public void writeDate(java.util.Date date);

public void writeMap(java.util.Collection<?> array,
                     org.red5.io.object.Serializer serializer);

public void writeMap(java.util.Map<java.lang.Object, java.lang.Object> map,
                     org.red5.io.object.Serializer serializer);

public void writeNull();

public void writeNumber(Number num);

public void writeObject(Object object,
                       org.red5.io.object.Serializer serializer);

public void writeObject(java.util.Map<java.lang.Object, java.lang.Object> map,
                       org.red5.io.object.Serializer serializer);

public void writeRecordSet(org.red5.io.object.RecordSet recordset,
                          org.red5.io.object.Serializer serializer);

public void writeReference(Object obj);

public void writeString(String string);

public void writeXML(org.w3c.dom.Document xml);

// Protected Methods

protected boolean checkWriteReference(Object obj);

protected boolean dontSerializeField(Class<?> objectClass,
                                     String keyName);

protected static byte[] encodeString(String string);

protected void writeArbitraryObject(Object object,
                                    org.red5.io.object.Serializer serializer);

}
```

**Direct known subclasses:** org.?red5.?io.?amf3.?Output

**Methods inherited from org.red5.io.object.BaseOutput:** clearReferences ,  
getReferenceId , hasReference , storeReference

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.io.object.BaseOutput:** refId , refMap

### 3.1. Output(ByteBuffer)

```
public Output(org.apache.mina.common.ByteBuffer buf);
```

Creates output with given byte buffer

#### Parameters

buf	Byte buffer
-----	-------------

### 3.2. buf

```
protected org.apache.mina.common.ByteBuffer buf ;
```

Output buffer

### 3.3. stringCache

```
protected static final java.util.Map<java.lang.String, byte[]> stringCache ;
```

Cache encoded strings.

### 3.4. buf()

```
public org.apache.mina.common.ByteBuffer buf();
```

Return buffer of this Output object

#### Parameters

return	Byte buffer of this Output object
--------	-----------------------------------

### 3.5. encodeString(String)

```
protected static byte[] encodeString(String string);
```

Encode string.

#### Parameters

string	
return	encoded string

### 3.6. isCustom(Object)

```
public boolean isCustom(Object custom);
```

**Specified by:** Method isCustom in interface Output

Whether object is custom

### 3.7. putString(ByteBuffer, String)

```
public static void putString(org.apache.mina.common.ByteBuffer buf,
                           String string);
```

Write out string

#### Parameters

buf	Byte buffer to write to
string	String to write

### 3.8. putString(String)

```
public void putString(String string);
```

**Specified by:** Method putString in interface Output

### 3.9. supportsDataType(byte)

```
public boolean supportsDataType(byte type);
```

**Specified by:** Method supportsDataType in interface Output

### 3.10. writeArbitraryObject(Object, Serializer)

```
protected void writeArbitraryObject(Object object,
                                    org.red5.io.object.Serializer serializer);
```

Writes an arbitrary object to the output.

#### Parameters

serializer	Output writer
object	Object to write

### 3.11. writeArray(Collection<?>, Serializer)

```
public void writeArray(java.util.Collection<?> array,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeArray in interface Output

Write array.

### 3.12. writeArray(Object[], Serializer)

```
public void writeArray(Object[] array,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeArray in interface Output

Write array.

### 3.13. writeArray(Object, Serializer)

```
public void writeArray(Object array,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeArray in interface Output

Write primitive array.

### 3.14. writeBoolean(Boolean)

```
public void writeBoolean(Boolean bol);
```

**Specified by:** Method writeBoolean in interface Output

Write boolean

### 3.15. writeByteArray(ByteArray)

```
public void writeByteArray(org.red5.io.amf.org.red5.io.amf3.ByteArray array);
```

**Specified by:** Method writeByteArray in interface Output

Write ByteArray object (AMF3 only).

### 3.16. writeCustom(Object)

```
public void writeCustom(Object custom);
```

**Specified by:** Method writeCustom in interface Output

Write custom (user) object

### 3.17. writeDate(Date)

```
public void writeDate(java.util.Date date);
```

**Specified by:** Method writeDate in interface Output

Write date

### 3.18. writeMap(Collection<?>, Serializer)

```
public void writeMap(java.util.Collection<?> array,
                     org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeMap in interface Output

Write array as map.

### 3.19. writeMap(Map<Object, Object>, Serializer)

```
public void writeMap(java.util.Map<java.lang.Object, java.lang.Object> map,
                     org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeMap in interface Output

Write map.

### 3.20. writeNull()

```
public void writeNull();
```

**Specified by:** Method writeNull in interface Output

### 3.21. writeNumber(Number)

```
public void writeNumber(Number num);
```

**Specified by:** Method writeNumber in interface Output

Write number

### 3.22. writeObject(Map<Object, Object>, Serializer)

```
public void writeObject(java.util.Map<java.lang.Object, java.lang.Object> map,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeObject in interface Output

Write map as object.

### 3.23. writeObject(Object, Serializer)

```
public void writeObject(Object object,
                       org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeObject in interface Output

Write object.

### 3.24. writeRecordSet(RecordSet, Serializer)

```
public void writeRecordSet(org.red5.io.object.RecordSet recordset,
                         org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeRecordSet in interface Output

Write recordset.

### 3.25. writeReference(Object)

```
public void writeReference(Object obj);
```

**Specified by:** Method writeReference in interface Output

Write reference to complex data type

### 3.26. writeString(String)

```
public void writeString(String string);
```

**Specified by:** Method writeString in interface Output

Write string

### 3.27. writeXML(Document)

```
public void writeXML(org.w3c.dom.Document xml);
```

**Specified by:** Method writeXML in interface Output

Write XML object

# 1. Class AMF3

AMF3 data type definitions. For detailed specification please see the link below.

## 1.1. Synopsis

```
public class AMF3 {  
    // Public Static Fields  
  
    public static final java.nio.charset.Charset CHARSET ;  
  
    public static final int LONG_STRING_LENGTH = 65535;  
  
    public static final long MAX_INTEGER_VALUE = 268435455L;  
  
    public static final long MIN_INTEGER_VALUE = -268435456L;  
  
    public static final byte TYPE_ARRAY = 9;  
  
    public static final byte TYPE_BOOLEAN_FALSE = 2;  
  
    public static final byte TYPE_BOOLEAN_TRUE = 3;  
  
    public static final byte TYPE_BYTEARRAY = 12;  
  
    public static final byte TYPE_DATE = 8;  
  
    public static final byte TYPE_INTEGER = 4;  
  
    public static final byte TYPE_NULL = 1;  
  
    public static final byte TYPE_NUMBER = 5;  
  
    public static final byte TYPE_OBJECT = 10;  
  
    public static final byte TYPE_OBJECT_EXTERNALIZABLE = 1;  
  
    public static final byte TYPE_OBJECT_PROPERTY = 0;  
  
    public static final byte TYPE_OBJECT_PROXY = 3;  
  
    public static final byte TYPE_OBJECT_VALUE = 2;  
  
    public static final byte TYPE_STRING = 6;  
  
    public static final byte TYPE_UNDEFINED = 0;  
  
    public static final byte TYPE_XML = 11;  
  
    public static final byte TYPE_XML_DOCUMENT = 7;  
  
    // Public Constructors  
  
    public AMF3();
```

```
}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

*See Also*

AMF3 specification (external) [<http://osflash.org/amf3/index>] , Official Adobe AMF3 Specification [[http://download.macromedia.com/pub/labs/amf/amf3\\_spec\\_121207.pdf](http://download.macromedia.com/pub/labs/amf/amf3_spec_121207.pdf)]

## 1.2. CHARSET

```
public static final java.nio.charset.Charset CHARSET ;
```

UTF-8 is used

## 1.3. LONG\_STRING\_LENGTH

```
public static final int LONG_STRING_LENGTH = 65535;
```

Max string length

## 1.4. MAX\_INTEGER\_VALUE

```
public static final long MAX_INTEGER_VALUE = 268435455L;
```

Maximum possible value for integer number encoding.

## 1.5. MIN\_INTEGER\_VALUE

```
public static final long MIN_INTEGER_VALUE = -268435456L;
```

Minimum possible value for integer number encoding.

## 1.6. TYPE\_ARRAY

```
public static final byte TYPE_ARRAY = 9;
```

Array start marker

## 1.7. TYPE\_BOOLEAN\_FALSE

```
public static final byte TYPE_BOOLEAN_FALSE = 2;
```

Boolean false marker

## 1.8. TYPE\_BOOLEAN\_TRUE

```
public static final byte TYPE_BOOLEAN_TRUE = 3;
```

Boolean true marker

## 1.9. TYPE\_BYTEARRAY

```
public static final byte TYPE_BYTEARRAY = 12;
```

ByteArray marker

## 1.10. TYPE\_DATE

```
public static final byte TYPE_DATE = 8;
```

Date marker

## 1.11. TYPE\_INTEGER

```
public static final byte TYPE_INTEGER = 4;
```

Integer marker

## 1.12. TYPE\_NULL

```
public static final byte TYPE_NULL = 1;
```

Null marker

## 1.13. TYPE\_NUMBER

```
public static final byte TYPE_NUMBER = 5;
```

Number marker

## 1.14. TYPE\_OBJECT

```
public static final byte TYPE_OBJECT = 10;
```

Object start marker

## 1.15. TYPE\_OBJECT\_EXTERNALIZABLE

```
public static final byte TYPE_OBJECT_EXTERNALIZABLE = 1;
```

Externalizable object. What follows is the value of the "inner" object, including type code. This value appears for objects that implement IExternalizable, such as ArrayCollection and ObjectProxy.

## 1.16. TYPE\_OBJECT\_PROPERTY

```
public static final byte TYPE_OBJECT_PROPERTY = 0;
```

Property list encoding. The remaining integer-data represents the number of class members that exist. The property names are read as string-data. The values are then read as AMF3-data.

## 1.17. TYPE\_OBJECT\_PROXY

```
public static final byte TYPE_OBJECT_PROXY = 3;
```

Proxy object.

## 1.18. TYPE\_OBJECT\_VALUE

```
public static final byte TYPE_OBJECT_VALUE = 2;
```

Name-value encoding. The property names and values are encoded as string-data followed by AMF3-data until there is an empty string property name. If there is a class-def reference there are no property names and the number of values is equal to the number of properties in the class-def.

## 1.19. TYPE\_STRING

```
public static final byte TYPE_STRING = 6;
```

String marker

## 1.20. TYPE\_UNDEFINED

```
public static final byte TYPE_UNDEFINED = 0;
```

Undefined marker

## 1.21. TYPE\_XML

```
public static final byte TYPE_XML = 11;
```

XML start marker

## 1.22. TYPE\_XML\_DOCUMENT

```
public static final byte TYPE_XML_DOCUMENT = 7;
```

XML document marker

This is for the legacy XMLDocument type is retained in the language as flash.xml.XMLDocument. Similar to AMF 0, the structure of an XMLDocument needs to be flattened into a string representation for serialization. As with other strings in AMF, the content is encoded in UTF-8. XMLDocuments can be sent as a reference to a previously occurring XMLDocument instance by using an index to the implicit object reference table.

## 2. Class ByteArray

Red5 version of the Flex ByteArray class.

### 2.1. Synopsis

```
public class ByteArray implements, org.?red5.?io.?amf3.?IDataInput, org.?red5.?io.?amf3.?IDataOutput {
```

```
// Protected Fields

protected org.apache.mina.common.ByteBuffer data ;

protected IDataInput dataInput ;

protected IDataOutput dataOutput ;

// Public Constructors

public ByteArray();

// Protected Constructors

protected ByteArray(org.apache.mina.common.ByteBuffer buffer,
                   int length);

// Public Methods

public int bytesAvailable();

public void compress();

public java.nio.ByteOrder getEndian();

public int length();

public int position();

public void position(int position);

public boolean readBoolean();

public byte readByte();

public void readBytes(byte[] bytes);

public void readBytes(byte[] bytes,
                     int offset);

public void readBytes(byte[] bytes,
                     int offset,
                     int length);

public double readDouble();

public float readFloat();

public int readInt();

public String readMultiByte(int length,
                           String charSet);

public Object readObject();

public short readShort();

public String readUTF();
```

## Package org.?red5.?io.?amf3

```
public String readUTFBytes(int length);

public int readUnsignedByte();

public long readUnsignedInt();

public int readUnsignedShort();

public void setEndian(java.nio.ByteOrder endian);

public String toString();

public void uncompress();

public void writeBoolean(boolean value);

public void writeByte(byte value);

public void writeBytes(byte[] bytes);

public void writeBytes(byte[] bytes,
                      int offset);

public void writeBytes(byte[] bytes,
                      int offset,
                      int length);

public void writeDouble(double value);

public void writeFloat(float value);

public void writeInt(int value);

public void writeMultiByte(String value,
                           String encoding);

public void writeObject(Object value);

public void writeShort(short value);

public void writeUTF(String value);

public void writeUTFBytes(String value);

public void writeUnsignedInt(long value);

// Protected Methods

protected org.apache.mina.common.ByteBuffer getData();

protected void prepareIO();

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 2.2. ByteArray()

```
public ByteArray();
```

Public constructor. Creates new empty ByteArray.

## 2.3. ByteArray(ByteBuffer, int)

```
protected ByteArray(org.apache.mina.common.ByteBuffer buffer,
                  int length);
```

Internal constructor used to create ByteArray during deserialization.

### Parameters

buffer	
--------	--

length	
--------	--

## 2.4. data

```
protected org.apache.mina.common.ByteBuffer data ;
```

Internal storage for array contents.

## 2.5. dataInput

```
protected IDataInput dataInput ;
```

Object used to read from array.

## 2.6. dataOutput

```
protected IDataOutput dataOutput ;
```

Object used to write to array.

## 2.7. bytesAvailable()

```
public int bytesAvailable();
```

Return number of bytes available for reading.

### Parameters

<i>return</i>	
---------------	--

## 2.8. compress()

```
public void compress();
```

Compress contents using zlib.

## 2.9. getData()

```
protected org.apache.mina.common.ByteBuffer getData();
```

Get internal data.

### Parameters

<i>return</i>
---------------

## 2.10. getEndian()

```
public java.nio.ByteOrder getEndian();
```

**Specified by:** Method `getEndian` in interface `IDataInput`

Return the byteorder used when loading values.

## 2.11. length()

```
public int length();
```

Return total number of bytes in array.

### Parameters

<i>return</i>
---------------

## 2.12. position()

```
public int position();
```

Get the current position in the data.

### Parameters

<i>return</i>
---------------

## 2.13. position(int)

```
public void position(int position);
```

Set the current position in the data.

### Parameters

position
----------

## 2.14. prepareIO()

```
protected void prepareIO();
```

Create internal objects used for reading and writing.

## 2.15. readBoolean()

```
public boolean readBoolean();
```

**Specified by:** Method readBoolean in interface IDataInput

Read boolean value.

## 2.16. readByte()

```
public byte readByte();
```

**Specified by:** Method readByte in interface IDataInput

Read signed single byte value.

## 2.17. readBytes(byte[])

```
public void readBytes(byte[] bytes);
```

**Specified by:** Method readBytes in interface IDataInput

Read list of bytes.

## 2.18. readBytes(byte[], int)

```
public void readBytes(byte[] bytes,
                      int offset);
```

**Specified by:** Method readBytes in interface IDataInput

Read list of bytes to given offset.

## 2.19. readBytes(byte[], int, int)

```
public void readBytes(byte[] bytes,
                      int offset,
                      int length);
```

**Specified by:** Method readBytes in interface IDataInput

Read given number of bytes to given offset.

## 2.20. readDouble()

```
public double readDouble();
```

**Specified by:** Method readDouble in interface IDataInput

Read double-precision floating point value.

## 2.21. readFloat()

```
public float readFloat();
```

**Specified by:** Method readFloat in interface IDataInput

Read single-precision floating point value.

## 2.22. **readInt()**

```
public int readInt();
```

**Specified by:** Method readInt in interface IDataInput

Read signed integer value.

## 2.23. **readMultiByte(int, String)**

```
public String readMultiByte(int length,  
                           String charSet);
```

**Specified by:** Method readMultiByte in interface IDataInput

Read multibyte string.

## 2.24. **readObject()**

```
public Object readObject();
```

**Specified by:** Method readObject in interface IDataInput

Read arbitrary object.

## 2.25. **readShort()**

```
public short readShort();
```

**Specified by:** Method readShort in interface IDataInput

Read signed short value.

## 2.26. **readUnsignedByte()**

```
public int readUnsignedByte();
```

**Specified by:** Method readUnsignedByte in interface IDataInput

Read unsigned single byte value.

## 2.27. **readUnsignedInt()**

```
public long readUnsignedInt();
```

**Specified by:** Method readUnsignedInt in interface IDataInput

Read unsigned integer value.

## 2.28. **readUnsignedShort()**

```
public int readUnsignedShort();
```

**Specified by:** Method readUnsignedShort in interface IDataInput

Read unsigned short value.

## 2.29. readUTF()

```
public String readUTF();
```

**Specified by:** Method readUTF in interface IDataInput

Read UTF-8 encoded string.

## 2.30. readUTFBytes(int)

```
public String readUTFBytes(int length);
```

**Specified by:** Method readUTFBytes in interface IDataInput

Read UTF-8 encoded string with given length.

## 2.31. setEndian(ByteOrder)

```
public void setEndian(java.nio.ByteOrder endian);
```

**Specified by:** Method setEndian in interface IDataInput

Set the byteorder to use when loading values.

## 2.32. toString()

```
public String toString();
```

Return string representation of the array's contents.

### Parameters

return
--------

## 2.33. uncompress()

```
public void uncompress();
```

Decompress contents using zlib.

## 2.34. writeBoolean(boolean)

```
public void writeBoolean(boolean value);
```

**Specified by:** Method writeBoolean in interface IDataOutput

Write boolean value.

## 2.35. writeByte(byte)

```
public void writeByte(byte value);
```

**Specified by:** Method writeByte in interface IDataOutput

Write signed byte value.

## 2.36. writeBytes(byte[])

```
public void writeBytes(byte[] bytes);
```

**Specified by:** Method writeBytes in interface IDataOutput

Write multiple bytes.

## 2.37. writeBytes(byte[], int)

```
public void writeBytes(byte[] bytes,
                      int offset);
```

**Specified by:** Method writeBytes in interface IDataOutput

Write multiple bytes from given offset.

## 2.38. writeBytes(byte[], int, int)

```
public void writeBytes(byte[] bytes,
                      int offset,
                      int length);
```

**Specified by:** Method writeBytes in interface IDataOutput

Write given number of bytes from given offset.

## 2.39. writeDouble(double)

```
public void writeDouble(double value);
```

**Specified by:** Method writeDouble in interface IDataOutput

Write double-precision floating point value.

## 2.40. writeFloat(float)

```
public void writeFloat(float value);
```

**Specified by:** Method writeFloat in interface IDataOutput

Write single-precision floating point value.

## 2.41. writeInt(int)

```
public void writeInt(int value);
```

**Specified by:** Method writeInt in interface IDataOutput

Write signed integer value.

## 2.42. writeMultiByte(String, String)

```
public void writeMultiByte(String value,
```

```
String encoding);
```

**Specified by:** Method writeMultiByte in interface IDataOutput

Write string in given character set.

## 2.43. writeObject(Object)

```
public void writeObject(Object value);
```

**Specified by:** Method writeObject in interface IDataOutput

Write arbitrary object.

## 2.44. writeShort(short)

```
public void writeShort(short value);
```

**Specified by:** Method writeShort in interface IDataOutput

Write signed short value.

## 2.45. writeUnsignedInt(long)

```
public void writeUnsignedInt(long value);
```

**Specified by:** Method writeUnsignedInt in interface IDataOutput

Write unsigned integer value.

## 2.46. writeUTF(String)

```
public void writeUTF(String value);
```

**Specified by:** Method writeUTF in interface IDataOutput

Write UTF-8 encoded string.

## 2.47. writeUTFBytes(String)

```
public void writeUTFBytes(String value);
```

**Specified by:** Method writeUTFBytes in interface IDataOutput

Write UTF-8 encoded string as byte array. This string is stored without informations about its length, so `readUTFBytes(int)` must be used to load it.

# 3. Class DataInput

Implementation of the IDataInput interface. Can be used to load an IExternalizable object.

## 3.1. Synopsis

```
public class DataInput implements, org.?red5.?io.?amf3.?IDataInput {
    // Protected Constructors
```

```

protected DataInput(Input input,
                  org.red5.io.object.Deserializer deserializer);

// Public Methods

public java.nio.ByteOrder getEndian();

public boolean readBoolean();

public byte readByte();

public void readBytes(byte[] bytes);

public void readBytes(byte[] bytes,
                     int offset);

public void readBytes(byte[] bytes,
                     int offset,
                     int length);

public double readDouble();

public float readFloat();

public int readInt();

public String readMultiByte(int length,
                           String charSet);

public Object readObject();

public short readShort();

public String readUTF();

public String readUTFBytes(int length);

public int readUnsignedByte();

public long readUnsignedInt();

public int readUnsignedShort();

public void setEndian(java.nio.ByteOrder endian);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

## 3.2. DataInput(Input, Deserializer)

```

protected DataInput(Input input,
                  org.red5.io.object.Deserializer deserializer);

```

Create a new DataInput.

#### Parameters

input	input to use
deserializer	the deserializer to use

### 3.3. getEndian()

```
public java.nio.ByteOrder getEndian();
```

**Specified by:** Method getEndian in interface IDataInput

Return the byteorder used when loading values.

### 3.4. readBoolean()

```
public boolean readBoolean();
```

**Specified by:** Method readBoolean in interface IDataInput

Read boolean value.

### 3.5. readByte()

```
public byte readByte();
```

**Specified by:** Method readByte in interface IDataInput

Read signed single byte value.

### 3.6. readBytes(byte[])

```
public void readBytes(byte[] bytes);
```

**Specified by:** Method readBytes in interface IDataInput

Read list of bytes.

### 3.7. readBytes(byte[], int)

```
public void readBytes(byte[] bytes,
                      int offset);
```

**Specified by:** Method readBytes in interface IDataInput

Read list of bytes to given offset.

### 3.8. readBytes(byte[], int, int)

```
public void readBytes(byte[] bytes,
                      int offset,
                      int length);
```

**Specified by:** Method readBytes in interface IDataInput

Read given number of bytes to given offset.

### 3.9. **readDouble()**

```
public double readDouble();
```

**Specified by:** Method readDouble in interface IDataInput

Read double-precision floating point value.

### 3.10. **readFloat()**

```
public float readFloat();
```

**Specified by:** Method readFloat in interface IDataInput

Read single-precision floating point value.

### 3.11. **readInt()**

```
public int readInt();
```

**Specified by:** Method readInt in interface IDataInput

Read signed integer value.

### 3.12. **readMultiByte(int, String)**

```
public String readMultiByte(int length,  
                           String charSet);
```

**Specified by:** Method readMultiByte in interface IDataInput

Read multibyte string.

### 3.13. **readObject()**

```
public Object readObject();
```

**Specified by:** Method readObject in interface IDataInput

Read arbitrary object.

### 3.14. **readShort()**

```
public short readShort();
```

**Specified by:** Method readShort in interface IDataInput

Read signed short value.

### 3.15. **readUnsignedByte()**

```
public int readUnsignedByte();
```

**Specified by:** Method readUnsignedByte in interface IDataInput

Read unsigned single byte value.

### 3.16. **readUnsignedInt()**

```
public long readUnsignedInt();
```

**Specified by:** Method readUnsignedInt in interface IDataInput

Read unsigned integer value.

### 3.17. **readUnsignedShort()**

```
public int readUnsignedShort();
```

**Specified by:** Method readUnsignedShort in interface IDataInput

Read unsigned short value.

### 3.18. **readUTF()**

```
public String readUTF();
```

**Specified by:** Method readUTF in interface IDataInput

Read UTF-8 encoded string.

### 3.19. **readUTFBytes(int)**

```
public String readUTFBytes(int length);
```

**Specified by:** Method readUTFBytes in interface IDataInput

Read UTF-8 encoded string with given length.

### 3.20. **setEndian(ByteOrder)**

```
public void setEndian(java.nio.ByteOrder endian);
```

**Specified by:** Method setEndian in interface IDataInput

Set the byteorder to use when loading values.

## 4. Class DataOutput

Implementation of the IDataOutput interface. Can be used to store an IExternalizable object.

### 4.1. Synopsis

```
public class DataOutput implements, org.?red5.?io.?amf3.?IDataOutput {
    // Protected Constructors

    protected DataOutput(Output output,
                        org.red5.io.object.Serializer serializer);
```

```
// Public Methods

    public java.nio.ByteOrder getEndian();

    public void setEndian(java.nio.ByteOrder endian);

    public void writeBoolean(boolean value);

    public void writeByte(byte value);

    public void writeBytes(byte[] bytes);

    public void writeBytes(byte[] bytes,
                          int offset);

    public void writeBytes(byte[] bytes,
                          int offset,
                          int length);

    public void writeDouble(double value);

    public void writeFloat(float value);

    public void writeInt(int value);

    public void writeMultiByte(String value,
                             String encoding);

    public void writeObject(Object value);

    public void writeShort(short value);

    public void writeUTF(String value);

    public void writeUTFBytes(String value);

    public void writeUnsignedInt(long value);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 4.2. DataOutput(Output, Serializer)

```
protected DataOutput(Output output,
                    org.red5.io.object.Serializer serializer);
```

Create a new DataOutput.

### Parameters

output	destination to write to
serializer	the serializer to use

## 4.3. getEndian()

```
public java.nio.ByteOrder getEndian();
```

**Specified by:** Method getEndian in interface IDataOutput

Return the byteorder used when storing values.

## 4.4. setEndian(ByteOrder)

```
public void setEndian(java.nio.ByteOrder endian);
```

**Specified by:** Method setEndian in interface IDataOutput

Set the byteorder to use when storing values.

## 4.5. writeBoolean(boolean)

```
public void writeBoolean(boolean value);
```

**Specified by:** Method writeBoolean in interface IDataOutput

Write boolean value.

## 4.6. writeByte(byte)

```
public void writeByte(byte value);
```

**Specified by:** Method writeByte in interface IDataOutput

Write signed byte value.

## 4.7. writeBytes(byte[])

```
public void writeBytes(byte[] bytes);
```

**Specified by:** Method writeBytes in interface IDataOutput

Write multiple bytes.

## 4.8. writeBytes(byte[], int)

```
public void writeBytes(byte[] bytes,
                      int offset);
```

**Specified by:** Method writeBytes in interface IDataOutput

Write multiple bytes from given offset.

## 4.9. writeBytes(byte[], int, int)

```
public void writeBytes(byte[] bytes,
                      int offset,
                      int length);
```

**Specified by:** Method writeBytes in interface IDataOutput

Write given number of bytes from given offset.

#### 4.10. writeDouble(double)

```
public void writeDouble(double value);
```

**Specified by:** Method writeDouble in interface IDataOutput

Write double-precision floating point value.

#### 4.11. writeFloat(float)

```
public void writeFloat(float value);
```

**Specified by:** Method writeFloat in interface IDataOutput

Write single-precision floating point value.

#### 4.12. writeInt(int)

```
public void writeInt(int value);
```

**Specified by:** Method writeInt in interface IDataOutput

Write signed integer value.

#### 4.13. writeMultiByte(String, String)

```
public void writeMultiByte(String value,
                           String encoding);
```

**Specified by:** Method writeMultiByte in interface IDataOutput

Write string in given character set.

#### 4.14. writeObject(Object)

```
public void writeObject(Object value);
```

**Specified by:** Method writeObject in interface IDataOutput

Write arbitrary object.

#### 4.15. writeShort(short)

```
public void writeShort(short value);
```

**Specified by:** Method writeShort in interface IDataOutput

Write signed short value.

#### 4.16. writeUnsignedInt(long)

```
public void writeUnsignedInt(long value);
```

**Specified by:** Method writeUnsignedInt in interface IDataOutput

Write unsigned integer value.

## 4.17. writeUTF(String)

```
public void writeUTF(String value);
```

**Specified by:** Method writeUTF in interface IDataOutput

Write UTF-8 encoded string.

## 4.18. writeUTFBytes(String)

```
public void writeUTFBytes(String value);
```

**Specified by:** Method writeUTFBytes in interface IDataOutput

Write UTF-8 encoded string as byte array. This string is stored without informations about its length, so `readUTFBytes(int)` must be used to load it.

# 5. Interface IDataInput

Interface implemented by classes that provide a way to load custom objects.

## 5.1. Synopsis

```
public interface IDataInput {
    // Public Methods

    public java.nio.ByteOrder getEndian();

    public boolean readBoolean();

    public byte readByte();

    public void readBytes(byte[] bytes);

    public void readBytes(byte[] bytes,
                         int offset);

    public void readBytes(byte[] bytes,
                         int offset,
                         int length);

    public double readDouble();

    public float readFloat();

    public int readInt();

    public String readMultiByte(int length,
                               String charSet);

    public Object readObject();

    public short readShort();
```

```

public String readUTF();

public String readUTFBytes(int length);

public int readUnsignedByte();

public long readUnsignedInt();

public int readUnsignedShort();

public void setEndian(java.nio.ByteOrder endian);

}

```

**See Also**

`readExternal(org.red5.io.amf3.IDataInput)` , [Adobe Livedocs \(external\) \[http://livedocs.adobe.com/flex/2/langref/flash/utils/IDataInput.html\]](#)

**5.2. getEndian()**

```
public java.nio.ByteOrder getEndian();
```

Return the byteorder used when loading values.

**Parameters**

<i>return</i>	the byteorder
---------------	---------------

**5.3. readBoolean()**

```
public boolean readBoolean();
```

Read boolean value.

**Parameters**

<i>return</i>	the value
---------------	-----------

**5.4. readByte()**

```
public byte readByte();
```

Read signed single byte value.

**Parameters**

<i>return</i>	the value
---------------	-----------

**5.5. readBytes(byte[])**

```
public void readBytes(byte[] bytes);
```

Read list of bytes.

#### Parameters

bytes	destination for read bytes
-------	----------------------------

### 5.6. **readBytes(byte[], int)**

```
public void readBytes(byte[] bytes,
                      int offset);
```

Read list of bytes to given offset.

#### Parameters

bytes	destination for read bytes
offset	offset in destination to write to

### 5.7. **readBytes(byte[], int, int)**

```
public void readBytes(byte[] bytes,
                      int offset,
                      int length);
```

Read given number of bytes to given offset.

#### Parameters

bytes	destination for read bytes
offset	offset in destination to write to
length	number of bytes to read

### 5.8. **readDouble()**

```
public double readDouble();
```

Read double-precision floating point value.

#### Parameters

return	the value
--------	-----------

### 5.9. **readFloat()**

```
public float readFloat();
```

Read single-precision floating point value.

#### Parameters

return	the value
--------	-----------

## 5.10. readInt()

```
public int readInt();
```

Read signed integer value.

### Parameters

<i>return</i>	the value
---------------	-----------

## 5.11. readMultiByte(int, String)

```
public String readMultiByte(int length,
                           String charSet);
```

Read multibyte string.

### Parameters

length	length of string to read
charSet	character set of string to read
<i>return</i>	the string

## 5.12. readObject()

```
public Object readObject();
```

Read arbitrary object.

### Parameters

<i>return</i>	the object
---------------	------------

## 5.13. readShort()

```
public short readShort();
```

Read signed short value.

### Parameters

<i>return</i>	the value
---------------	-----------

## 5.14. readUnsignedByte()

```
public int readUnsignedByte();
```

Read unsigned single byte value.

### Parameters

<i>return</i>	the value
---------------	-----------

## 5.15. readUnsignedInt()

```
public long readUnsignedInt();
```

Read unsigned integer value.

### Parameters

<i>return</i>	the value
---------------	-----------

## 5.16. readUnsignedShort()

```
public int readUnsignedShort();
```

Read unsigned short value.

### Parameters

<i>return</i>	the value
---------------	-----------

## 5.17. readUTF()

```
public String readUTF();
```

Read UTF-8 encoded string.

### Parameters

<i>return</i>	the string
---------------	------------

## 5.18. readUTFBytes(int)

```
public String readUTFBytes(int length);
```

Read UTF-8 encoded string with given length.

### Parameters

length	the length of the string
<i>return</i>	the string

## 5.19. setEndian(ByteOrder)

```
public void setEndian(java.nio.ByteOrder endian);
```

Set the byteorder to use when loading values.

### Parameters

endian	the byteorder to use
--------	----------------------

# 6. Interface IDataOutput

Interface implemented by classes that provide a way to store custom objects.

## 6.1. Synopsis

```

public interface IDataOutput {
// Public Methods

    public java.nio.ByteOrder getEndian();

    public void setEndian(java.nio.ByteOrder endian);

    public void writeBoolean(boolean value);

    public void writeByte(byte value);

    public void writeBytes(byte[] bytes);

    public void writeBytes(byte[] bytes,
                         int offset);

    public void writeBytes(byte[] bytes,
                         int offset,
                         int length);

    public void writeDouble(double value);

    public void writeFloat(float value);

    public void writeInt(int value);

    public void writeMultiByte(String value,
                           String encoding);

    public void writeObject(Object value);

    public void writeShort(short value);

    public void writeUTF(String value);

    public void writeUTFBytes(String value);

    public void writeUnsignedInt(long value);

}

```

### See Also

`writeExternal(org.red5.io.amf3.IDataOutput)` , [Adobe Livedocs \(external\) \[http://livedocs.adobe.com/flex/2/langref/flash/utils/IDataOutput.html\]](#)

## 6.2. `getEndian()`

```
public java.nio.ByteOrder getEndian();
```

Return the byteorder used when storing values.

## Parameters

<i>return</i>	the byteorder
---------------	---------------

**6.3. setEndian(ByteOrder)**

```
public void setEndian(java.nio.ByteOrder endian);
```

Set the byteorder to use when storing values.

## Parameters

endian	the byteorder to use
--------	----------------------

**6.4. writeBoolean(boolean)**

```
public void writeBoolean(boolean value);
```

Write boolean value.

## Parameters

value	the value
-------	-----------

**6.5. writeByte(byte)**

```
public void writeByte(byte value);
```

Write signed byte value.

## Parameters

value	the value
-------	-----------

**6.6. writeBytes(byte[])**

```
public void writeBytes(byte[] bytes);
```

Write multiple bytes.

## Parameters

bytes	the bytes
-------	-----------

**6.7. writeBytes(byte[], int)**

```
public void writeBytes(byte[] bytes,
                      int offset);
```

Write multiple bytes from given offset.

## Parameters

bytes	the bytes
-------	-----------

offset	offset in bytes to start writing from
--------	---------------------------------------

## 6.8. writeBytes(byte[], int, int)

```
public void writeBytes(byte[] bytes,
                      int offset,
                      int length);
```

Write given number of bytes from given offset.

### Parameters

bytes	the bytes
offset	offset in bytes to start writing from
length	number of bytes to write

## 6.9. writeDouble(double)

```
public void writeDouble(double value);
```

Write double-precision floating point value.

### Parameters

value	the value
-------	-----------

## 6.10. writeFloat(float)

```
public void writeFloat(float value);
```

Write single-precision floating point value.

### Parameters

value	the value
-------	-----------

## 6.11. writeInt(int)

```
public void writeInt(int value);
```

Write signed integer value.

### Parameters

value	the value
-------	-----------

## 6.12. writeMultiByte(String, String)

```
public void writeMultiByte(String value,
                           String encoding);
```

Write string in given character set.

**Parameters**

value	the string
encoding	the character set

**6.13. writeObject(Object)**

```
public void writeObject(Object value);
```

Write arbitrary object.

**Parameters**

value	the object
-------	------------

**6.14. writeShort(short)**

```
public void writeShort(short value);
```

Write signed short value.

**Parameters**

value	the value
-------	-----------

**6.15. writeUnsignedInt(long)**

```
public void writeUnsignedInt(long value);
```

Write unsigned integer value.

**Parameters**

value	the value
-------	-----------

**6.16. writeUTF(String)**

```
public void writeUTF(String value);
```

Write UTF-8 encoded string.

**Parameters**

value	the string
-------	------------

**6.17. writeUTFBytes(String)**

```
public void writeUTFBytes(String value);
```

Write UTF-8 encoded string as byte array. This string is stored without informations about its length, so `readUTFBytes(int)` must be used to load it.

**Parameters**

value	the string
-------	------------

## 7. Interface IExternalizable

Interface that needs to be implemented by classes that serialize / deserialize themselves.

### 7.1. Synopsis

```
public interface IExternalizable {
// Public Methods

    public void readExternal(IDataInput input);

    public void writeExternal(IDataOutput output);

}
```

#### See Also

Adobe Livedocs (external) [<http://livedocs.adobe.com/flex/2/langref/flash/utils/IExternalizable.html>]

### 7.2. readExternal(IDataInput)

```
public void readExternal(IDataInput input);
```

Load custom object from stream.

#### Parameters

input	object to be used for data loading
-------	------------------------------------

### 7.3. writeExternal(IDataOutput)

```
public void writeExternal(IDataOutput output);
```

Store custom object to stream.

#### Parameters

output	object to be used for data storing
--------	------------------------------------

## 8. Class Input

Input for Red5 data (AMF3) types

### 8.1. Synopsis

```
public class Input extends, org.?red5.?io.?amf.?Input
    implements, org.?red5.?io.?object.?Input {
// Protected Fields

    protected static org.slf4j.Logger log ;
```

## Package org.?red5.?io.?amf3

```
// Public Constructors

    public Input(org.apache.mina.common.ByteBuffer buf);

    public Input(org.apache.mina.common.ByteBuffer buf,
                org.red5.io.amf3.Input.RefStorage refStorage);

// Public Methods

    public void enforceAMF3();

    public String getString();

    public Object readArray(org.red5.io.object.Deserializer deserializer);

    public Boolean readBoolean();

    public ByteArray readByteArray();

    public Object readCustom();

    public byte readDataType();

    public java.util.Date readDate();

    public Object readMap(org.red5.io.object.Deserializer deserializer);

    public Object readNull();

    public Number readNumber();

    public Object readObject(org.red5.io.object.Deserializer deserializer);

    public Object readReference();

    public String readString();

    public org.w3c.dom.Document readXML();

    public void reset();

// Protected Methods

    protected org.apache.mina.common.ByteBuffer getBuffer();

    protected Object newInstance(String className);

}
```

**Methods inherited from org.red5.io.amf.Input:** getPropertyType , getString , hasMoreProperties , newInstance , readArray , readBean , readBoolean , readByteArray , readCustom , readDataType , readDate , readKeyValues , readMap , readNull , readNumber , readObject , readPropertyName , readReference , readSimpleObject , readString , readXML , reset , skipEndObject , skipPropertySeparator

**Methods inherited from org.red5.io.object.BaseInput:** clearReferences , getReference , storeReference

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.io.amf.Input:** buf , currentDataType , log

**Fields inherited from org.red5.io.object.BaseInput:** referenceMode , refId , refMap

## 8.2. Input(ByteBuffer)

```
public Input(org.apache.mina.common.ByteBuffer buf);
```

Creates Input object for AMF3 from byte buffer

### Parameters

buf	Byte buffer
-----	-------------

## 8.3. Input(ByteBuffer, Input.RefStorage)

```
public Input(org.apache.mina.common.ByteBuffer buf,
org.red5.io.amf3.Input.RefStorage refStorage);
```

Creates Input object for AMF3 from byte buffer and initializes references from passed RefStorage

### Parameters

buf	
-----	--

refStorage	
------------	--

## 8.4. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 8.5. enforceAMF3()

```
public void enforceAMF3();
```

Force using AMF3 everywhere

## 8.6. getBuffer()

```
protected org.apache.mina.common.ByteBuffer getBuffer();
```

Provide access to raw data.

### Parameters

<i>return</i>	ByteBuffer
---------------	------------

## 8.7. getString()

```
public String getString();
```

**Specified by:** Method getString in interface Input

Read a string without the string type header.

### Parameters

<i>return</i>	String
---------------	--------

**Description copied from interface: getString**

## 8.8. newInstance(String)

```
protected Object newInstance(String className);
```

## 8.9. readArray(Deserializer)

```
public Object readArray(org.red5.io.object.Deserializer deserializer);
```

**Specified by:** Method readArray in interface Input

Returns an array

### Parameters

<i>return</i>	int Length of array
---------------	---------------------

## 8.10. readBoolean()

```
public Boolean readBoolean();
```

**Specified by:** Method readBoolean in interface Input

Reads a boolean

### Parameters

<i>return</i>	boolean Boolean value
---------------	-----------------------

## 8.11. readByteArray()

```
public ByteArray readByteArray();
```

**Specified by:** Method readByteArray in interface Input

Read ByteArray object.

### Parameters

<i>return</i>	ByteArray object
---------------	------------------

**Description copied from interface: readByteArray**

## 8.12. readCustom()

public Object readCustom();
-----------------------------

**Specified by:** Method readCustom in interface Input

Reads Custom

### Parameters

<i>return</i>	Object Custom type object
---------------	---------------------------

## 8.13. readDataType()

public byte readDataType();
-----------------------------

**Specified by:** Method readDataType in interface Input

Reads the data type

### Parameters

<i>return</i>	byte Data type
---------------	----------------

## 8.14. readDate()

public java.util.Date readDate();
-----------------------------------

**Specified by:** Method readDate in interface Input

Returns a date

### Parameters

<i>return</i>	Date Date object
---------------	------------------

## 8.15. readMap(Deserializer)

public Object readMap(org.red5.io.object.Deserializer deserializer);
--

**Specified by:** Method readMap in interface Input

Read a map containing key - value pairs. This can result in a List or Map being deserialized depending on the map type found.

### Parameters

<i>return</i>	Map
---------------	-----

**Description copied from interface: readMap**

## 8.16. readNull()

```
public Object readNull();
```

**Specified by:** Method readNull in interface Input

Reads a null (value)

**Parameters**

<i>return</i>	Object null
---------------	-------------

## 8.17. readNumber()

```
public Number readNumber();
```

**Specified by:** Method readNumber in interface Input

Reads a Number

**Parameters**

<i>return</i>	Number Number
---------------	---------------

## 8.18. readObject(Deserializer)

```
public Object readObject(org.red5.io.object.Deserializer deserializer);
```

**Specified by:** Method readObject in interface Input

Read an object.

**Parameters**

<i>return</i>	object
---------------	--------

**Description copied from interface: readObject**

## 8.19. readReference()

```
public Object readReference();
```

**Specified by:** Method readReference in interface Input

Read reference to Complex Data Type. Objects that are collaborators (properties) of other objects must be stored as references in map of id-reference pairs.

## 8.20. readString()

```
public String readString();
```

**Specified by:** Method readString in interface Input

Reads a string

#### Parameters

<i>return</i>	String String
---------------	---------------

## 8.21. readXML()

```
public org.w3c.dom.Document readXML();
```

**Specified by:** Method readXML in interface Input

Read XML document

## 8.22. reset()

```
public void reset();
```

Resets map

# 9. Class Input.ClassReference

Holds informations about already deserialized classes.

## 9.1. Synopsis

```
protected class Input.ClassReference {
// Protected Fields

    protected java.util.List<java.lang.String> attributeNames ;

    protected String className ;

    protected int type ;

// Public Constructors

    public Input.ClassReference(String className,
                               int type,
                               java.util.List<java.lang.String> attributeNames);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 9.2. Input.ClassReference(String, int, List<String>)

```
public Input.ClassReference(String className,
                           int type,
                           java.util.List<java.lang.String> attributeNames);
```

Create new informations about a class.

## 9.3. attributeNames

```
protected java.util.List<java.lang.String> attributeNames ;
```

Names of the attributes of the class.

## 9.4. className

```
protected String className ;
```

Name of the serialized class.

## 9.5. type

```
protected int type ;
```

Type of the class.

# 10. Class Input.PendingObject

Dummy class that is stored as reference for objects currently being serialized that reference themselves.

## 10.1. Synopsis

```
protected class Input.PendingObject {
    // Protected Constructors

    protected Input.PendingObject();

    // Public Methods

    public void addPendingProperty(Object obj,
                                    Class<?> klass,
                                    String name);

    public void resolveProperties(Object result);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

# 11. Class Input.RefStorage

Class used to collect AMF3 references. In AMF3 references should be collected through the whole "body" (across several Input objects).

## 11.1. Synopsis

```
public static class Input.RefStorage {
    // Public Constructors
```

```
public Input.RefStorage();
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 12. Class Output

AMF3 output writer

### 12.1. Synopsis

```
public class Output extends, org.?red5.?io.?amf.?Output
    implements, org.?red5.?io.?object.?Output {
// Protected Fields

    protected static org.slf4j.Logger log ;

// Public Constructors

    public Output(org.apache.mina.common.ByteBuffer buf);

// Public Methods

    public void enforceAMF3();

    public void putString(String string);

    public boolean supportsDataType(byte type);

    public void writeArray(Object array,
                          org.red5.io.object.Serializer serializer);

    public void writeArray(Object[] array,
                          org.red5.io.object.Serializer serializer);

    public void writeArray(java.util.Collection<?> array,
                          org.red5.io.object.Serializer serializer);

    public void writeBoolean(Boolean bol);

    public void writeByteArray(ByteArray array);

    public void writeDate(java.util.Date date);

    public void writeMap(java.util.Collection<?> array,
                       org.red5.io.object.Serializer serializer);

    public void writeMap(java.util.Map<java.lang.Object, java.lang.Object> map,
                       org.red5.io.object.Serializer serializer);

    public void writeNull();

    public void writeNumber(Number num);
```

```

public void writeObject(Object object,
                      org.red5.io.object.Serializer serializer);

public void writeObject(java.util.Map<java.lang.Object, java.lang.Object> map,
                      org.red5.io.object.Serializer serializer);

public void writeRecordSet(org.red5.io.object.RecordSet recordset,
                         org.red5.io.object.Serializer serializer);

public void writeString(String string);

public void writeXML(org.w3c.dom.Document xml);

// Protected Methods

protected static byte[] encodeString(String string);

protected org.apache.mina.common.ByteBuffer getBuffer();

protected void putInteger(long value);

protected void putString(String str,
                        byte[] encoded);

protected void writeAMF3();

protected void writeArbitraryObject(Object object,
                                   org.red5.io.object.Serializer serializer);

}

```

**Methods inherited from org.red5.io.amf.Output:** buf , checkWriteReference , dontSerializeField , encodeString , isCustom , putString , reset , supportsDataType , writeArbitraryObject , writeArray , writeBoolean , writeByteArray , writeCustom , writeDate , writeMap , writeNull , writeNumber , writeObject , writeRecordSet , writeReference , writeString , writeXML

**Methods inherited from org.red5.io.object.BaseOutput:** clearReferences , getReferenceId , hasReference , storeReference

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.io.amf.Output:** buf , log , stringCache

**Fields inherited from org.red5.io.object.BaseOutput:** refId , refMap

#### See Also

org.red5.io.amf3.AMF3 , org.red5.io.amf3.Input

## 12.2. Output(ByteBuffer)

```
public Output(org.apache.mina.common.ByteBuffer buf);
```

Constructor of AMF3 output.

#### Parameters

buf	instance of ByteBuffer
-----	------------------------

See Also

[org.apache.mina.common.ByteBuffer](#)

## 12.3. encodeString(String)

```
protected static byte[] encodeString(String string);
```

## 12.4. enforceAMF3()

```
public void enforceAMF3();
```

Force using AMF3 everywhere

## 12.5. getBuffer()

```
protected org.apache.mina.common.ByteBuffer getBuffer();
```

Provide access to raw data.

#### Parameters

return	ByteBuffer
--------	------------

## 12.6. putInteger(long)

```
protected void putInteger(long value);
```

## 12.7. putString(String)

```
public void putString(String string);
```

**Specified by:** Method putString in interface Output

## 12.8. putString(String, byte[])

```
protected void putString(String str,
                        byte[] encoded);
```

## 12.9. supportsDataType(byte)

```
public boolean supportsDataType(byte type);
```

**Specified by:** Method supportsDataType in interface Output

## 12.10. writeArbitraryObject(Object, Serializer)

```
protected void writeArbitraryObject(Object object,
```

```
org.red5.io.object.Serializer serializer);
```

## 12.11. writeArray(Collection<?>, Serializer)

```
public void writeArray(java.util.Collection<?> array,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeArray in interface Output

Write array.

## 12.12. writeArray(Object[], Serializer)

```
public void writeArray(Object[] array,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeArray in interface Output

Write array.

## 12.13. writeArray(Object, Serializer)

```
public void writeArray(Object array,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeArray in interface Output

Write primitive array.

## 12.14. writeBoolean(Boolean)

```
public void writeBoolean(Boolean bol);
```

**Specified by:** Method writeBoolean in interface Output

Write boolean

## 12.15. writeByteArray(ByteArray)

```
public void writeByteArray(ByteArray array);
```

**Specified by:** Method writeByteArray in interface Output

Write ByteArray object (AMF3 only).

## 12.16. writeDate(Date)

```
public void writeDate(java.util.Date date);
```

**Specified by:** Method writeDate in interface Output

Write date

## 12.17. writeMap(Collection<?>, Serializer)

```
public void writeMap(java.util.Collection<?> array,
```

```
org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeMap in interface Output

Write array as map.

## 12.18. writeMap(Map<Object, Object>, Serializer)

```
public void writeMap(java.util.Map<java.lang.Object, java.lang.Object> map,
                     org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeMap in interface Output

Write map.

## 12.19. writeNull()

```
public void writeNull();
```

**Specified by:** Method writeNull in interface Output

## 12.20. writeNumber(Number)

```
public void writeNumber(Number num);
```

**Specified by:** Method writeNumber in interface Output

Write number

## 12.21. writeObject(Map<Object, Object>, Serializer)

```
public void writeObject(java.util.Map<java.lang.Object, java.lang.Object> map,
                       org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeObject in interface Output

Write map as object.

## 12.22. writeObject(Object, Serializer)

```
public void writeObject(Object object,
                       org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeObject in interface Output

Write object.

## 12.23. writeRecordSet(RecordSet, Serializer)

```
public void writeRecordSet(org.red5.io.object.RecordSet recordset,
                          org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeRecordSet in interface Output

Write recordset.

## 12.24. writeString(String)

```
public void writeString(String string);
```

**Specified by:** Method writeString in interface Output

Write string

## 12.25. writeXML(Document)

```
public void writeXML(org.w3c.dom.Document xml);
```

**Specified by:** Method writeXML in interface Output

Write XML object

# 1. Class ExecutorFilter

A filter that forward events to java.util.concurrent.Executor. You can apply various thread model by inserting this filter to the org.apache.mina.common.IoFilterChain. This filter is usually inserted by org.apache.mina.common.ThreadModel automatically, so you don't need to add this filter in most cases.

Please note that this filter doesn't manage the life cycle of the underlying java.util.concurrent.Executor. You have to destroy or stop it by yourself. Apache Directory Project [mailto:dev@directory.apache.org]

## 1.1. Synopsis

```
public class ExecutorFilter extends org.apache.mina.common.IoFilterAdapter {  
    // Public Constructors  
  
    public ExecutorFilter();  
  
    public ExecutorFilter(int corePoolSize,  
                         int maximumPoolSize,  
                         long keepAliveTime);  
  
    public ExecutorFilter(java.util.concurrent.Executor executor);  
  
    // Public Methods  
  
    public void exceptionCaught(org.apache.mina.common.IoFilter.NextFilter nextFilter,  
                               org.apache.mina.common.IoSession session,  
                               Throwable cause);  
  
    public void filterClose(org.apache.mina.common.IoFilter.NextFilter nextFilter,  
                           org.apache.mina.common.IoSession session)  
        throws Exception;  
  
    public void filterWrite(org.apache.mina.common.IoFilter.NextFilter nextFilter,  
                           org.apache.mina.common.IoSession session,  
                           org.apache.mina.common.IoFilter.WriteRequest writeRequest);  
  
    public java.util.concurrent.Executor getExecutor();  
  
    public void messageReceived(org.apache.mina.common.IoFilter.NextFilter nextFilter,  
                               org.apache.mina.common.IoSession session,  
                               Object message);  
  
    public void messageSent(org.apache.mina.common.IoFilter.NextFilter nextFilter,  
                           org.apache.mina.common.IoSession session,  
                           Object message);  
  
    public void sessionClosed(org.apache.mina.common.IoFilter.NextFilter nextFilter,  
                            org.apache.mina.common.IoSession session);  
  
    public void sessionCreated(org.apache.mina.common.IoFilter.NextFilter nextFilter,  
                             org.apache.mina.common.IoSession session);  
  
    public void sessionIdle(org.apache.mina.common.IoFilter.NextFilter nextFilter,  
                           org.apache.mina.common.IoSession session,  
                           org.apache.mina.common.IdleStatus status);
```

```

public void sessionOpened(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                         org.apache.mina.common.IoSession session);

// Protected Methods

protected void processEvent(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                           org.apache.mina.common.IoSession session,
                           org.red5.io.filter.ExecutorFilter.EventType type,
                           Object data);

}

```

**Methods inherited from org.apache.mina.common.IoFilterAdapter:** `destroy`, `exceptionCaught`, `filterClose`, `filterWrite`, `init`, `messageReceived`, `messageSent`, `onPostAdd`, `onPostRemove`, `onPreAdd`, `onPreRemove`, `sessionClosed`, `sessionCreated`, `sessionIdle`, `sessionOpened`

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

## 1.2. ExecutorFilter()

```
public ExecutorFilter();
```

Creates a new instance with the default thread pool implementation (`new ThreadPoolExecutor(16, 16, 60, TimeUnit.SECONDS, new LinkedBlockingQueue())`).

## 1.3. ExecutorFilter(Executor)

```
public ExecutorFilter(java.util.concurrent.Executor executor);
```

Creates a new instance with the specified `executor`.

### Parameters

executor	Executor
----------	----------

## 1.4. ExecutorFilter(int, int, long)

```
public ExecutorFilter(int corePoolSize,
                      int maximumPoolSize,
                      long keepAliveTime);
```

Creates new instance with specified core pool size, maximum pool size and keep alive time

### Parameters

corePoolSize	Core pool size
maximumPoolSize	Maximum pool size
keepAliveTime	Keep alive time (in seconds)

## 1.5. exceptionCaught(IoFilter.NextFilter, IoSession, Throwable)

```
public void exceptionCaught(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                           org.apache.mina.common.IoSession session,
                           Throwable cause);
```

## 1.6. filterClose(IoFilter.NextFilter, IoSession)

```
public void filterClose(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                       org.apache.mina.common.IoSession session)
                     throws Exception;
```

## 1.7. filterWrite(IoFilter.NextFilter, IoSession, IoFilter.WriteRequest)

```
public void filterWrite(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                       org.apache.mina.common.IoSession session,
                       org.apache.mina.common.IoFilter.WriteRequest writeRequest);
```

## 1.8. getExecutor()

```
public java.util.concurrent.Executor getExecutor();
```

Returns the underlying java.util.concurrent.Executor instance this filter uses.

### Parameters

<i>return</i>	Executor object
---------------	-----------------

## 1.9. messageReceived(IoFilter.NextFilter, IoSession, Object)

```
public void messageReceived(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                           org.apache.mina.common.IoSession session,
                           Object message);
```

## 1.10. messageSent(IoFilter.NextFilter, IoSession, Object)

```
public void messageSent(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                       org.apache.mina.common.IoSession session,
                       Object message);
```

## 1.11. processEvent(IoFilter.NextFilter, IoSession, ExecutorFilter.EventType, Object)

```
protected void processEvent(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                           org.apache.mina.common.IoSession session,
                           org.red5.io.filter.ExecutorFilter.EventType type,
                           Object data);
```

Handles event

### Parameters

nextFilter	Next filter in queue
session	IoSession object (connection between two ends)

type	Event type
data	Event data

## 1.12. sessionClosed(IoFilter.NextFilter, IoSession)

```
public void sessionClosed(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                        org.apache.mina.common.IoSession session);
```

## 1.13. sessionCreated(IoFilter.NextFilter, IoSession)

```
public void sessionCreated(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                           org.apache.mina.common.IoSession session);
```

## 1.14. sessionIdle(IoFilter.NextFilter, IoSession, IdleStatus)

```
public void sessionIdle(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                       org.apache.mina.common.IoSession session,
                       org.apache.mina.common.IdleStatus status);
```

## 1.15. sessionOpened(IoFilter.NextFilter, IoSession)

```
public void sessionOpened(org.apache.mina.common.IoFilter.NextFilter nextFilter,
                         org.apache.mina.common.IoSession session);
```

# 2. Class ExecutorFilter.Event

Connection event

## 2.1. Synopsis

```
protected static class ExecutorFilter.Event {
    // Public Methods

    public Object getData();

    public org.apache.mina.common.IoFilter.NextFilter getNextFilter();

    public org.red5.io.filter.ExecutorFilter.EventType getType();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. getData()

```
public Object getData();
```

Getter for event data

Parameters

return	Event data
--------	------------

## 2.3. getNextFilter()

public org.apache.mina.common.IoFilter.NextFilter getNextFilter();	
--	--

Getter for next filter in queue

Parameters	
------------	--

return	Next filter
--------	-------------

## 2.4. getType()

public org.red5.io.filter.ExecutorFilter.EventType getType();	
---	--

Getter for type

Parameters	
------------	--

return	Type of event
--------	---------------

## 3. Class ExecutorFilter.EventType

Type of event

### 3.1. Synopsis

```
protected static class ExecutorFilter.EventType {
    // Public Static Fields

        public static final org.red5.io.filter.ExecutorFilter.EventType CLOSED ;
        public static final org.red5.io.filter.ExecutorFilter.EventType EXCEPTION ;
        public static final org.red5.io.filter.ExecutorFilter.EventType IDLE ;
        public static final org.red5.io.filter.ExecutorFilter.EventType OPENED ;
        public static final org.red5.io.filter.ExecutorFilter.EventType READ ;
        public static final org.red5.io.filter.ExecutorFilter.EventType RECEIVED ;
        public static final org.red5.io.filter.ExecutorFilter.EventType SENT ;
        public static final org.red5.io.filter.ExecutorFilter.EventType WRITTEN ;

    // Public Methods

        public String toString();
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 3.2. CLOSED

```
public static final org.red5.io.filter.ExecutorFilter.EventType CLOSED ;
```

On connection closed

## 3.3. EXCEPTION

```
public static final org.red5.io.filter.ExecutorFilter.EventType EXCEPTION ;
```

On exception

## 3.4. IDLE

```
public static final org.red5.io.filter.ExecutorFilter.EventType IDLE ;
```

On connection idle

## 3.5. OPENED

```
public static final org.red5.io.filter.ExecutorFilter.EventType OPENED ;
```

On connection opened

## 3.6. READ

```
public static final org.red5.io.filter.ExecutorFilter.EventType READ ;
```

On data read

## 3.7. RECEIVED

```
public static final org.red5.io.filter.ExecutorFilter.EventType RECEIVED ;
```

On data received

## 3.8. SENT

```
public static final org.red5.io.filter.ExecutorFilter.EventType SENT ;
```

On data sent

## 3.9. WRITTEN

```
public static final org.red5.io.filter.ExecutorFilter.EventType WRITTEN ;
```

On data written

## 3.10. `toString()`

```
public String toString();
```

# 1. Class FLVHeader

FLVHeader parses out the contents of a FLV video file and returns the Header data

## 1.1. Synopsis

```
public class FLVHeader {  
    // Public Fields  
  
    public int dataOffset ;  
  
    public boolean flagAudio ;  
  
    public byte flagReserved01 ;  
  
    public byte flagReserved02 ;  
  
    public boolean flagVideo ;  
  
    public byte[] signature ;  
  
    public byte version ;  
  
    // Public Constructors  
  
    public FLVHeader();  
  
    // Public Methods  
  
    public int getDataOffset();  
  
    public boolean getFlagAudio();  
  
    public byte getFlagReserved01();  
  
    public byte getFlagReserved02();  
  
    public boolean getFlagVideo();  
  
    public byte[] getSignature();  
  
    public byte getVersion();  
  
    public void setDataOffset(int data_offset);  
  
    public void setFlagAudio(boolean flagAudio);  
  
    public void setFlagReserved01(byte flagReserved01);  
  
    public void setFlagReserved02(byte flagReserved02);  
  
    public void setFlagVideo(boolean type_flags_video);  
  
    public void setSignature(byte[] signature);
```

```

public void setTypeFlags(byte typeFlags);

public void setVersion(byte version);

public String toString();

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

#### See Also

OSFlash (external) [[http://osflash.org/flv#flv\\_header](http://osflash.org/flv#flv_header)]

## 1.2. dataOffset

```
public int dataOffset ;
```

reserved for data up to 4,294,967,295

## 1.3. flagAudio

```
public boolean flagAudio ;
```

Audio flag

## 1.4. flagReserved01

```
public byte flagReserved01 ;
```

Reserved flag, one

## 1.5. flagReserved02

```
public byte flagReserved02 ;
```

Reserved flag, two

## 1.6. flagVideo

```
public boolean flagVideo ;
```

Video flag

## 1.7. signature

```
public byte[] signature ;
```

Signature

## 1.8. version

```
public byte version ;
```

FLV version

## 1.9. getDataOffset()

```
public int getDataOffset();
```

Returns the data offset bytes

### Parameters

<i>return</i>	int Data offset
---------------	-----------------

## 1.10. getFlagAudio()

```
public boolean getFlagAudio();
```

Returns a boolean on whether this data contains audio

### Parameters

<i>return</i>	boolean <code>true</code> if this FLV header contains audio data, <code>false</code> otherwise
---------------	--

## 1.11. getFlagReserved01()

```
public byte getFlagReserved01();
```

Gets the FlagReserved01 which is a datatype specified in the Flash Specification

### Parameters

<i>return</i>	byte Flag reserved, first
---------------	---------------------------

## 1.12. getFlagReserved02()

```
public byte getFlagReserved02();
```

Gets the FlagReserved02 which is a datatype specified in the Flash Specification

### Parameters

<i>return</i>	byte FlagReserved02
---------------	---------------------

## 1.13. getFlagVideo()

```
public boolean getFlagVideo();
```

Returns a boolean on whether this data contains video

### Parameters

<i>return</i>	boolean <code>true</code> if this FLV header contains vide data, <code>false</code> otherwise
---------------	---

## 1.14. **getSignature()**

```
public byte[] getSignature();
```

Returns the signature bytes

### Parameters

<i>return</i>	byte[] Signature
---------------	------------------

## 1.15. **getVersion()**

```
public byte getVersion();
```

Gets the version byte

### Parameters

<i>return</i>	byte FLV version byte
---------------	-----------------------

## 1.16. **setDataOffset(int)**

```
public void setDataOffset(int data_offset);
```

Sets the data offset bytes

### Parameters

data_offset	Data offset
-------------	-------------

## 1.17. **setFlagAudio(boolean)**

```
public void setFlagAudio(boolean flagAudio);
```

Sets the audioflag on whether this data contains audio

### Parameters

flagAudio	<code>true</code> if this FLV header contains audio data, <code>false</code> otherwise
-----------	--

## 1.18. **setFlagReserved01(byte)**

```
public void setFlagReserved01(byte flagReserved01);
```

Sets the FlagReserved01 which is a datatype specified in the Flash Specification

### Parameters

flagReserved01	Flag reserved, first
----------------	----------------------

## 1.19. setFlagReserved02(byte)

```
public void setFlagReserved02(byte flagReserved02);
```

Sets the Flag Reserved02 which is a datatype specified in the Flash Specification

### Parameters

flagReserved02	FlagReserved02
----------------	----------------

## 1.20. setFlagVideo(boolean)

```
public void setFlagVideo(boolean type_flags_video);
```

Sets the audioflag on whether this data contains audio

### Parameters

type_flags_video	true if this FLV header contains video data, false otherwise
------------------	--

## 1.21. setSignature(byte[])

```
public void setSignature(byte[] signature);
```

Sets the signature bytes

### Parameters

signature	Signature
-----------	-----------

## 1.22. setTypeFlags(byte)

```
public void setTypeFlags(byte typeFlags);
```

Sets the type flags on whether this data is audio or video

### Parameters

typeFlags	Type flags determining data types (audio or video)
-----------	--

## 1.23. setVersion(byte)

```
public void setVersion(byte version);
```

Sets the version byte

### Parameters

version	FLV version byte
---------	------------------

## 1.24. toString()

```
public String toString();
```

Overrides the `toString` method so that a FLVHeader can be represented by its datatypes

#### Parameters

<code>return</code>	String String representation
---------------------	------------------------------

## 2. Interface IFLV

Represents FLV file

### 2.1. Synopsis

```
public interface IFLV extends, org.?red5.?io.?IStreamableFile {
// Public Methods

    public void flushHeaders()
        throws IOException;

    public java.util.Map getKeyFrameData();

    public org.red5.io.flv.meta.IMetaData getMetaData()
        throws FileNotFoundException;

    public boolean hasKeyFrameData();

    public boolean hasMetaData();

    public org.red5.io.ITagReader readerFromNearestKeyFrame(int seekPoint);

    public void refreshHeaders()
        throws IOException;

    public void setCache(org.red5.server.api.cache.ICacheStore cache);

    public void setKeyFrameData(java.util.Map keyframedata);

    public void setMetaData(org.red5.io.flv.meta.IMetaData metadata)
        throws FileNotFoundException, IOException;

    public void setMetaService(org.red5.io.flv.meta.IMetaService service);

    public org.red5.io.ITagWriter writerFromNearestKeyFrame(int seekPoint);

}
```

### 2.2. flushHeaders()

```
public void flushHeaders()
    throws IOException;
```

Flushes Header

IOException

Any I/O exception

## 2.3. getKeyFrameData()

```
public java.util.Map getKeyFrameData();
```

Gets the keyframe data

### Parameters

<i>return</i>	keyframedata Keyframe metadata
---------------	--------------------------------

## 2.4. getMetaData()

```
public org.red5.io.flv.meta.IMetaData getMetaData()
throws FileNotFoundException;
```

Returns a map of the metadata

### Parameters

<i>return</i>	metadata File metadata
---------------	------------------------

`FileNotFoundException`

File not found

## 2.5. hasKeyFrameData()

```
public boolean hasKeyFrameData();
```

Returns a boolean stating whether a flv has keyframedata

### Parameters

<i>return</i>	boolean <code>true</code> if file has keyframe metadata, <code>false</code> otherwise
---------------	---

## 2.6. hasMetaData()

```
public boolean hasMetaData();
```

Returns a boolean stating whether the flv has metadata

### Parameters

<i>return</i>	boolean <code>true</code> if file has injected metadata, <code>false</code> otherwise
---------------	---

## 2.7. readerFromNearestKeyFrame(int)

```
public org.red5.io.ITagReader readerFromNearestKeyFrame(int seekPoint);
```

Returns a Reader closest to the nearest keyframe

### Parameters

<code>seekPoint</code>	Point in file we are seeking around
------------------------	-------------------------------------

`return`

reader Tag reader closest to that point

## 2.8. refreshHeaders()

```
public void refreshHeaders()
    throws IOException;
```

Refreshes the headers. Usually used after data is added to the flv file

`IOException`

Any I/O exception

## 2.9. setCache(ICacheStore)

```
public void setCache(org.red5.server.api.cache.ICacheStore cache);
```

Sets the caching implemenation

Parameters

cache	
-------	--

## 2.10. setKeyFrameData(Map)

```
public void setKeyFrameData(java.util.Map keyframedata);
```

Sets the keyframe data of a flv file

Parameters

keyframedata	Keyframe metadata
--------------	-------------------

## 2.11. setMetaData(IMetaData)

```
public void setMetaData(org.red5.io.flv.meta.IMetaData metadata)
    throws FileNotFoundException, IOException;
```

Sets the metadata

Parameters

metadata	Metadata object
----------	-----------------

`FileNotFoundException`

File not found

`IOException`

Any other I/O exception

## 2.12. setMetaService(IMetaService)

```
public void setMetaService(org.red5.io.flv.meta.IMetaService service);
```

Sets the MetaService through Spring

**Parameters**

service	Metadata service
---------	------------------

**2.13. writerFromNearestKeyFrame(int)**

```
public org.red5.io.ITagWriter writerFromNearestKeyFrame(int seekPoint);
```

Returns a Writer based on the nearest key frame

**Parameters**

seekPoint	Point in file we are seeking around
<i>return</i>	writer Tag writer closest to that point

**3. Interface IFLVService**

A FLVService sets up the service and hands out FLV objects to its callers

**3.1. Synopsis**

```
public interface IFLVService extends, org.?red5.?io.?IStreamableFileService {
// Public Methods

    public void setDeserializer(org.red5.io.object.Deserializer deserializer);

    public void setSerializer(org.red5.io.object.Serializer serializer);

}
```

**3.2. setDeserializer(Deserializer)**

```
public void setDeserializer(org.red5.io.object.Deserializer deserializer);
```

Sets the deserializer

**Parameters**

deserializer	Deserializer object
--------------	---------------------

**3.3. setSerializer(Serializer)**

```
public void setSerializer(org.red5.io.object.Serializer serializer);
```

Sets the serializer

**Parameters**

serializer	Serializer object
------------	-------------------

**4. Interface IKeyFrameDataAnalyzer**

Analyzes key frame data.

## 4.1. Synopsis

```
public interface IKeyFrameDataAnalyzer {
// Public Methods

    public org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta analyzeKeyFrames();
}

}
```

## 4.2. analyzeKeyFrames()

```
public org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta analyzeKeyFrames();
```

Analyze and return keyframe metadata.

### Parameters

<i>return</i>	Metadata object
---------------	-----------------

## 5. Class IKeyFrameDataAnalyzer.KeyFrameMeta

Keyframe metadata.

### 5.1. Synopsis

```
public static class IKeyFrameDataAnalyzer.KeyFrameMeta implements, java.?io.?Serializable {
// Public Fields

    public boolean audioOnly ;

    public long duration ;

    public long[] positions ;

    public int[] timestamps ;

// Public Constructors

    public IKeyFrameDataAnalyzer.KeyFrameMeta();

}

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 5.2. audioOnly

```
public boolean audioOnly ;
```

Only audio frames?

### 5.3. duration

```
public long duration ;
```

Duration in milliseconds

### 5.4. positions

```
public long[] positions ;
```

Keyframe positions

### 5.5. timestamps

```
public int[] timestamps ;
```

Keyframe timestamps

# 1. Class FLV

A FLVImpl implements the FLV api

## 1.1. Synopsis

```
public class FLVImplements, org.red5.io.flv.IFLV {  
    // Protected Fields  
  
    protected static org.slf4j.Logger log ;  
  
    // Public Constructors  
  
    public FLV();  
  
    public FLV(java.io.File file);  
  
    public FLV(java.io.File file,  
              boolean generateMetadata);  
  
    // Public Methods  
  
    public void flushHeaders()  
        throws IOException;  
  
    public org.red5.io.ITagWriter getAppendWriter()  
        throws IOException;  
  
    public java.util.Map getKeyFrameData();  
  
    public org.red5.io.flv.meta.IMetaData getMetaData()  
        throws FileNotFoundException;  
  
    public org.red5.io.ITagReader getReader()  
        throws IOException;  
  
    public org.red5.io.ITagWriter getWriter()  
        throws IOException;  
  
    public boolean hasKeyFrameData();  
  
    public boolean hasMetaData();  
  
    public org.red5.io.ITagReader readerFromNearestKeyFrame(int seekPoint);  
  
    public void refreshHeaders()  
        throws IOException;  
  
    public void setCache(org.red5.server.api.cache.ICacheStore cache);  
  
    public void setKeyFrameData(java.util.Map keyframedata);  
  
    public void setMetaData(org.red5.io.flv.meta.IMetaData meta)  
        throws IOException;  
  
    public void setMetaService(org.red5.io.flv.meta.IMetaService service);
```

```
public org.red5.io.ITagWriter writerFromNearestKeyFrame(int seekPoint);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. FLV()

```
public FLV();
```

Default constructor, used by Spring so that parameters may be injected.

## 1.3. FLV(File)

```
public FLV(java.io.File file);
```

Create FLV from given file source

### Parameters

file	File source
------	-------------

## 1.4. FLV(File, boolean)

```
public FLV(java.io.File file,
          boolean generateMetadata);
```

Create FLV from given file source and with specified metadata generation option

### Parameters

file	File source
generateMetadata	Metadata generation option

## 1.5. flushHeaders()

```
public void flushHeaders()
    throws IOException;
```

**Specified by:** Method flushHeaders in interface IFLV

Flushes Header

## 1.6. getAppendWriter()

```
public org.red5.io.ITagWriter getAppendWriter()
    throws IOException;
```

## 1.7. getKeyFrameData()

```
public java.util.Map getKeyFrameData();
```

**Specified by:** Method getKeyFrameData in interface IFLV

Gets the keyframe data

## 1.8. getMetaData()

```
public org.red5.io.flv.meta.IMetaData getMetaData()
throws FileNotFoundException;
```

**Specified by:** Method getMetaData in interface IFLV

Returns a map of the metadata

## 1.9. getReader()

```
public org.red5.io.ITagReader getReader()
throws IOException;
```

## 1.10. getWriter()

```
public org.red5.io.ITagWriter getWriter()
throws IOException;
```

## 1.11. hasKeyFrameData()

```
public boolean hasKeyFrameData();
```

**Specified by:** Method hasKeyFrameData in interface IFLV

Returns a boolean stating whether a flv has keyframedata

## 1.12. hasMetaData()

```
public boolean hasMetaData();
```

**Specified by:** Method hasMetaData in interface IFLV

Returns a boolean stating whether the flv has metadata

## 1.13. readerFromNearestKeyFrame(int)

```
public org.red5.io.ITagReader readerFromNearestKeyFrame(int seekPoint);
```

**Specified by:** Method readerFromNearestKeyFrame in interface IFLV

Returns a Reader closest to the nearest keyframe

## 1.14. refreshHeaders()

```
public void refreshHeaders()
throws IOException;
```

**Specified by:** Method refreshHeaders in interface IFLV

Refreshes the headers. Usually used after data is added to the flv file

## 1.15. setCache(ICacheStore)

```
public void setCache(org.red5.server.api.cache.ICacheStore cache);
```

**Specified by:** Method setCache in interface IFLV

Sets the cache implementation to be used.

### Parameters

cache	Cache store
-------	-------------

## 1.16. setKeyFrameData(Map)

```
public void setKeyFrameData(java.util.Map keyframedata);
```

**Specified by:** Method setKeyFrameData in interface IFLV

Sets the keyframe data of a flv file

## 1.17. setMetaData(IMetaData)

```
public void setMetaData(org.red5.io.flv.meta.IMetaData meta)
    throws IOException;
```

**Specified by:** Method setMetaData in interface IFLV

Sets the metadata

## 1.18. setMetaService(IMetaService)

```
public void setMetaService(org.red5.io.flv.meta.IMetaService service);
```

**Specified by:** Method setMetaService in interface IFLV

Sets the MetaService through Spring

## 1.19. writerFromNearestKeyFrame(int)

```
public org.red5.io.ITagWriter writerFromNearestKeyFrame(int seekPoint);
```

**Specified by:** Method writerFromNearestKeyFrame in interface IFLV

Returns a Writer based on the nearest key frame

## 2. Class FLVReader

A Reader is used to read the contents of a FLV file. NOTE: This class is not implemented as threading-safe. The caller should make sure the threading-safety.

### 2.1. Synopsis

```
public class FLVReader implements, org.?red5.?io.?IoConstants, org.?red5.?io.?ITagReader, org.?red5.?i
```

```

public FLVReader(java.io.File f)
    throws IOException;

public FLVReader(java.io.File f,
                 boolean generateMetadata)
    throws IOException;

public FLVReader(org.apache.mina.common.ByteBuffer buffer,
                 boolean generateMetadata);

// Public Static Methods

public static int getBufferSize();

public static String getBufferType();

public static long getDuration(java.io.File flvFile);

public static void setBufferSize(int bufferSize);

public static void setBufferType(String bufferType);

// Public Methods

public synchronized org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta analyzeKeyFrames();

public void close();

public void decodeHeader();

public int getAudioCodecId();

public long getBytesRead();

public long getDuration();

public org.red5.io.IStreamableFile getFile();

public org.apache.mina.common.ByteBuffer getFileData();

public int getOffset();

public long getTotalBytes();

public int getVideoCodecId();

public boolean hasMoreTags();

public boolean hasVideo();

public void position(long pos);

public synchronized org.red5.io.ITag readTag();

public void setKeyFrameCache(org.red5.io.IKeyFrameMetaCache keyframeCache);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. FLVReader(ByteBuffer, boolean)

```
public FLVReader(org.apache.mina.common.ByteBuffer buffer,
                 boolean generateMetadata);
```

Accepts mapped file bytes to construct internal members.

### Parameters

generateMetadata	true if metadata generation required, false otherwise
buffer	Byte buffer

## 2.3. FLVReader(File)

```
public FLVReader(java.io.File f)
                 throws IOException;
```

Creates FLV reader from file input stream.

### Parameters

f	File
---	------

## 2.4. FLVReader(File, boolean)

```
public FLVReader(java.io.File f,
                 boolean generateMetadata)
                 throws IOException;
```

Creates FLV reader from file input stream, sets up metadata generation flag.

### Parameters

f	File input stream
generateMetadata	true if metadata generation required, false otherwise

## 2.5. analyzeKeyFrames()

```
public synchronized org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta analyzeKeyFrames();
```

**Specified by:** Method analyzeKeyFrames in interface IKeyFrameDataAnalyzer

Key frames analysis may be used as a utility method so synchronize it.

### Parameters

return	Keyframe metadata
--------	-------------------

## 2.6. close()

```
public void close();
```

**Specified by:** Method close in interface ITagReader

Closes the reader and free any allocated memory.

## 2.7. decodeHeader()

```
public void decodeHeader();
```

**Specified by:** Method decodeHeader in interface ITagReader

Decode the header of the stream;

## 2.8. getBufferSize()

```
public static int getBufferSize();
```

Getter for buffer size.

### Parameters

<i>return</i>	Value for property 'bufferSize'
---------------	---------------------------------

## 2.9. getBufferType()

```
public static String getBufferType();
```

Getter for buffer type (auto, direct or heap).

### Parameters

<i>return</i>	Value for property 'bufferType'
---------------	---------------------------------

## 2.10. getBytesRead()

```
public long getBytesRead();
```

**Specified by:** Method getBytesRead in interface ITagReader

Returns the amount of bytes read

## 2.11. getDuration()

```
public long getDuration();
```

**Specified by:** Method getDuration in interface ITagReader

Return length in seconds

## 2.12. getFile()

```
public org.red5.io.IStreamableFile getFile();
```

**Specified by:** Method getFile in interface ITagReader

Return the file that is loaded.

## 2.13. getFileData()

```
public org.apache.mina.common.ByteBuffer getFileData();
```

Returns the file buffer.

**Parameters**

<i>return</i>	File contents as byte buffer
---------------	------------------------------

## 2.14. getOffset()

```
public int getOffset();
```

**Specified by:** Method getOffset in interface ITagReader

Returns the offset length

## 2.15. getTotalBytes()

```
public long getTotalBytes();
```

**Specified by:** Method getTotalBytes in interface ITagReader

Get the total readable bytes in a file or ByteBuffer.

**Parameters**

<i>return</i>	Total readable bytes
---------------	----------------------

## 2.16. hasMoreTags()

```
public boolean hasMoreTags();
```

**Specified by:** Method hasMoreTags in interface ITagReader

Returns a boolean stating whether the FLV has more tags

## 2.17. hasVideo()

```
public boolean hasVideo();
```

**Specified by:** Method hasVideo in interface ITagReader

Check if the reader also has video tags.

## 2.18. position(long)

```
public void position(long pos);
```

**Specified by:** Method position in interface ITagReader

Put the current position to pos. The caller must ensure the pos is a valid one (eg. not sit in the middle of a frame).

#### Parameters

pos	New position in file. Pass <code>Long.MAX_VALUE</code> to seek to end of file.
-----	--

## 2.19. **readTag()**

```
public synchronized org.red5.io.ITag readTag();
```

**Specified by:** Method `readTag` in interface `ITagReader`

Returns a Tag object

## 2.20. **setBufferSize(int)**

```
public static void setBufferSize(int bufferSize);
```

Setter for property 'bufferSize'.

#### Parameters

bufferSize	Value to set for property 'bufferSize'
------------	--

## 2.21. **setBufferType(String)**

```
public static void setBufferType(String bufferType);
```

Setter for buffer type.

#### Parameters

bufferType	Value to set for property 'bufferType'
------------	--

## 3. Class FLVService

A `FLVService` sets up the service and hands out `FLV` objects to its callers.

### 3.1. Synopsis

```
public class FLVService extends, org.?red5.?io.?BaseStreamableFileService
    implements, org.?red5.?io.?flv.?IFLVService {
    // Public Constructors

    public FLVService();

    // Public Methods

    public org.red5.io.object.Deserializer getDeserializer();

    public StringgetExtension();

    public StringgetPrefix();
```

```

    public org.red5.io.object.Serializer getSerializer();

    public org.red5.io.IStreamableFile getStreamableFile(java.io.File file)
        throws IOException;

    public void setDeserializer(org.red5.io.object.Deserializer deserializer);

    public void setGenerateMetadata(boolean generate);

    public void setSerializer(org.red5.io.object.Serializer serializer);

}

```

**Methods inherited from org.red5.io.BaseStreamableFileService:** canHandle ,  
getExtension , getPrefix , getStreamableFile , prepareFilename

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

### 3.2. getDeserializer()

```
public org.red5.io.object.Deserializer getDeserializer();
```

Getter for deserializer

#### Parameters

<i>return</i>	Deserializer
---------------	--------------

### 3.3. getExtension()

```
public String getExtension();
```

### 3.4. getPrefix()

```
public String getPrefix();
```

### 3.5. getSerializer()

```
public org.red5.io.object.Serializer getSerializer();
```

Getter for serializer

#### Parameters

<i>return</i>	Serializer
---------------	------------

### 3.6. getStreamableFile(File)

```
public org.red5.io.IStreamableFile getStreamableFile(java.io.File file)
    throws IOException;
```

### 3.7. setDeserializer(Deserializer)

```
public void setDeserializer(org.red5.io.object.Deserializer deserializer);
```

**Specified by:** Method setDeserializer in interface IFLVService

Sets the deserializer

### 3.8. setGenerateMetadata(boolean)

```
public void setGenerateMetadata(boolean generate);
```

Generate metadata or not

#### Parameters

generate	true if there's need to generate metadata, false otherwise
----------	--

### 3.9. setSerializer(Serializer)

```
public void setSerializer(org.red5.io.object.Serializer serializer);
```

**Specified by:** Method setSerializer in interface IFLVService

Sets the serializer

## 4. Class FLVWriter

A Writer is used to write the contents of a FLV file

### 4.1. Synopsis

```
public class FLVWriter implements, org.?red5.?io.?ITagWriter {
// Public Constructors

    public FLVWriter(java.io.File file,
                    boolean append);

    public FLVWriter(java.io.FileOutputStream fos,
                    boolean append);

// Public Methods

    public void close();

    public long getBytesWritten();

    public org.red5.io.IStreamableFile getFile();

    public int getOffset();

    public void setFLV(org.red5.io.flv.IFLV flv);

    public void setOffset(int offset);

    public void writeHeader()
```

```

    throws IOException;

    public boolean writeStream(byte[] b);

    public boolean writeTag(byte type,
                           org.apache.mina.common.ByteBuffer data)
    throws IOException;

    public boolean writeTag(org.red5.io.ITag tag)
    throws IOException;

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 4.2. FLVWriter(File, boolean)

```
public FLVWriter(java.io.File file,
                 boolean append);
```

Creates writer implementation with given file and last tag FLV.java uses this constructor so we have access to the file object

### Parameters

file	File output stream
lastTag	Last tag

## 4.3. FLVWriter(FileOutputStream, boolean)

```
public FLVWriter(java.io.FileOutputStream fos,
                 boolean append);
```

Creates writer implementation with given file output stream and last tag

### Parameters

fos	File output stream
lastTag	Last tag

## 4.4. close()

```
public void close();
```

**Specified by:** Method close in interface ITagWriter

Closes a Writer

## 4.5. getBytesWritten()

```
public long getBytesWritten();
```

**Specified by:** Method getBytesWritten in interface ITagWriter

Return the bytes written

## 4.6. getFile()

```
public org.red5.io.IStreamableFile getFile();
```

**Specified by:** Method getFile in interface ITagWriter

Return the file that is written.

## 4.7. getOffset()

```
public int getOffset();
```

**Specified by:** Method getOffset in interface ITagWriter

Return the offset

## 4.8. setFLV(IFLV)

```
public void setFLV(org.red5.io.flv.IFLV flv);
```

Setter for FLV object

Parameters

flv	FLV source
-----	------------

## 4.9. setOffset(int)

```
public void setOffset(int offset);
```

Setter for offset

Parameters

offset	Value to set for offset
--------	-------------------------

## 4.10. writeHeader()

```
public void writeHeader()
throws IOException;
```

**Specified by:** Method writeHeader in interface ITagWriter

Writes the header bytes

IOException

Any I/O exception

## 4.11. writeStream(byte[])

```
public boolean writeStream(byte[] b);
```

**Specified by:** Method writeStream in interface ITagWriter

Write a Stream to disk using bytes

## 4.12. writeTag(byte, ByteBuffer)

```
public boolean writeTag(byte type,
                      org.apache.mina.common.ByteBuffer data)
throws IOException;
```

**Specified by:** Method writeTag in interface ITagWriter

Write a Tag using bytes

## 4.13. writeTag(ITag)

```
public boolean writeTag(org.red5.io.ITag tag)
throws IOException;
```

**Specified by:** Method writeTag in interface ITagWriter

Writes a Tag object

# 5. Class Tag

A Tag represents the contents or payload of a FLV file.

## 5.1. Synopsis

```
public class Tag implements org.?red5.?io.?ITag {
// Public Constructors

    public Tag();

    public Tag(byte dataType,
              int timestamp,
              int bodySize,
              org.apache.mina.common.ByteBuffer body,
              int previousTagSize);

// Public Methods

    public byte getBitflags();

    public org.apache.mina.common.ByteBuffer getBody();

    public int getBodySize();

    public org.apache.mina.common.ByteBuffer getData();

    public byte getDataType();

    public int getPreviousTagSize();

    public int getPreviuosTagSize();

    public int getTimestamp();
```

```

    public byte getType();

    public void setBitflags(byte bitflags);

    public void setBody(org.apache.mina.common.ByteBuffer body);

    public void setBodySize(int bodySize);

    public void setData();

    public void setDataType(byte dataType);

    public void setPreviousTagSize(int size);

    public void setPreviuosTagSize(int previuosTagSize);

    public void setTimestamp(int timestamp);

    public void setType(byte type);

    public String toString();

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

#### See Also

OSFlash (external) [[http://osflash.org/flv#flv\\_tag](http://osflash.org/flv#flv_tag)]

## 5.2. Tag()

```
public Tag();
```

Constructs a new Tag.

## 5.3. Tag(byte, int, int, ByteBuffer, int)

```

public Tag(byte dataType,
          int timestamp,
          int bodySize,
          org.apache.mina.common.ByteBuffer body,
          int previousTagSize);

```

TagImpl Constructor

#### Parameters

dataType	Tag data type
timestamp	Timestamp
bodySize	Tag body size

body	Tag body
previousTagSize	Previous tag size information

## 5.4. getBitflags()

```
public byte getBitflags();
```

Getter for bit flags

Parameters	
<i>return</i>	Value for bit flags

## 5.5. getBody()

```
public org.apache.mina.common.ByteBuffer getBody();
```

**Specified by:** Method getBody in interface ITag

Return the body ByteBuffer

Parameters	
<i>return</i>	Tag body

## 5.6. getBodySize()

```
public int getBodySize();
```

**Specified by:** Method getBodySize in interface ITag

Return the size of the body

Parameters	
<i>return</i>	Tag body size

## 5.7. getData()

```
public org.apache.mina.common.ByteBuffer getData();
```

**Specified by:** Method getData in interface ITag

Returns the data as a ByteBuffer

## 5.8. getDataType()

```
public byte getDataType();
```

**Specified by:** Method getDataType in interface ITag

Get the data type

**Parameters**

<i>return</i>	Tag data type
---------------	---------------

**5.9. getPreviousTagSize()**

```
public int getPreviousTagSize();
```

**Specified by:** Method `getPreviousTagSize` in interface `ITag`

Return previous tag size

**Parameters**

<i>return</i>	Previous tag size
---------------	-------------------

**5.10. getPreviuosTagSize()**

```
public int getPreviuosTagSize();
```

Getter for previous tag size

**Parameters**

<i>return</i>	Value for previous tag size
---------------	-----------------------------

**5.11. getTimestamp()**

```
public int getTimestamp();
```

**Specified by:** Method `getTimestamp` in interface `ITag`

Return the timestamp

**Parameters**

<i>return</i>	Tag timestamp
---------------	---------------

**5.12. getType()**

```
public byte getType();
```

Getter for tag type

**Parameters**

<i>return</i>	Tag type
---------------	----------

**5.13. setBitflags(byte)**

```
public void setBitflags(byte bitflags);
```

Setter for bit flags

## Parameters

bitflags	Bit flags
----------	-----------

**5.14. setBody(ByteBuffer)**

```
public void setBody(org.apache.mina.common.ByteBuffer body);
```

**Specified by:** Method setBody in interface ITag

Set the body ByteBuffer.

**5.15. setBodySize(int)**

```
public void setBodySize(int bodySize);
```

**Specified by:** Method setBodySize in interface ITag

Set the size of the body.

**5.16. setData()**

```
public void setData();
```

Setter for tag data. Empty method.

**5.17. setDataType(byte)**

```
public void setDataType(byte dataType);
```

**Specified by:** Method setDataType in interface ITag

Set the data type.

**5.18. setPreviousTagSize(int)**

```
public void setPreviousTagSize(int size);
```

**Specified by:** Method setPreviousTagSize in interface ITag

Set the size of the previous tag.

**5.19. setPreviuosTagSize(int)**

```
public void setPreviuosTagSize(int previuosTagSize);
```

Setter for previous tag size

## Parameters

previuosTagSize	Value to set for previous tag size
-----------------	------------------------------------

**5.20. setTimestamp(int)**

```
public void setTimestamp(int timestamp);
```

**Specified by:** Method setTimestamp in interface ITag

Set the timestamp.

## 5.21. setType(byte)

```
public void setType(byte type);
```

Setter for tag type

Parameters

type	Tag type
------	----------

## 5.22. toString()

```
public String toString();
```

Prints out the contents of the tag

Parameters

<i>return</i>	Tag contents
---------------	--------------

# 1. Interface ICueType

Cue point type

## 1.1. Synopsis

```
public interface ICueType {  
    // Public Static Fields  
  
    public static final String EVENT = "event";  
  
    public static final String NAVIGATION = "navigation";  
}
```

## 1.2. EVENT

```
public static final String EVENT = "event";
```

Cue point type of event

## 1.3. NAVIGATION

```
public static final String NAVIGATION = "navigation";
```

Cue point type of navigation

# 2. Interface IMeta

IMeta is a Marker Interface CuePoint and MetaData both implement this interface

## 2.1. Synopsis

```
public interface IMeta extends, java.io.Serializable {  
}
```

# 3. Interface IMetaCue

ICuePoint defines contract methods for use with cuepoints

## 3.1. Synopsis

```
public interface IMetaCue extends, org.red5.io.flv.meta.IMeta, java.lang.Comparable {  
    // Public Methods  
  
    public String getName();  
  
    public double getTime();
```

```

public String getType();

public void setName(String name);

public void setTime(double d);

public void setType(String type);

}

```

### 3.2. getName()

```
public String getName();
```

Gets the name

#### Parameters

<i>return</i>	name Cue point name
---------------	---------------------

### 3.3. getTime()

```
public double getTime();
```

Gets the time

#### Parameters

<i>return</i>	time Timestamp
---------------	----------------

### 3.4. getType()

```
public String getType();
```

Gets the type

#### Parameters

<i>return</i>	type Cue point type
---------------	---------------------

### 3.5. setName(String)

```
public void setName(String name);
```

Sets the name

#### Parameters

name	Cue point name
------	----------------

### 3.6. setTime(double)

```
public void setTime(double d);
```

Sets the time

#### Parameters

d	Timestamp
---	-----------

### 3.7. setType(String)

```
public void setType(String type);
```

Sets the type type can be "event" or "navigation"

#### Parameters

type	Cue point type
------	----------------

## 4. Interface IMetaData

FLV MetaData interface

### 4.1. Synopsis

```
public interface IMetaData<K,V> extends, org.?red5.?io.?flv.?meta.?IMeta {
// Public Methods
```

```
    public int getAudioCodecId();
```

```
    public boolean getCanSeekToEnd();
```

```
    public double getDuration();
```

```
    public double getFrameRate();
```

```
    public int getHeight();
```

```
    public IMetaCue[] getMetaCue();
```

```
    public int getVideoCodecId();
```

```
    public int getVideoDataRate();
```

```
    public int getWidth();
```

```
    public void setAudioCodecId(int id);
```

```
    public void setCanSeekToEnd(boolean b);
```

```
    public void setDuration(double d);
```

```
    public void setFrameRate(double rate);
```

```
    public void setHeight(int h);
```

```
    public void setMetaCue(IMetaCue[] metaCue);
```

```

    public void setVideoCodecId(int id);

    public void setVideoDataRate(int rate);

    public void setWidth(int w);

}

```

## 4.2. getCanSeekToEnd()

```
public boolean getCanSeekToEnd();
```

Returns a boolean depending on whether the video can seek to end

### Parameters

<i>return</i>	true if file is seekable to the end, false otherwise
---------------	--

## 4.3. getDuration()

```
public double getDuration();
```

Returns the duration.

### Parameters

<i>return</i>	duration Video duration in seconds
---------------	------------------------------------

## 4.4. getFrameRate()

```
public double getFrameRate();
```

Returns the framerate.

### Parameters

<i>return</i>	FLV framerate in frames per second
---------------	------------------------------------

## 4.5. getHeight()

```
public int getHeight();
```

Returns the height

### Parameters

<i>return</i>	height Video height
---------------	---------------------

## 4.6. getMetaCue()

```
public IMetaCue[] getMetaCue();
```

Gets the cue points

## Parameters

<i>return</i>	Cue points
---------------	------------

**4.7. getVideoCodecId()**

```
public int getVideoCodecId();
```

Returns the video codec id

## Parameters

<i>return</i>	Video codec id
---------------	----------------

**4.8. getVideoDataRate()**

```
public int getVideoDataRate();
```

Returns the videodatarate

## Parameters

<i>return</i>	Video data rate
---------------	-----------------

**4.9. getWidth()**

```
public int getWidth();
```

Returns the width Video width

## Parameters

<i>return</i>	width
---------------	-------

**4.10. setCanSeekToEnd(boolean)**

```
public void setCanSeekToEnd(boolean b);
```

Sets whether a video can seek to end

## Parameters

b	true if file is seekable to the end, false otherwise
---	--

**4.11. setDuration(double)**

```
public void setDuration(double d);
```

Sets the duration.

## Parameters

d	Video duration in seconds
---	---------------------------

## 4.12. setFrameRate(double)

```
public void setFrameRate(double rate);
```

Sets the framerate.

### Parameters

rate	FLV framerate in frames per second
------	------------------------------------

## 4.13. setHeight(int)

```
public void setHeight(int h);
```

Sets the height

### Parameters

h	Video height
---	--------------

## 4.14. setMetaCue(IMetaCue[])

```
public void setMetaCue(IMetaCue[] metaCue);
```

Sets the cue points

### Parameters

metaCue	Cue points
---------	------------

## 4.15. setVideoCodecId(int)

```
public void setVideoCodecId(int id);
```

Sets the video codec id

### Parameters

id	Video codec id
----	----------------

## 4.16. setVideoDataRate(int)

```
public void setVideoDataRate(int rate);
```

Sets the videodatarate

### Parameters

rate	Video data rate
------	-----------------

## 4.17. setWidth(int)

```
public void setWidth(int w);
```

Sets the width

#### Parameters

w	Video width
---	-------------

## 5. Interface IMetaService

IMetaService Defines the MetaData Service API

### 5.1. Synopsis

```
public interface IMetaService {
// Public Methods

    public IMetaCue[] readMetaCue();

    public MetaData readMetaData(org.apache.mina.common.ByteBuffer buffer);

    public void setInStream(java.io.FileInputStream fis);

    public void setOutStream(java.io.FileOutputStream fos);

    public void write(IMetaData meta)
        throws IOException;

    public void writeMetaCue();

    public void writeMetaData(IMetaData metaData);

}
```

### 5.2. readMetaCue()

```
public IMetaCue[] readMetaCue();
```

Read the Meta Cue Points

#### Parameters

return	Meta cue points
--------	-----------------

### 5.3. readMetaData(ByteBuffer)

```
public MetaData readMetaData(org.apache.mina.common.ByteBuffer buffer);
```

Read the MetaData

#### Parameters

buffer	Byte buffer source
return	metaData Metadata

## 5.4. setInStream(FileInputStream)

```
public void setInStream(java.io.FileInputStream fis);
```

Setter for input stream

Parameters

fis	File input stream
-----	-------------------

## 5.5. setOutStream(FileOutputStream)

```
public void setOutStream(java.io.FileOutputStream fos);
```

Setter for output stream

Parameters

fos	File output stream
-----	--------------------

## 5.6. write(IMetaData)

```
public void write(IMetaData meta)
throws IOException;
```

Initiates writing of the MetaData

Parameters

meta	Metadata
------	----------

IOException

I/O exception

## 5.7. writeMetaCue()

```
public void writeMetaCue();
```

Writes the Meta Cue Points

## 5.8. writeMetaData(IMetaData)

```
public void writeMetaData(IMetaData metaData);
```

Writes the MetaData

Parameters

metaData	Metadata
----------	----------

# 6. Interface IResolver

Merge metadata objects

## 6.1. Synopsis

```
public interface IResolver {
    // Public Methods

    public IMeta resolve(IMeta m1,
                         IMeta m2);

}
```

## 6.2. resolve(IMeta, IMeta)

```
public IMeta resolve(IMeta m1,
                     IMeta m2);
```

Merges the two Meta objects

### Parameters

m1	
m2	
return	IMeta Meta

## 7. Class MetaCue

Cue point is metadata marker used to control and accompany video playback with client-side application events. Each cue point have at least one attribute, timestamp. Timestamp specifies position of cue point in FLV file.

Cue points are usually used as event triggers down video flow or navigation points in a file. Cue points are of two types:

- Embedded into FLV or SWF
- External, or added on fly (e.g. with FLVPlayback component or ActionScript) on both server-side and client-side.

To add cue point trigger event listener at client-side in Flex/Flash application, use NetStream.onCuePoint event handler.

## 7.1. Synopsis

```
public class MetaCue<K,V> extends java.util.HashMap
    implements org.red5.io.flv.meta.IMetaCue {
    // Public Constructors

    public MetaCue();

    // Public Methods

    public int compareTo(Object arg0);

    public String getName();
```

```

    public double getTime();

    public String getType();

    public void setName(String name);

    public void setTime(double d);

    public void setType(String type);

    public String toString();

}

```

**Methods inherited from java.util.HashMap:** clear , clone , containsKey , containsValue , entrySet , get , isEmpty , keySet , put , putAll , remove , size , values

**Methods inherited from java.util.AbstractMap:** equals , hashCode , toString

**Methods inherited from java.lang.Object:** finalize , getClass , notify , notifyAll , wait

## 7.2. MetaCue()

```
public MetaCue();
```

CuePoint constructor

## 7.3. compareTo(Object)

```
public int compareTo(Object arg0);
```

## 7.4. getName()

```
public String getName();
```

**Specified by:** Method getName in interface IMetaCue

Gets the name

## 7.5. getTime()

```
public double getTime();
```

**Specified by:** Method getTime in interface IMetaCue

Gets the time

## 7.6. getType()

```
public String getType();
```

**Specified by:** Method getType in interface IMetaCue

Gets the type

## 7.7. setName(String)

```
public void setName(String name);
```

**Specified by:** Method setName in interface IMetaCue

Sets the name

## 7.8. setTime(double)

```
public void setTime(double d);
```

**Specified by:** Method setTime in interface IMetaCue

Sets the time

## 7.9. setType(String)

```
public void setType(String type);
```

**Specified by:** Method setType in interface IMetaCue

Sets the type type can be "event" or "navigation"

## 7.10. toString()

```
public String toString();
```

# 8. Class MetaData

MetaData Implementation

## 8.1. Synopsis

```
public class MetaData<K,V> extends, java.util.HashMap
    implements, org.red5.io.flv.meta.IMetaData {
// Public Constructors

    public MetaData();

// Public Methods

    public int getAudioCodecId();

    public boolean getCanSeekToEnd();

    public double getDuration();

    public double getFrameRate();

    public int getHeight();

    public IMetaCue[] getMetaCue();

    public int getVideoCodecId();
```

```

    public int getVideoDataRate();

    public int getWidth();

    public void setAudioCodecId(int id);

    public void setCanSeekToEnd(boolean b);

    public void setDuration(double d);

    public void setFrameRate(double rate);

    public void setHeight(int h);

    public void setMetaCue(IMetaCue[] cuePoints);

    public void setVideoCodecId(int id);

    public void setVideoDataRate(int rate);

    public void setWidth(int w);

    public String toString();

}

```

**Methods inherited from java.util.HashMap:** clear, clone, containsKey, containsValue, entrySet, get, isEmpty, keySet, put, putAll, remove, size, values

**Methods inherited from java.util.AbstractMap:** equals, hashCode, toString

**Methods inherited from java.lang.Object:** finalize, getClass, notify, notifyAll, wait

## 8.2. MetaData()

```
public MetaData();
```

MetaData constructor

## 8.3. getCanSeekToEnd()

```
public boolean getCanSeekToEnd();
```

**Specified by:** Method getCanSeekToEnd in interface IMetaData

Returns a boolean depending on whether the video can seek to end

## 8.4. getDuration()

```
public double getDuration();
```

**Specified by:** Method getDuration in interface IMetaData

Returns the duration.

## 8.5. getFrameRate()

```
public double getFrameRate();
```

**Specified by:** Method getFrameRate in interface IMetaData

Returns the framerate.

## 8.6. getHeight()

```
public int getHeight();
```

**Specified by:** Method getHeight in interface IMetaData

Returns the height

## 8.7. getMetaCue()

```
public IMetaCue[] getMetaCue();
```

**Specified by:** Method getMetaCue in interface IMetaData

Return array of cue points

### Parameters

<i>return</i>	Array of cue points
---------------	---------------------

## 8.8. getVideoCodecId()

```
public int getVideoCodecId();
```

**Specified by:** Method getVideoCodecId in interface IMetaData

Returns the video codec id

## 8.9. getVideoDataRate()

```
public int getVideoDataRate();
```

**Specified by:** Method getVideoDataRate in interface IMetaData

Returns the videodatarate

## 8.10. getWidth()

```
public int getWidth();
```

**Specified by:** Method getWidth in interface IMetaData

Returns the width Video width

## 8.11. setCanSeekToEnd(boolean)

```
public void setCanSeekToEnd(boolean b);
```

**Specified by:** Method setCanSeekToEnd in interface IMetaData

Sets whether a video can seek to end

## 8.12. setDuration(double)

```
public void setDuration(double d);
```

**Specified by:** Method setDuration in interface IMetaData

Sets the duration.

## 8.13. setFrameRate(double)

```
public void setFrameRate(double rate);
```

**Specified by:** Method setFrameRate in interface IMetaData

Sets the framerate.

## 8.14. setHeight(int)

```
public void setHeight(int h);
```

**Specified by:** Method setHeight in interface IMetaData

Sets the height

## 8.15. setMetaCue(IMetaCue[])

```
public void setMetaCue(IMetaCue[] cuePoints);
```

**Specified by:** Method setMetaCue in interface IMetaData

Sets the Meta Cue Points

### Parameters

cuePoints	The cuePoints to set.
-----------	-----------------------

## 8.16. setVideoCodecId(int)

```
public void setVideoCodecId(int id);
```

**Specified by:** Method setVideoCodecId in interface IMetaData

Sets the video codec id

## 8.17. setVideoDataRate(int)

```
public void setVideoDataRate(int rate);
```

**Specified by:** Method setVideoDataRate in interface IMetaData

Sets the videodatarate

## 8.18. setWidth(int)

```
public void setWidth(int w);
```

**Specified by:** Method setWidth in interface IMetaData

Sets the width

## 8.19. toString()

```
public String toString();
```

# 9. Class MetaService

MetaService represents a MetaData service in Spring

## 9.1. Synopsis

```
public class MetaService implements org.?red5.?io.?flv.?meta.?IMetaService {
// Public Constructors

    public MetaService();

    public MetaService(java.io.File poFil);

// Public Methods

    public org.red5.io.object.Deserializer getDeserializer();

    public java.io.File getFile();

    public Resolver getResolver();

    public org.red5.io.object.Serializer getSerializer();

    public IMetaCue[] readMetaCue();

    public MetaData readMetaData(org.apache.mina.common.ByteBuffer buffer);

    public void setDeserializer(org.red5.io.object.Deserializer deserializer);

    public void setFile(java.io.File file);

    public void setInStream(java.io.FileInputStream fis);

    public void setOutStream(java.io.FileOutputStream fos);

    public void setResolver(Resolver resolver);

    public void setSerializer(org.red5.io.object.Serializer serializer);

    public void write(IMetaData meta)
        throws IOException;

    public void writeMetaCue();
}
```

```
public void writeMetaData(IMetaData metaData);  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 9.2. MetaService()

```
public MetaService();
```

MetaService constructor

## 9.3. getDeserializer()

```
public org.red5.io.object.Deserializer getDeserializer();
```

### Parameters

return	Returns the deserializer.
--------	---------------------------

## 9.4. getFile()

```
public java.io.File getFile();
```

### Parameters

return	Returns the file.
--------	-------------------

## 9.5. getResolver()

```
public Resolver getResolver();
```

### Parameters

return	Returns the resolver.
--------	-----------------------

## 9.6. getSerializer()

```
public org.red5.io.object.Serializer getSerializer();
```

### Parameters

return	Returns the serializer.
--------	-------------------------

## 9.7. readMetaCue()

```
public IMetaCue[] readMetaCue();
```

**Specified by:** Method readMetaCue in interface IMetaService

Read the Meta Cue Points

## 9.8. readMetaData(ByteBuffer)

```
public MetaData readMetaData(org.apache.mina.common.ByteBuffer buffer);
```

**Specified by:** Method readMetaData in interface IMetaService

Read the MetaData

## 9.9. setDeserializer(Deserializer)

```
public void setDeserializer(org.red5.io.object.Deserializer deserializer);
```

Parameters

deserializer	The deserializer to set.
--------------	--------------------------

## 9.10. setFile(File)

```
public void setFile(java.io.File file);
```

Parameters

file	The file to set.
------	------------------

## 9.11. setInStream(FileInputStream)

```
public void setInStream(java.io.FileInputStream fis);
```

**Specified by:** Method setInStream in interface IMetaService

Setter for input stream

## 9.12. setOutStream(OutputStream)

```
public void setOutStream(java.io.OutputStream fos);
```

**Specified by:** Method setOutStream in interface IMetaService

Setter for output stream

## 9.13. setResolver(Resolver)

```
public void setResolver(Resolver resolver);
```

Parameters

resolver	The resolver to set.
----------	----------------------

## 9.14. setSerializer(Serializer)

```
public void setSerializer(org.red5.io.object.Serializer serializer);
```

**Parameters**

serializer	The serializer to set.
------------	------------------------

**9.15. write(IMetaData)**

```
public void write(IMetaData meta)
    throws IOException;
```

**Specified by:** Method write in interface IMetaService

Initiates writing of the MetaData

**9.16. writeMetaCue()**

```
public void writeMetaCue();
```

**Specified by:** Method writeMetaCue in interface IMetaService

Writes the Meta Cue Points

**9.17. writeMetaData(IMetaData)**

```
public void writeMetaData(IMetaData metaData);
```

**Specified by:** Method writeMetaData in interface IMetaService

Writes the MetaData

**10. Class Resolver**

Metadata resolver implementation, merges metadata objects

**10.1. Synopsis**

```
public class Resolver implements org.?red5.?io.?flv.?meta.?IResolver {
    // Public Constructors

    public Resolver();

    // Public Methods

    public IMeta resolve(IMeta m1,
        IMeta m2);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

**10.2. Resolver()**

```
public Resolver();
```

Creates resolver object

### 10.3. resolve(IMeta, IMeta)

```
public IMeta resolve(IMeta m1,  
                     IMeta m2);
```

**Specified by:** Method resolve in interface IResolver

Merges the two Meta objects

# 1. Class Input

```
public class Input extends org.red5.io.object.BaseInput
    implements org.red5.io.object.Input {
// Protected Fields

    protected int idx ;

    protected java.util.List<java.lang.Object> list ;

    protected static org.slf4j.Logger log ;

// Public Constructors

    public Input(java.util.List<java.lang.Object> list);

// Public Methods

    public String getString();

    public Object readArray(org.red5.io.object.Deserializer deserializer);

    public Boolean readBoolean();

    public org.red5.io.amf3.ByteArray readByteArray();

    public Object readCustom();

    public byte readDataType();

    public java.util.Date readDate();

    public java.util.Map<java.lang.String, java.lang.Object> readKeyValues(org.red5.io.object.Deserializer deserializer);

    public Object readMap(org.red5.io.object.Deserializer deserializer);

    public Object readNull();

    public Number readNumber();

    public Object readObject(org.red5.io.object.Deserializer deserializer);

    public Object readReference();

    public String readString();

    public org.w3c.dom.Document readXML();

// Protected Methods

    protected Object getNext();

}
```

**Methods inherited from org.red5.io.object.BaseInput:** clearReferences , getReference , storeReference

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.io.object.BaseInput:** referenceMode , refId , refMap

## 1.1. getNext()

```
protected Object getNext();
```

Getter for property 'next'.

### Parameters

<i>return</i>	Value for property 'next'.
---------------	----------------------------

## 1.2. getString()

```
public String getString();
```

**Specified by:** Method getString in interface Input

Read a string without the string type header.

## 1.3. readArray(Deserializer)

```
public Object readArray(org.red5.io.object.Deserializer deserializer);
```

**Specified by:** Method readArray in interface Input

Read an array. This can result in a List or Map being deserialized depending on the array type found.

## 1.4. readBoolean()

```
public Boolean readBoolean();
```

**Specified by:** Method readBoolean in interface Input

Read Boolean value

## 1.5. readByteArray()

```
public org.red5.io.amf3.ByteArray readByteArray();
```

**Specified by:** Method readByteArray in interface Input

Read ByteArray object.

## 1.6. readCustom()

```
public Object readCustom();
```

**Specified by:** Method readCustom in interface Input

Read custom object

## 1.7. **readDataType()**

```
public byte readDataType();
```

**Specified by:** Method readDataType in interface Input

Read type of data

## 1.8. **readDate()**

```
public java.util.Date readDate();
```

**Specified by:** Method readDate in interface Input

Read date object

## 1.9. **readKeyValues(Deserializer)**

```
public java.util.Map<java.lang.String, java.lang.Object> readKeyValues(org.red5.io.object.Deserializer);
```

**Specified by:** Method readKeyValues in interface Input

Read key - value pairs. This is required for the RecordSet deserializer.

## 1.10. **readMap(Deserializer)**

```
public Object readMap(org.red5.io.object.Deserializer deserializer);
```

**Specified by:** Method readMap in interface Input

Read a map containing key - value pairs. This can result in a List or Map being deserialized depending on the map type found.

## 1.11. **readNull()**

```
public Object readNull();
```

**Specified by:** Method readNull in interface Input

Read Null data type

## 1.12. **readNumber()**

```
public Number readNumber();
```

**Specified by:** Method readNumber in interface Input

Read Number object

## 1.13. **readObject(Deserializer)**

```
public Object readObject(org.red5.io.object.Deserializer deserializer);
```

**Specified by:** Method readObject in interface Input

Read an object.

## 1.14. readReference()

```
public Object readReference();
```

**Specified by:** Method readReference in interface Input

Read reference to Complex Data Type. Objects that are collaborators (properties) of other objects must be stored as references in map of id-reference pairs.

## 1.15. readString()

```
public String readString();
```

**Specified by:** Method readString in interface Input

Read String object

## 1.16. readXML()

```
public org.w3c.dom.Document readXML();
```

**Specified by:** Method readXML in interface Input

Read XML document

## 2. Class Mock

```
public class Mock {
    // Public Static Fields

    public static final byte TYPE_ELEMENT_SEPARATOR = 35;
    public static final byte TYPE_END_OF_ARRAY = 34;
    public static final byte TYPE_END_OF_MAP = 38;
    public static final byte TYPE_END_OF_OBJECT = 33;
    public static final byte TYPE_ITEM_SEPARATOR = 37;
    public static final byte TYPE_PROPERTY_SEPARATOR = 36;

    // Public Constructors

    public Mock();
    // Public Static Methods

    public static String listToString(java.util.List<java.lang.Object> list);
    public static String toStringValue(byte dataType);
```

```
}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

### 3. Class Output

```
public class Output extends, org.?red5.?io.?object.?BaseOutput
    implements, org.?red5.?io.?object.?Output {
// Protected Fields

    protected java.util.List<java.lang.Object> list ;

    protected static org.slf4j.Logger log ;

// Public Constructors

    public Output(java.util.List<java.lang.Object> list);

// Public Methods

    public boolean isCustom(Object custom);

    public void putString(String string);

    public boolean supportsDataType(byte type);

    public void writeArray(Object array,
                          org.red5.io.object.Serializer serializer);

    public void writeArray(Object[] array,
                          org.red5.io.object.Serializer serializer);

    public void writeArray(java.util.Collection<?> array,
                          org.red5.io.object.Serializer serializer);

    public void writeBoolean(Boolean bol);

    public void writeByteArray(org.red5.io.amf3.ByteArray array);

    public void writeCustom(Object custom);

    public void writeDate(java.util.Date date);

    public void writeMap(java.util.Collection<?> array,
                       org.red5.io.object.Serializer serializer);

    public void writeMap(java.util.Map<java.lang.Object, java.lang.Object> map,
                       org.red5.io.object.Serializer serializer);

    public void writeNull();

    public void writeNumber(Number num);

    public void writeObject(Object object,
```

```

        org.red5.io.object.Serializer serializer);

public void writeObject(java.util.Map<java.lang.Object, java.lang.Object> map,
                      org.red5.io.object.Serializer serializer);

public void writeRecordSet(org.red5.io.object.RecordSet recordset,
                         org.red5.io.object.Serializer serializer);

public void writeReference(Object obj);

public void writeString(String string);

public void writeXML(org.w3c.dom.Document xml);

}

```

**Methods inherited from org.red5.io.object.BaseOutput:** clearReferences ,  
getReferenceId , hasReference , storeReference

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.io.object.BaseOutput:** refId , refMap

### 3.1. isCustom(Object)

```
public boolean isCustom(Object custom);
```

**Specified by:** Method isCustom in interface Output

Whether object is custom

### 3.2. putString(String)

```
public void putString(String string);
```

**Specified by:** Method putString in interface Output

### 3.3. supportsDataType(byte)

```
public boolean supportsDataType(byte type);
```

**Specified by:** Method supportsDataType in interface Output

### 3.4. writeArray(Collection<?>, Serializer)

```
public void writeArray(java.util.Collection<?> array,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeArray in interface Output

Write array.

### 3.5. writeArray(Object[], Serializer)

```
public void writeArray(Object[] array,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeArray in interface Output

Write array.

### 3.6. writeArray(Object, Serializer)

```
public void writeArray(Object array,
                      org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeArray in interface Output

Write primitive array.

### 3.7. writeBoolean(Boolean)

```
public void writeBoolean(Boolean bol);
```

**Specified by:** Method writeBoolean in interface Output

Write boolean

### 3.8. writeByteArray(ByteArray)

```
public void writeByteArray(org.red5.io.amf3.ByteArray array);
```

**Specified by:** Method writeByteArray in interface Output

Write ByteArray object (AMF3 only).

### 3.9. writeCustom(Object)

```
public void writeCustom(Object custom);
```

**Specified by:** Method writeCustom in interface Output

Write custom (user) object

### 3.10. writeDate(Date)

```
public void writeDate(java.util.Date date);
```

**Specified by:** Method writeDate in interface Output

Write date

### 3.11. writeMap(Collection<?>, Serializer)

```
public void writeMap(java.util.Collection<?> array,
                     org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeMap in interface Output

Write array as map.

### 3.12. writeMap(Map<Object, Object>, Serializer)

```
public void writeMap(java.util.Map<java.lang.Object, java.lang.Object> map,
                     org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeMap in interface Output

Write map.

### 3.13. writeNull()

```
public void writeNull();
```

**Specified by:** Method writeNull in interface Output

### 3.14. writeNumber(Number)

```
public void writeNumber(Number num);
```

**Specified by:** Method writeNumber in interface Output

Write number

### 3.15. writeObject(Map<Object, Object>, Serializer)

```
public void writeObject(java.util.Map<java.lang.Object, java.lang.Object> map,
                       org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeObject in interface Output

Write map as object.

### 3.16. writeObject(Object, Serializer)

```
public void writeObject(Object object,
                       org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeObject in interface Output

Write object.

### 3.17. writeRecordSet(RecordSet, Serializer)

```
public void writeRecordSet(org.red5.io.object.RecordSet recordset,
                         org.red5.io.object.Serializer serializer);
```

**Specified by:** Method writeRecordSet in interface Output

Write recordset.

### 3.18. writeReference(Object)

```
public void writeReference(Object obj);
```

**Specified by:** Method writeReference in interface Output

Write reference to complex data type

### 3.19. **writeString(String)**

```
public void writeString(String string);
```

**Specified by:** Method writeString in interface Output

Write string

### 3.20. **writeXML(Document)**

```
public void writeXML(org.w3c.dom.Document xml);
```

**Specified by:** Method writeXML in interface Output

Write XML object

# 1. Interface IMP3

Informations about a MP3 file. Marker interface by now.

## 1.1. Synopsis

```
public interface IMP3 extends, org.?red5.?io.?IStreamableFile {  
}
```

# 2. Interface IMP3Service

Provide access to MP3 objects. Marker interface.

## 2.1. Synopsis

```
public interface IMP3Service extends, org.?red5.?io.?IStreamableFileService {  
}
```

# 1. Class MP3

Represents MP3 file

## 1.1. Synopsis

```
public class MP3 implements, org.red5.io.mp3.IMP3 {  
    // Public Constructors  
  
    public MP3(java.io.File file);  
  
    // Public Methods  
  
    public org.red5.io.ITagWriter getAppendWriter()  
        throws IOException;  
  
    public org.red5.io.ITagReader getReader()  
        throws IOException;  
  
    public org.red5.io.ITagWriter getWriter()  
        throws IOException;  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. MP3(File)

```
public MP3(java.io.File file);
```

Creates MP3 object using given file

### Parameters

file	File object to use
------	--------------------

## 1.3. getAppendWriter()

```
public org.red5.io.ITagWriter getAppendWriter()  
    throws IOException;
```

## 1.4. getReader()

```
public org.red5.io.ITagReader getReader()  
    throws IOException;
```

## 1.5. getWriter()

```
public org.red5.io.ITagWriter getWriter()  
    throws IOException;
```

# 2. Class MP3Header

Header of a MP3 frame.

## 2.1. Synopsis

```
public class MP3Header {
// Public Constructors

    public MP3Header(int data)
        throws Exception;

// Public Methods

    public double frameDuration();

    public int frameSize();

    public int getBitRate();

    public int getData();

    public int getSampleRate();

    public boolean isProtected();

    public boolean isStereo();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

*See Also*

File format [[http://mpgedit.org/mpgedit/mpeg\\_format/mpeghdr.htm](http://mpgedit.org/mpgedit/mpeg_format/mpeghdr.htm)]

## 2.2. MP3Header(int)

```
public MP3Header(int data)
    throws Exception;
```

Creates MP3 header from frame sync value

### Parameters

data	Frame sync data
------	-----------------

Exception

On invalid frame synchronization

## 2.3. frameDuration()

```
public double frameDuration();
```

Return the duration of the frame for this header.

**Parameters**

<i>return</i>	The duration in milliseconds
---------------	------------------------------

**2.4. frameSize()**

public int frameSize();
-------------------------

Calculate the size of a MP3 frame for this header.

**Parameters**

<i>return</i>	size of the frame including the header
---------------	--

**2.5. getBitRate()**

public int getBitRate();
--------------------------

Getter for bitrate

**Parameters**

<i>return</i>	File bitrate
---------------	--------------

**2.6. getData()**

public int getData();
-----------------------

Getter for frame sync word data

**Parameters**

<i>return</i>	Frame sync word data
---------------	----------------------

**2.7. getSampleRate()**

public int getSampleRate();
-----------------------------

Getter for sample rate

**Parameters**

<i>return</i>	Sampling rate
---------------	---------------

**2.8. isProtected()**

public boolean isProtected();
-------------------------------

Whether MP3 has protection bit

**Parameters**

`return``true if MP3 has protection bit, false otherwise`

## 2.9. isStereo()

```
public boolean isStereo();
```

Whether stereo playback mode is used

### Parameters

`return``true if stereo mode is used, false otherwise`

## 3. Class MP3Reader

Read MP3 files

### 3.1. Synopsis

```
public class MP3Reader implements org.red5.io.ITagReader, org.red5.io.flv.IKeyFrameDataAnalyzer
// Protected Fields
protected static org.slf4j.Logger log ;
// Public Constructors
public MP3Reader(java.io.File file)
throws FileNotFoundException;
// Public Methods
public synchronized org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta analyzeKeyFrames();
public void close();
public void decodeHeader();
public long getBytesRead();
public long getDuration();
public org.red5.io.IStreamableFile getFile();
public int getOffset();
public long getTotalBytes();
public boolean hasMoreTags();
public boolean hasVideo();
public void position(long pos);
public synchronized org.red5.io.ITag readTag();
public void searchNextFrame();
```

```
public void setFrameCache(org.red5.io.IKeyFrameMetaCache frameCache);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 3.2. MP3Reader(File)

```
public MP3Reader(java.io.File file)
    throws FileNotFoundException;
```

Creates reader from file input stream

### Parameters

stream	File input stream source
--------	--------------------------

## 3.3. log

```
protected static org.slf4j.Logger log;
```

Logger

## 3.4. analyzeKeyFrames()

```
public synchronized org.red5.io.flv.IKeyFrameDataAnalyzer.KeyFrameMeta analyzeKeyFrames();
```

**Specified by:** Method analyzeKeyFrames in interface IKeyFrameDataAnalyzer

Analyze and return keyframe metadata.

## 3.5. close()

```
public void close();
```

**Specified by:** Method close in interface ITagReader

Closes the reader and free any allocated memory.

## 3.6. decodeHeader()

```
public void decodeHeader();
```

**Specified by:** Method decodeHeader in interface ITagReader

Decode the header of the stream;

## 3.7. getBytesRead()

```
public long getBytesRead();
```

**Specified by:** Method getBytesRead in interface ITagReader

Returns the amount of bytes read

### 3.8. getDuration()

```
public long getDuration();
```

**Specified by:** Method getDuration in interface ITagReader

Return length in seconds

### 3.9. getFile()

```
public org.red5.io.IStreamableFile getFile();
```

**Specified by:** Method getFile in interface ITagReader

Return the file that is loaded.

### 3.10. getOffset()

```
public int getOffset();
```

**Specified by:** Method getOffset in interface ITagReader

Returns the offset length

### 3.11. getTotalBytes()

```
public long getTotalBytes();
```

**Specified by:** Method getTotalBytes in interface ITagReader

Get the total readable bytes in a file or ByteBuffer.

#### Parameters

return	Total readable bytes
--------	----------------------

### 3.12. hasMoreTags()

```
public boolean hasMoreTags();
```

**Specified by:** Method hasMoreTags in interface ITagReader

Returns a boolean stating whether the FLV has more tags

### 3.13. hasVideo()

```
public boolean hasVideo();
```

**Specified by:** Method hasVideo in interface ITagReader

A MP3 stream never has video.

**Parameters**

<i>return</i>	always returns false
---------------	----------------------

**3.14. position(long)**

```
public void position(long pos);
```

**Specified by:** Method position in interface ITagReader

Move the reader pointer to given position in file.

**3.15. readTag()**

```
public synchronized org.red5.io.ITag readTag();
```

**Specified by:** Method readTag in interface ITagReader

Returns a Tag object

**3.16. searchNextFrame()**

```
public void searchNextFrame();
```

Search for next frame sync word. Sync word identifies valid frame.

**4. Class MP3Service**

Streamable file service extension for MP3

**4.1. Synopsis**

```
public class MP3Service extends org.red5.io.BaseStreamableFileService
    implements org.red5.io.mp3.IMP3Service {
    // Public Constructors

    public MP3Service();

    // Public Methods

    public String getExtension();

    public String getPrefix();

    public org.red5.io.IStreamableFile getStreamableFile(java.io.File file)
        throws IOException;

}
```

**Methods inherited from org.red5.io.BaseStreamableFileService:** canHandle ,  
getExtension , getPrefix , getStreamableFile , prepareFilename

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 4.2. getExtension()

```
public String getExtension();
```

## 4.3. getPrefix()

```
public String getPrefix();
```

## 4.4. getStreamableFile(File)

```
public org.red5.io.IStreamableFile getStreamableFile(java.io.File file)
throws IOException;
```

# 1. Class BaseInput

BaseInput represents a way to map input to a HashMap. This class is meant to be extended.

## 1.1. Synopsis

```
public class BaseInput {  
    // Protected Fields  
  
    protected int refId ;  
  
    protected java.util.Map<java.lang.Integer, java.lang.Object> refMap ;  
  
    protected org.red5.io.object.BaseInput.ReferenceMode referenceMode ;  
  
    // Public Constructors  
  
    public BaseInput();  
  
    // Public Methods  
  
    public void clearReferences();  
  
    // Protected Methods  
  
    protected Object getReference(int id);  
  
    protected void storeReference(int refId,  
                                Object newRef);  
  
    protected int storeReference(Object obj);  
}
```

**Direct known subclasses:** org.red5.io.amf.Input , org.red5.io.mock.Input

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 1.2. referenceMode

```
protected org.red5.io.object.BaseInput.ReferenceMode referenceMode ;
```

Mode how to handle references.

## 1.3. refId

```
protected int refId ;
```

References id

## 1.4. refMap

```
protected java.util.Map<java.lang.Integer, java.lang.Object> refMap ;
```

References map

## 1.5. clearReferences()

```
public void clearReferences();
```

Clears the map

## 1.6. getReference(int)

```
protected Object getReference(int id);
```

Returns the object with the parameters id

### Parameters

id	Object reference id
<i>return</i>	Object Object reference with given id

## 1.7. storeReference(int, Object)

```
protected void storeReference(int refId,  
                             Object newRef);
```

Replace a referenced object with another one. This is used by the AMF3 deserializer to handle circular references.

### Parameters

refId	
<i>newRef</i>	

## 1.8. storeReference(Object)

```
protected int storeReference(Object obj);
```

Store an object into a map

### Parameters

obj	Object to store
-----	-----------------

## 2. Class BaseInput.ReferenceMode

Mode how references should be handled.

### 2.1. Synopsis

```
public static final class BaseInput.ReferenceMode extends java.lang.Enum {
```

```
// Public Static Fields

    public static final org.red5.io.object.BaseInput.ReferenceMode MODE_REMOTING ;

    public static final org.red5.io.object.BaseInput.ReferenceMode MODE_RTMP ;

// Public Static Methods

    public static org.red5.io.object.BaseInput.ReferenceMode valueOf(String name);

    public static org.red5.io.object.BaseInput.ReferenceMode[] values();

}
```

**Methods inherited from java.lang.Enum:** clone, compareTo, equals, finalize, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

**Methods inherited from java.lang.Object:** getClass, notify, notifyAll, wait

## 3. Class BaseOutput

BaseOutput represents a way to map input to a HashMap. This class is meant to be extended.

### 3.1. Synopsis

```
public class BaseOutput {
// Protected Fields

    protected short refId ;

    protected java.util.Map<org.red5.io.object.BaseOutput.IdentityWrapper, java.lang.Short> refMap ;

// Protected Constructors

    protected BaseOutput();

// Public Methods

    public void clearReferences();

// Protected Methods

    protected short getReferenceId(Object obj);

    protected boolean hasReference(Object obj);

    protected void storeReference(Object obj);

}
```

**Direct known subclasses:** org.?red5.?io.?amf.?Output , org.?red5.?io.?mock.?Output

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 3.2. BaseOutput()

```
protected BaseOutput();
```

BaseOutput Constructor

### 3.3. refId

```
protected short refId ;
```

Reference id

### 3.4. refMap

```
protected java.util.Map<org.red5.io.object.BaseOutput.IdentityWrapper, java.lang.Short> refMap ;
```

References map

### 3.5. clearReferences()

```
public void clearReferences();
```

Clears the map

### 3.6. getReferenceId(Object)

```
protected short getReferenceId(Object obj);
```

Returns the reference id based on the parameter obj

#### Parameters

obj	Object
-----	--------

return	short Reference id
--------	--------------------

### 3.7. hasReference(Object)

```
protected boolean hasReference(Object obj);
```

Returns a boolean stating whether the map contains an object with that key

#### Parameters

obj	Object
-----	--------

return	boolean true if it does contain it, false otherwise
--------	---

### 3.8. storeReference(Object)

```
protected void storeReference(Object obj);
```

Store an object into a map

#### Parameters

obj	Object to store
-----	-----------------

## 4. Class DataTypes

The core datatypes supported by red5, I have left out undefined (this is up for debate). If a codec returns one of these datatypes its handled by the base serializer.

### 4.1. Synopsis

```
public class DataTypes {
// Public Static Fields

    public static final byte CORE_ARRAY = 6;

    public static final byte CORE_BOOLEAN = 2;

    public static final byte CORE_BYTEARRAY = 16;

    public static final byte CORE_DATE = 5;

    public static final byte CORE_MAP = 7;

    public static final byte CORE_NULL = 1;

    public static final byte CORE_NUMBER = 3;

    public static final byte CORE_OBJECT = 9;

    public static final byte CORE_SKIP = 0;

    public static final byte CORE_STRING = 4;

    public static final byte CORE_XML = 8;

    public static final byte CUSTOM_AMF_MASK = 48;

    public static final byte CUSTOM_JSON_MASK = 80;

    public static final byte CUSTOM_MOCK_MASK = 32;

    public static final byte CUSTOM_RTMP_MASK = 64;

    public static final byte CUSTOM_XML_MASK = 96;

    public static final byte OPT_REFERENCE = 17;

// Public Constructors

    public DataTypes();

// Public Static Methods
```

```

    public static boolean isBasicType(byte type);

    public static boolean isComplexType(byte type);

    public static boolean isCustomType(byte type);

    public static String toStringValue(byte dataType);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 4.2. CORE\_ARRAY

```
public static final byte CORE_ARRAY = 6;
```

Array type marker

## 4.3. CORE\_BOOLEAN

```
public static final byte CORE_BOOLEAN = 2;
```

Boolean type marker

## 4.4. CORE\_BYTEARRAY

```
public static final byte CORE_BYTEARRAY = 16;
```

ByteArray type marker (AMF3 only)

## 4.5. CORE\_DATE

```
public static final byte CORE_DATE = 5;
```

Date type marker

## 4.6. CORE\_MAP

```
public static final byte CORE_MAP = 7;
```

Map type marker

## 4.7. CORE\_NULL

```
public static final byte CORE_NULL = 1;
```

Null type marker

## 4.8. CORE\_NUMBER

```
public static final byte CORE_NUMBER = 3;
```

Number type marker

## 4.9. CORE\_OBJECT

```
public static final byte CORE_OBJECT = 9;
```

Object (Hash) type marker

## 4.10. CORE\_SKIP

```
public static final byte CORE_SKIP = 0;
```

Padding marker

## 4.11. CORE\_STRING

```
public static final byte CORE_STRING = 4;
```

String type marker

## 4.12. CORE\_XML

```
public static final byte CORE_XML = 8;
```

XML type marker

## 4.13. CUSTOM\_AMF\_MASK

```
public static final byte CUSTOM_AMF_MASK = 48;
```

Custom datatype AMF mask

## 4.14. CUSTOM\_JSON\_MASK

```
public static final byte CUSTOM_JSON_MASK = 80;
```

Custom datatype JSON mask

## 4.15. CUSTOM\_MOCK\_MASK

```
public static final byte CUSTOM_MOCK_MASK = 32;
```

Custom datatype mock mask marker

## 4.16. CUSTOM\_RTMP\_MASK

```
public static final byte CUSTOM_RTMP_MASK = 64;
```

Custom datatype RTMP mask

## 4.17. CUSTOM\_XML\_MASK

```
public static final byte CUSTOM_XML_MASK = 96;
```

Custom datatype XML mask

## 4.18. OPT\_REFERENCE

```
public static final byte OPT_REFERENCE = 17;
```

Reference type, this is optional for codecs to support

## 4.19. isBasicType(byte)

```
public static boolean isBasicType(byte type);
```

Returns whether it is a basic data type

### Parameters

<i>type</i>	Data type as byte
<i>return</i>	boolean <code>true</code> if data type is primitive, <code>false</code> otherwise

## 4.20. isComplexType(byte)

```
public static boolean isComplexType(byte type);
```

Returns whether it is a complex data type

### Parameters

<i>type</i>	Data type as byte
<i>return</i>	boolean <code>true</code> if data type is complex (non-primitive), <code>false</code> otherwise

## 4.21. isCustomType(byte)

```
public static boolean isCustomType(byte type);
```

Returns whether it is a custom data type

### Parameters

<i>type</i>	Data type as byte
<i>return</i>	boolean Whether given type is of custom type

## 4.22. toStringValue(byte)

```
public static String toStringValue(byte dataType);
```

Returns the string value of the data type

### Parameters

dataType	AS data type as byte
<i>return</i>	String String value of given ActionScript data type

## 5. Class Deserializer

The Deserializer class reads data input and handles the data according to the core data types

### 5.1. Synopsis

```
public class Deserializer {
    // Protected Fields

    protected static org.slf4j.Logger log ;

    // Public Constructors

    public Deserializer();

    // Public Methods

    public T deserialize(Input in,
                         Class<T> target);

    // Protected Methods

    protected Object postProcessExtension(Object result,
                                          Class target);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

### 5.2. deserialize(Input, Class<T>)

```
public T deserialize(Input in,
                     Class<T> target);
```

Deserializes the input parameter and returns an Object which must then be cast to a core data type

Parameters	
in	
target	
<i>return</i>	Object

### 5.3. postProcessExtension(Object, Class)

```
protected Object postProcessExtension(Object result,
```

```
Class target);
```

Post processes the result TODO Extension Point

#### Parameters

result	
--------	--

target	
--------	--

<i>return</i>	
---------------	--

## 6. Class Flag

Serial flag options

### 6.1. Synopsis

```
public final class Flag extends java.lang.Enum {
// Public Static Fields

    public static final Flag Default ;

    public static final Flag Disabled ;

    public static final Flag Enabled ;

// Public Static Methods

    public static Flag valueOf(String name);

    public static Flag[] values();

}
```

**Methods inherited from java.lang.Enum:** clone, compareTo, equals, finalize, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

**Methods inherited from java.lang.Object:** getClass, notify, notifyAll, wait

## 7. Interface ICustomSerializable

Interface for objects that know how to serialize their contents. NOTE: This is only used for AMF0 encoding and you should not need to implement this in your own objects.

### 7.1. Synopsis

```
public interface ICustomSerializable {
// Public Methods

    public void serialize(Output output,
                         Serializer serializer);

}
```

## 7.2. serialize(Output, Serializer)

```
public void serialize(Output output,
                      Serializer serializer);
```

Serialize this object to the given output stream.

### Parameters

output	
--------	--

serializer	
------------	--

## 8. Interface Input

Interface for Input which defines the contract methods which are to be implemented. Input object provides ways to read primitives, complex object and object references from byte buffer.

### 8.1. Synopsis

```
public interface Input {
    // Public Methods

    public void clearReferences();

    public String getString();

    public Object readArray(Deserializer deserializer);

    public Boolean readBoolean();

    public org.red5.io.amf3.ByteArray readByteArray();

    public Object readCustom();

    public byte readDataType();

    public java.util.Date readDate();

    public java.util.Map<java.lang.String, java.lang.Object> readKeyValues(Deserializer deserializer);

    public Object readMap(Deserializer deserializer);

    public Object readNull();

    public Number readNumber();

    public Object readObject(Deserializer deserializer);

    public Object readReference();

    public String readString();
```

```
public org.w3c.dom.Document readXML();
}
```

## 8.2. clearReferences()

```
public void clearReferences();
```

Clears all references

## 8.3. getString()

```
public String getString();
```

Read a string without the string type header.

### Parameters

<i>return</i>	String
---------------	--------

## 8.4. readArray(Deserializer)

```
public Object readArray(Deserializer deserializer);
```

Read an array. This can result in a List or Map being deserialized depending on the array type found.

### Parameters

<i>return</i>	array
---------------	-------

## 8.5. readBoolean()

```
public Boolean readBoolean();
```

Read Boolean value

### Parameters

<i>return</i>	Boolean
---------------	---------

## 8.6. readByteArray()

```
public org.red5.io.amf3.ByteArray readByteArray();
```

Read ByteArray object.

### Parameters

<i>return</i>	ByteArray object
---------------	------------------

## 8.7. readCustom()

```
public Object readCustom();
```

Read custom object

### Parameters

<i>return</i>	Custom object
---------------	---------------

## 8.8. readDataType()

```
public byte readDataType();
```

Read type of data

### Parameters

<i>return</i>	Type of data as byte
---------------	----------------------

## 8.9. readDate()

```
public java.util.Date readDate();
```

Read date object

### Parameters

<i>return</i>	Date
---------------	------

## 8.10. readKeyValues(Deserializer)

```
public java.util.Map<java.lang.String, java.lang.Object> readKeyValues(Deserializer deserializer);
```

Read key - value pairs. This is required for the RecordSet deserializer.

## 8.11. readMap(Deserializer)

```
public Object readMap(Deserializer deserializer);
```

Read a map containing key - value pairs. This can result in a List or Map being deserialized depending on the map type found.

### Parameters

<i>return</i>	Map
---------------	-----

## 8.12. readNull()

```
public Object readNull();
```

Read Null data type

**Parameters**

<i>return</i>	Null datatype (AS)
---------------	--------------------

**8.13. readNumber()**

```
public Number readNumber();
```

Read Number object

**Parameters**

<i>return</i>	Number
---------------	--------

**8.14. readObject(Deserializer)**

```
public Object readObject(Deserializer deserializer);
```

Read an object.

**Parameters**

<i>return</i>	object
---------------	--------

**8.15. readReference()**

```
public Object readReference();
```

Read reference to Complex Data Type. Objects that are collaborators (properties) of other objects must be stored as references in map of id-reference pairs.

**8.16. readString()**

```
public String readString();
```

Read String object

**Parameters**

<i>return</i>	String
---------------	--------

**8.17. readXML()**

```
public org.w3c.dom.Document readXML();
```

Read XML document

**Parameters**

<i>return</i>	XML DOM document
---------------	------------------

**9. Interface Output**

Output interface which defines contract methods to be implemented

## 9.1. Synopsis

```
public interface Output {
// Public Methods

    public void clearReferences();

    public boolean isCustom(Object custom);

    public void putString(String string);

    public boolean supportsDataType(byte type);

    public void writeArray(Object array,
                          Serializer serializer);

    public void writeArray(Object[] array,
                          Serializer serializer);

    public void writeArray(java.util.Collection<?> array,
                          Serializer serializer);

    public void writeBoolean(Boolean bol);

    public void writeByteArray(org.red5.io.amf3.ByteArray array);

    public void writeCustom(Object custom);

    public void writeDate(java.util.Date date);

    public void writeMap(java.util.Collection<?> array,
                       Serializer serializer);

    public void writeMap(java.util.Map<java.lang.Object, java.lang.Object> map,
                       Serializer serializer);

    public void writeNull();

    public void writeNumber(Number num);

    public void writeObject(Object object,
                           Serializer serializer);

    public void writeObject(java.util.Map<java.lang.Object, java.lang.Object> map,
                           Serializer serializer);

    public void writeRecordSet(RecordSet recordset,
                             Serializer serializer);

    public void writeReference(Object obj);

    public void writeString(String string);

    public void writeXML(org.w3c.dom.Document xml);
}
```

}

## 9.2. clearReferences()

```
public void clearReferences();
```

Clear references

## 9.3. isCustom(Object)

```
public boolean isCustom(Object custom);
```

Whether object is custom

### Parameters

custom	Object
--------	--------

<i>return</i>	true if object is of user type, false otherwise
---------------	---

## 9.4. writeArray(Collection<?>, Serializer)

```
public void writeArray(java.util.Collection<?> array,
                      Serializer serializer);
```

Write array.

### Parameters

array	Array to write.
-------	-----------------

serializer	Serializer to use for subobjects.
------------	-----------------------------------

## 9.5. writeArray(Object[], Serializer)

```
public void writeArray(Object[] array,
                      Serializer serializer);
```

Write array.

### Parameters

array	Array to write.
-------	-----------------

serializer	Serializer to use for subobjects.
------------	-----------------------------------

## 9.6. writeArray(Object, Serializer)

```
public void writeArray(Object array,
                      Serializer serializer);
```

Write primitive array.

### Parameters

array	Array to write.
serializer	Serializer to use for subobjects.

## 9.7. writeBoolean(Boolean)

```
public void writeBoolean(Boolean bol);
```

Write boolean

### Parameters

bol	Boolean
-----	---------

## 9.8. writeByteArray(ByteArray)

```
public void writeByteArray(org.red5.io.amf3.ByteArray array);
```

Write ByteArray object (AMF3 only).

### Parameters

array	object to write
-------	-----------------

## 9.9. writeCustom(Object)

```
public void writeCustom(Object custom);
```

Write custom (user) object

### Parameters

custom	Custom data type object
--------	-------------------------

## 9.10. writeDate(Date)

```
public void writeDate(java.util.Date date);
```

Write date

### Parameters

date	Date
------	------

## 9.11. writeMap(Collection<?>, Serializer)

```
public void writeMap(java.util.Collection<?> array,
                     Serializer serializer);
```

Write array as map.

### Parameters

array	Array to write
-------	----------------

serializer	Serializer to use for subobjects.
------------	-----------------------------------

## 9.12. writeMap(Map<Object, Object>, Serializer)

```
public void writeMap(java.util.Map<java.lang.Object, java.lang.Object> map,
                     Serializer serializer);
```

Write map.

### Parameters

map	Map to write
serializer	Serializer to use for subobjects.

## 9.13. writeNumber(Number)

```
public void writeNumber(Number num);
```

Write number

### Parameters

num	Number
-----	--------

## 9.14. writeObject(Map<Object, Object>, Serializer)

```
public void writeObject(java.util.Map<java.lang.Object, java.lang.Object> map,
                       Serializer serializer);
```

Write map as object.

### Parameters

map	Map to write
serializer	Serializer to use for subobjects.

## 9.15. writeObject(Object, Serializer)

```
public void writeObject(Object object,
                       Serializer serializer);
```

Write object.

### Parameters

object	Object to write
serializer	Serializer to use for subobjects.

## 9.16. writeRecordSet(RecordSet, Serializer)

```
public void writeRecordSet(RecordSet recordset,
                          Serializer serializer);
```

Write recordset.

Parameters	
recordset	Recordset to write.
serializer	Serializer to use for subobjects.

## 9.17. writeReference(Object)

```
public void writeReference(Object obj);
```

Write reference to complex data type

Parameters	
obj	Referenced object

## 9.18. writeString(String)

```
public void writeString(String string);
```

Write string

Parameters	
string	String

## 9.19. writeXML(Document)

```
public void writeXML(org.w3c.dom.Document xml);
```

Write XML object

Parameters	
xml	XML document

# 10. Class RecordSet

Read only RecordSet object that might be received through remoting calls. There are 3 types of data fetching:

- On demand (used by default)
- Fetch all at once
- Page-by-page fetching

For last mode, use `page size` property to specify maximum number of rows on one page

## 10.1. Synopsis

```
public class RecordSet {
```

```
// Public Constructors

    public RecordSet(Input input);

// Public Methods

    public java.util.List<java.lang.String> getColumnNames();

    public java.util.List<java.lang.Object> getItemAt(int index);

    public int getLength();

    public int getNumberAvailable();

    public boolean isFullyPopulated();

    public java.util.Map<java.lang.String, java.lang.Object> serialize();

    public void setDeliveryMode(String mode);

    public void setDeliveryMode(String mode,
                                int pageSize);

    public void setDeliveryMode(String mode,
                                int pageSize,
                                int prefetchCount);

    public void setRemotingClient(org.red5.server.net.remoting.RemotingClient client);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

#### See Also

osflash.org documentation [<http://www.osflash.org/amf/recordset>]

## 10.2. RecordSet(Input)

```
public RecordSet(Input input);
```

Creates recordset from Input object

#### Parameters

input
-------

## 10.3. getColumnNames()

```
public java.util.List<java.lang.String> getColumnNames();
```

Return a list containing the names of the columns in the recordset.

#### Parameters

<i>return</i>	Column names set
---------------	------------------

## 10.4. getItemAt(int)

```
public java.util.List<java.lang.Object> getItemAt(int index);
```

Return a specified item from the recordset. If the item is not available yet, it will be received from the server.

### Parameters

index	Item index
<i>return</i>	Item from recordset

## 10.5. getLength()

```
public int getLength();
```

Get the total number of items.

### Parameters

<i>return</i>	Number of items
---------------	-----------------

## 10.6. getNumberAvailable()

```
public int getNumberAvailable();
```

Get the number of items already received from the server.

### Parameters

<i>return</i>	Nsumber of received items
---------------	---------------------------

## 10.7. isFullyPopulated()

```
public boolean isFullyPopulated();
```

Check if all items are available on the client.

### Parameters

<i>return</i>	number of available items
---------------	---------------------------

## 10.8. serialize()

```
public java.util.Map<java.lang.String, java.lang.Object> serialize();
```

Return Map that can be serialized as result.

### Parameters

<i>return</i>	serializable informations
---------------	---------------------------

## 10.9. setDeliveryMode(String)

```
public void setDeliveryMode(String mode);
```

Set the mode for fetching paged results.

### Parameters

mode	Mode for fetching of results
------	------------------------------

## 10.10. setDeliveryMode(String, int)

```
public void setDeliveryMode(String mode,
                            int pageSize);
```

Set the mode for fetching paged results with given max page size.

### Parameters

mode	Mode for fetching of results
pageSize	Max page size

## 10.11. setDeliveryMode(String, int, int)

```
public void setDeliveryMode(String mode,
                            int pageSize,
                            int prefetchCount);
```

Set the mode for fetching paged results with given max page size and number of prefetched pages.

### Parameters

mode	Mode for fetching of results
pageSize	Max page size
prefetchCount	Number of prefetched pages (not implemented yet)

## 10.12. setRemotingClient(RemotingClient)

```
public void setRemotingClient(org.red5.server.net.remoting.RemotingClient client);
```

Set the remoting client to use for retrieving of paged results.

### Parameters

client	Remoting client that works with this Recordset
--------	--

# 11. Class RecordSetPage

Result of pageable request, one page of data.

## 11.1. Synopsis

```
public class RecordSetPage {
    // Public Constructors

    public RecordSetPage(Input input);

    // Protected Methods

    protected int getCursor();

    protected java.util.List<java.util.List<java.lang.Object>> getData();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### See Also

osflash.org documentation [<http://www.osflash.org/amf/recordset>]

## 11.2. RecordSetPage(Input)

```
public RecordSetPage(Input input);
```

Creates recordset page from Input object

#### Parameters

input	Input object to use as source for data that has to be deserialized
-------	--

## 11.3. getCursor()

```
protected int getCursor();
```

Getter for recordset cursor

#### Parameters

return	Recordset cursor
--------	------------------

## 11.4. getData()

```
protected java.util.List<java.util.List<java.lang.Object>> getData();
```

Getter for page data

#### Parameters

return	Page data as unmodifiable list
--------	--------------------------------

## 12. Class Serializer

The Serializer class writes data output and handles the data according to the core data types

## 12.1. Synopsis

```

public class Serializer {
    // Protected Fields

    protected static org.slf4j.Logger log;

    // Public Constructors

    public Serializer();

    // Public Methods

    public Object preProcessExtension(Object any);

    public void serialize(Output out,
                          Object any);

    public boolean writeComplex(Output out,
                               Object complex);

    // Protected Methods

    protected boolean writeArrayType(Output out,
                                    Object arrType);

    protected boolean writeBasic(Output out,
                               Object basic);

    protected boolean writeCustomType(Output out,
                                    Object obj);

    protected void writeDocument(Output out,
                               org.w3c.dom.Document doc);

    protected void writeIterator(Output out,
                               java.util.Iterator<java.lang.Object> it);

    protected void writeList(Output out,
                           java.util.List<?> list);

    protected boolean writeListType(Output out,
                                Object listType);

    protected boolean writeObjectType(Output out,
                                    Object obj);

    protected boolean writeXMLType(Output out,
                                Object xml);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 12.2. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 12.3. preProcessExtension(Object)

```
public Object preProcessExtension(Object any);
```

Pre processes an object TODO // must be implemented

### Parameters

any	Object to preprocess
<i>return</i>	Prerocessed object

## 12.4. serialize(Output, Object)

```
public void serialize(Output out,
                      Object any);
```

Serializes output to a core data type object

### Parameters

out	Output writer
any	Object to serialize

## 12.5. writeArrayType(Output, Object)

```
protected boolean writeArrayType(Output out,
                                 Object arrType);
```

Writes array (or collection) out as output Arrays, Collections, etc

### Parameters

out	Output object
arrType	Array or collection type
<i>return</i>	true if the object has been written, otherwise false

## 12.6. writeBasic(Output, Object)

```
protected boolean writeBasic(Output out,
                            Object basic);
```

Writes a primitive out as an object

### Parameters

out	Output writer
-----	---------------

basic	Primitive
<i>return</i>	boolean true if object was successfully serialized, false otherwise

## 12.7. writeComplex(Output, Object)

```
public boolean writeComplex(Output out,
                           Object complex);
```

Writes a complex type object out

Parameters	
out	Output writer
complex	Complex datatype object
<i>return</i>	boolean true if object was successfully serialized, false otherwise

## 12.8. writeCustomType(Output, Object)

```
protected boolean writeCustomType(Output out,
                                 Object obj);
```

Writes a custom data to the output

Parameters	
out	Output writer
obj	Custom data
<i>return</i>	true if the object has been written, otherwise false

## 12.9. writeDocument(Output, Document)

```
protected void writeDocument(Output out,
                            org.w3c.dom.Document doc);
```

Writes a document to the output

Parameters	
out	Output writer
doc	Document to write

## 12.10. writeIterator(Output, Iterator<Object>)

```
protected void writeIterator(Output out,
                            java.util.Iterator<java.lang.Object> it);
```

Writes an iterator out to the output

Parameters	
------------	--

out	Output writer
it	Iterator to write

## 12.11. writeList(Output, List<?>)

```
protected void writeList(Output out,
                      java.util.List<?> list);
```

Writes a List out as an Object

Parameters	
out	Output writer
list	List to write as Object

## 12.12. writeListType(Output, Object)

```
protected boolean writeListType(Output out,
                               Object listType);
```

Writes Lists out as a data type

Parameters	
out	Output write
listType	List type
<i>return</i>	boolean true if object was successfully serialized, false otherwise

## 12.13. writeObjectType(Output, Object)

```
protected boolean writeObjectType(Output out,
                                 Object obj);
```

Write typed object to the output

Parameters	
out	Output writer
obj	Object type to write
<i>return</i>	true if the object has been written, otherwise false

## 12.14. writeXMLType(Output, Object)

```
protected boolean writeXMLType(Output out,
                               Object xml);
```

Writes an xml type out to the output

Parameters	

out	Output writer
xml	XML
<i>return</i>	boolean <code>true</code> if object was successfully written, <code>false</code> otherwise

# 1. Class BufferUtils

Buffer Utility class which reads/writes intergers to the input/output buffer

## 1.1. Synopsis

```
public class BufferUtils {  
    // Public Constructors  
  
    public BufferUtils();  
  
    // Public Static Methods  
  
    public static int put(org.apache.mina.common.ByteBuffer out,  
                         org.apache.mina.common.ByteBuffer in,  
                         int numBytesMax);  
  
    public static int readMediumInt(org.apache.mina.common.ByteBuffer in);  
  
    public static int readUnsignedMediumInt(org.apache.mina.common.ByteBuffer in);  
  
    public static void writeMediumInt(org.apache.mina.common.ByteBuffer out,  
                                     int value);  
}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 1.2. put(ByteBuffer, ByteBuffer, int)

```
public static int put(org.apache.mina.common.ByteBuffer out,  
                     org.apache.mina.common.ByteBuffer in,  
                     int numBytesMax);
```

Puts input buffer stream to output buffer and returns number of bytes written

Parameters	
out	Output buffer
in	Input buffer
numBytesMax	Number of bytes max
<i>return</i>	int Number of bytes written

## 1.3. readMediumInt(ByteBuffer)

```
public static int readMediumInt(org.apache.mina.common.ByteBuffer in);
```

Reads a Medium Int to the in buffer

Parameters
------------

in	Source
return	int Medium int

## 1.4. readUnsignedMediumInt(ByteBuffer)

```
public static int readUnsignedMediumInt(org.apache.mina.common.ByteBuffer in);
```

Reads an unsigned Medium Int from the in buffer

Parameters	
in	Source
return	int Integer value

## 1.5. writeMediumInt(ByteBuffer, int)

```
public static void writeMediumInt(org.apache.mina.common.ByteBuffer out,
                                 int value);
```

Writes a Medium Int to the output buffer

Parameters	
out	Container to write to
value	Integer to write

## 2. Class ByteBufferUtil

ByteBuffer utility.

*Note: Paul added this back in for use with Mina due to its removal from Mina 2.0*

### 2.1. Synopsis

```
@Deprecated public class ByteBufferUtil {
// Public Static Methods

    public static void acquireIfPossible(Object message);

    public static void releaseIfPossible(Object message);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 3. Class DOM2Writer

This class is a utility to serialize a DOM node as XML. This class uses the DOM Level 2 APIs. The main difference between this class and DOMWriter is that this class generates and prints out namespace declarations.

### 3.1. Synopsis

```
public class DOM2Writer {
// Public Constructors

    public DOM2Writer();

// Public Static Methods

    public static void serializeAsXML(org.w3c.dom.Node node,
                                      java.io.Writer writer);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 3.2. serializeAsXML(Node, Writer)

```
public static void serializeAsXML(org.w3c.dom.Node node,
                                  java.io.Writer writer);
```

Serialize this node into the writer as XML.

Parameters	
writer	Writer object
node	DOM node

## 4. Class HexCharset

This was borrowed from the Soupdragon base64 library.

### 4.1. Synopsis

```
public class HexCharset extends java.nio.charset.Charset {
// Public Constructors

    public HexCharset(boolean caps);

    public HexCharset(boolean caps,
                      int measure);

// Public Methods

    public boolean contains(java.nio.charset.Charset cs);

    public java.nio.charset.CharsetDecoder newDecoder();

    public java.nio.charset.CharsetEncoder newEncoder();
```

```
}
```

**Methods inherited from java.nio.charset.Charset:** aliases, availableCharsets, canEncode, compareTo, contains, decode, defaultCharset, displayName, encode, equals, forName, hashCode, isRegistered, isSupported, name, newDecoder, newEncoder, toString

**Methods inherited from java.lang.Object:** clone, finalize, getClass, notify, notifyAll, wait

## 5. Class HexCharset.Decoder

```
public class HexCharset.Decoder extends java.nio.charset.CharsetDecoder {
// Public Methods

    public java.nio.charset.CoderResult decodeLoop(java.nio.ByteBuffer in,
                                                    java.nio.CharBuffer out);

// Protected Methods

    protected void implReset();

}
```

**Methods inherited from java.nio.charset.CharsetDecoder:** averageCharsPerByte, charset, decode, decodeLoop, detectedCharset, flush, implFlush, implOnMalformedInput, implOnUnmappableCharacter, implReplaceWith, implReset, isAutoDetecting, isCharsetDetected, malformedInputAction, maxCharsPerByte, onMalformedInput, onUnmappableCharacter, replacement, replaceWith, reset, unmappableCharacterAction

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 6. Class HexCharset.Encoder

```
public class HexCharset.Encoder extends java.nio.charset.CharsetEncoder {
// Public Methods

    public java.nio.charset.CoderResult encodeLoop(java.nio.CharBuffer in,
                                                    java.nio.ByteBuffer out);

// Protected Methods

    protected java.nio.charset.CoderResult implFlush(java.nio.ByteBuffer out);

    protected void implReset();

}
```

**Methods inherited from java.nio.charset.CharsetEncoder:** averageBytesPerChar, canEncode, charset, encode, encodeLoop, flush, implFlush, implOnMalformedInput, implOnUnmappableCharacter, implReplaceWith, implReset, isLegalReplacement,

```
malformedInputAction , maxBytesPerChar , onMalformedInput , onUnmappableCharacter ,
replacement , replaceWith , reset , unmappableCharacterAction
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 7. Class HexDump

Hexadecimal byte dumper

### 7.1. Synopsis

```
public class HexDump {
// Public Constructors

    public HexDump();

// Public Static Methods

    public static String byteArrayToBinaryString(byte[] block);

    public static String byteArrayToHexString(byte[] block);

    public static String byteArrayToHexString(byte[] block,
                                              int offset,
                                              int length);

    public static String formatHexDump(String in);

    public static byte[] hexStringToByteArray(String strA);

    public static void main(String[] args);

    public static String prettyPrintHex(byte[] baToConvert);

    public static String prettyPrintHex(String sToConvert);

    public static String prettyPrintHex(java.nio.ByteBuffer bbToConvert);

    public static void setBitDigits(char zeroBit,
                                   char oneBit);

    public static void setBitDigits(char[] bd)
        throws Exception;

    public static void setByteSeparator(char bs);

    public static void setWithByteSeparator(boolean bs);

    public static String stringToHexString(String in);

    public static String toBinaryString(byte b);

    public static String toBinaryString(byte[] ba);
```

```

public static String toBinaryString(int i);

public static String toBinaryString(long l);

public static String toBinaryString(short s);

public static final byte[] toByteArray(int i);

public static final byte[] toByteArray(long l);

public static final byte[] toByteArray(short s);

public static String toHexString(byte b);

public static String toHexString(byte[] ba);

public static String toHexString(byte[] ba,
                               int offset,
                               int length);

public static String toHexString(int i);

public static String toHexString(long l);

public static String toHexString(short s);

public static String toString(byte b);

public static String toString(byte[] ba);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 7.2. byteArrayToBinaryString(byte[])

```
public static String byteArrayToBinaryString(byte[] block);
```

Description of the Method

### Parameters

block	Description of Parameter
<i>return</i>	Description of the Returned Value

## 7.3. byteArrayToHexString(byte[])

```
public static String byteArrayToHexString(byte[] block);
```

Description of the Method

**Parameters**

<i>block</i>	Description of Parameter
<i>return</i>	Description of the Returned Value

**7.4. byteArrayToHexString(byte[], int, int)**

```
public static String byteArrayToHexString(byte[] block,
                                         int offset,
                                         int length);
```

**Description of the Method****Parameters**

<i>block</i>	Description of Parameter
<i>offset</i>	Description of Parameter
<i>length</i>	Description of Parameter
<i>return</i>	Description of the Returned Value

**7.5. hexStringToByteArray(String)**

```
public static byte[] hexStringToByteArray(String strA);
```

**Converts readable hex-String to byteArray****Parameters**

<i>strA</i>	
<i>return</i>	the hexadecimal string as byte array

**7.6. main(String[])**

```
public static void main(String[] args);
```

**test and demo for the Convert class****Parameters**

<i>args</i>	none needed
-------------	-------------

**7.7. prettyPrintHex(byte[])**

```
public static String prettyPrintHex(byte[] baToConvert);
```

**Method prettyPrintHex****Parameters**

baToConvert	Array of bytes to encode
<i>return</i>	Hexdump string

## 7.8. prettyPrintHex(ByteBuffer)

```
public static String prettyPrintHex(java.nio.ByteBuffer bbToConvert);
```

Method prettyPrintHex

Parameters	
bbToConvert	ByteBuffer to encode
<i>return</i>	Hexdump string

## 7.9. prettyPrintHex(String)

```
public static String prettyPrintHex(String sToConvert);
```

Method prettyPrintHex

Parameters	
sToConvert	
<i>return</i>	hexdump string

## 7.10. setBitDigits(char[])

```
public static void setBitDigits(char[] bd)
throws Exception;
```

Sets the BitDigits attribute of the Convert class

Parameters	
bd	The new BitDigits value

Exception

Description of Exception

## 7.11. setBitDigits(char, char)

```
public static void setBitDigits(char zeroBit,
char oneBit);
```

Method setBitDigits

Parameters	
zeroBit	
oneBit	

## 7.12. setByteSeparator(char)

```
public static void setByteSeparator(char bs);
```

Sets the ByteSeparator attribute of the Convert class

### Parameters

bs	The new ByteSeparator value
----	-----------------------------

## 7.13. setWithByteSeparator(boolean)

```
public static void setWithByteSeparator(boolean bs);
```

Sets the WithByteSeparator attribute of the Convert class

### Parameters

bs	The new WithByteSeparator value
----	---------------------------------

## 7.14. stringToHexString(String)

```
public static String stringToHexString(String in);
```

Description of the Method

### Parameters

in	string to be converted
<i>return</i>	String in readable hex encoding

## 7.15. toBinaryString(byte)

```
public static String toBinaryString(byte b);
```

Method toBinaryString

### Parameters

b	
<i>return</i>	the binary representation of the byte

## 7.16. toBinaryString(byte[])

```
public static String toBinaryString(byte[] ba);
```

Method toBinaryString

### Parameters

ba	
<i>return</i>	the binary representation of the byte array

## 7.17. toBinaryString(int)

```
public static String toBinaryString(int i);
```

Method toBinaryString

### Parameters

i	
---	--

<i>return</i>	the binary representation of the int
---------------	--------------------------------------

## 7.18. toBinaryString(long)

```
public static String toBinaryString(long l);
```

Method toBinaryString

### Parameters

l	
---	--

<i>return</i>	the binary representation of the long
---------------	---------------------------------------

## 7.19. toBinaryString(short)

```
public static String toBinaryString(short s);
```

Method toBinaryString

### Parameters

s	
---	--

<i>return</i>	the binary representation of the short
---------------	--

## 7.20. toByteArray(int)

```
public static final byte[] toByteArray(int i);
```

Method toByteArray

### Parameters

i	
---	--

<i>return</i>	the int as array of bytes
---------------	---------------------------

## 7.21. toByteArray(long)

```
public static final byte[] toByteArray(long l);
```

Method toByteArray

### Parameters

<i>return</i>	the long as array of bytes

## 7.22. toByteArray(short)

```
public static final byte[] toByteArray(short s);
```

Method toByteArray

Parameters	
s	
<i>return</i>	the short as array of bytes

## 7.23. toHexString(byte)

```
public static String toHexString(byte b);
```

Method toHexString

Parameters	
b	
<i>return</i>	the hexadecimal representation of the byte

## 7.24. toHexString(byte[])

```
public static String toHexString(byte[] ba);
```

Returns a string of hexadecimal digits from a byte array. Each byte is converted to 2 hex symbols.

Parameters	
ba	Description of Parameter
<i>return</i>	Description of the Returned Value

## 7.25. toHexString(byte[], int, int)

```
public static String toHexString(byte[] ba,
                                int offset,
                                int length);
```

converts String to Hex String. Example: niko ->6E696B6F

Parameters	
ba	Description of Parameter
offset	Description of Parameter

length	Description of Parameter
<i>return</i>	Description of the Returned Value

## 7.26. toHexString(int)

```
public static String toHexString(int i);
```

Method toHexString

Parameters	
i	
<i>return</i>	the hexadecimal representation of the int

## 7.27. toHexString(long)

```
public static String toHexString(long l);
```

Method toHexString

Parameters	
l	
<i>return</i>	the hexadecimal representation of the long

## 7.28. toHexString(short)

```
public static String toHexString(short s);
```

Description of the Method

Parameters	
s	
<i>return</i>	Description of the Returned Value

## 7.29. toString(byte)

```
public static String toString(byte b);
```

Method toString

Parameters	
b	
<i>return</i>	the byte as string

## 7.30. toString(byte[])

```
public static String toString(byte[] ba);
```

Method `toString`

Parameters	
ba	
<i>return</i>	the byte array as string

## 8. Class IOUtils

Misc I/O util methods

### 8.1. Synopsis

```

public class IOUtils {
    // Public Static Fields

        public static final java.nio.charset.Charset CHARSET ;

    // Public Constructors

        public IOUtils();

    // Public Static Methods

        public static void debug(org.slf4j.Logger log,
                               String msg,
                               org.apache.mina.common.ByteBuffer buf);

        public static int readMediumInt(org.apache.mina.common.ByteBuffer in);

        public static int readMediumInt2(org.apache.mina.common.ByteBuffer in);

        public static int readReverseInt(org.apache.mina.common.ByteBuffer in);

        public static int readUnsignedMediumInt(org.apache.mina.common.ByteBuffer in);

        public static String toString(org.apache.mina.common.ByteBuffer buf);

        public static void writeMediumInt(org.apache.mina.common.ByteBuffer out,
                                         int value);

        public static void writeReverseInt(org.apache.mina.common.ByteBuffer out,
                                         int value);

}

```

**Methods inherited from `java.lang.Object`:** `clone` , `equals` , `finalize` , `getClass` , `hashCode` , `notify` , `notifyAll` , `toString` , `wait`

## 8.2. CHARSET

public static final java.nio.charset.Charset CHARSET ;
--

UTF-8 is used

### 8.3. debug(Logger, String, ByteBuffer)

```
public static void debug(org.slf4j.Logger log,
                        String msg,
                        org.apache.mina.common.ByteBuffer buf);
```

Format debug message

#### Parameters

log	Logger
msg	Message
buf	Byte buffer to debug

### 8.4. readMediumInt(ByteBuffer)

```
public static int readMediumInt(org.apache.mina.common.ByteBuffer in);
```

Reads medium int

#### Parameters

in	Source
<i>return</i>	int value

### 8.5. readMediumInt2(ByteBuffer)

```
public static int readMediumInt2(org.apache.mina.common.ByteBuffer in);
```

Alternate method for reading medium int

#### Parameters

in	Source
<i>return</i>	int value

### 8.6. readReverseInt(ByteBuffer)

```
public static int readReverseInt(org.apache.mina.common.ByteBuffer in);
```

Reads reverse int

#### Parameters

in	Source
<i>return</i>	int

### 8.7. readUnsignedMediumInt(ByteBuffer)

```
public static int readUnsignedMediumInt(org.apache.mina.common.ByteBuffer in);
```

Reads unsigned medium integer

#### Parameters

in	Unsigned medium int source
<i>return</i>	int value

## 8.8. **toString(ByteBuffer)**

```
public static String toString(org.apache.mina.common.ByteBuffer buf);
```

String representation of byte buffer

#### Parameters

buf	Byte buffer
<i>return</i>	String representation

## 8.9. **writeMediumInt(ByteBuffer, int)**

```
public static void writeMediumInt(org.apache.mina.common.ByteBuffer out,
                                 int value);
```

Writes medium integer

#### Parameters

out	Output buffer
<i>value</i>	Integer to write

## 8.10. **writeReverseInt(ByteBuffer, int)**

```
public static void writeReverseInt(org.apache.mina.common.ByteBuffer out,
                                   int value);
```

Writes integer in reverse order

#### Parameters

out	Data buffer to fill
<i>value</i>	Integer

## 9. Class ObjectMap

Map that should be transmitted as object through RTMP.

### 9.1. Synopsis

```
public class ObjectMap<K,V> extends java.util.HashMap {
    // Public Constructors
```

```
public ObjectMap();
}
```

**Methods inherited from java.util.HashMap:** clear, clone, containsKey, containsValue, entrySet, get, isEmpty, keySet, put, putAll, remove, size, values

**Methods inherited from java.util.AbstractMap:** equals, hashCode, toString

**Methods inherited from java.lang.Object:** finalize, getClass, notify, notifyAll, wait

## 10. Class RandomGUID

```
public class RandomGUID {
// Public Fields

    public String valueAfterMD5;

    public String valueBeforeMD5;

// Public Constructors

    public RandomGUID();

    public RandomGUID(boolean secure);

// Public Static Methods

    public static void main(String[] args);

// Public Methods

    public String toString();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, , notify, notifyAll, toString, wait

## 11. Class XMLUtils

Misc XML utils

### 11.1. Synopsis

```
public class XMLUtils {
// Protected Fields

    protected static org.slf4j.Logger log;

// Public Constructors

    public XMLUtils();
```

```
// Public Static Methods

    public static String docToString(org.w3c.dom.Document dom);

    public static String docToString1(org.w3c.dom.Document dom);

    public static String docToString2(org.w3c.dom.Document domDoc)
        throws IOException;

    public static org.w3c.dom.Document stringToDoc(String str)
        throws IOException;

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 11.2. log

```
protected static org.slf4j.Logger log;
```

Logger

## 11.3. docToString(Document)

```
public static String docToString(org.w3c.dom.Document dom);
```

Converts doc to String

### Parameters

dom	DOM object to convert
<i>return</i>	XML as String

## 11.4. docToString1(Document)

```
public static String docToString1(org.w3c.dom.Document dom);
```

Convert a DOM tree into a String using Dom2Writer

### Parameters

dom	DOM object to convert
<i>return</i>	XML as String

## 11.5. docToString2(Document)

```
public static String docToString2(org.w3c.dom.Document domDoc)
    throws IOException;
```

Convert a DOM tree into a String using transform

**Parameters**

domDoc	DOM object
<i>return</i>	XML as String

java.io.IOException

I/O exception

## 11.6. stringToDoc(String)

```
public static org.w3c.dom.Document stringToDoc(String str)
throws IOException;
```

Converts string representation of XML into Document

**Parameters**

str	String representation of XML
<i>return</i>	DOM object

IOException

I/O exception

# 1. Class ContextLoggingListener

A servlet context listener that puts this contexts LoggerContext into a static map of logger contexts within an overall singleton log context selector. To use it, add the following line to a web.xml file

```
<listener>
    <listener-class>org.red5.logging.ContextLoggingListener</listene\
ner-class>
</listener>
```

## 1.1. Synopsis

```
public class ContextLoggingListener implements javax.servlet.ServletContextListener {
// Public Constructors

    public ContextLoggingListener();

// Public Methods

    public void contextDestroyed(javax.servlet.ServletContextEvent event);

    public void contextInitialized(javax.servlet.ServletContextEvent event);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

# 2. Class LoggerContextFilter

A servlet filter that puts this contexts LoggerContext into a Threadlocal variable. It removes it after the request is processed. To use it, add the following lines to a web.xml file

```
<filter>
    <filter-name>LoggerContextFilter</filter-name>
    <filter-class>org.red5.logging.LoggerContextFilter</filter-cla\
ss>
</filter>
<filter-mapping>
    <filter-name>LoggerContextFilter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

## 2.1. Synopsis

```
public class LoggerContextFilter implements javax.servlet.Filter {
// Public Constructors

    public LoggerContextFilter();

// Public Methods

    public void destroy();
```

```

public void doFilter(javax.servlet.ServletRequest request,
                     javax.servlet.ServletResponse response,
                     javax.servlet.FilterChain chain)
throws IOException, ServletException;

public void init(javax.servlet.FilterConfig config)
throws ServletException;

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 3. Class LoggingContextSelector

A class that allows the LoggerFactory to access an web context based LoggerContext. Add this java option -Dlogback.ContextSelector=org.red5.logging.LoggingContextSelector

### 3.1. Synopsis

```

public class LoggingContextSelector implements ch.qos.logback.classic.selector.ContextSelector {
// Public Constructors

    public LoggingContextSelector(ch.qos.logback.classic.LoggerContext context);

// Public Methods

    public void attachLoggerContext(String contextName,
                                    ch.qos.logback.classic.LoggerContext loggerContext);

    public ch.qos.logback.classic.LoggerContext detachLoggerContext(String loggerContextName);

    public java.util.List<java.lang.String> getContextNames();

    public int getCount();

    public ch.qos.logback.classic.LoggerContext getDefaultLoggerContext();

    public ch.qos.logback.classic.LoggerContext getLoggerContext();

    public ch.qos.logback.classic.LoggerContext getLoggerContext(String name);

    public void removeLocalContext();

    public void setContextConfigFile(String contextConfigFile);

    public void setContextName(String contextName);

    public void setLocalContext(ch.qos.logback.classic.LoggerContext context);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 3.2. getCount()

```
public int getCount();
```

Returns the number of managed contexts Used for testing purposes

#### Parameters

<i>return</i>	the number of managed contexts
---------------	--------------------------------

### 3.3. setLocalContext(LoggerContext)

```
public void setLocalContext(ch.qos.logback.classic.LoggerContext context);
```

These methods are used by the LoggerContextFilter. They provide a way to tell the selector which context to use, thus saving the cost of a JNDI call at each new request.

#### Parameters

context
---------

## 4. Class W3CAppender

Logback appender for the Extended W3C format.

### 4.1. Synopsis

```
public class W3CAppender extends ch.qos.logback.core.FileAppender {
// Public Constructors

    public W3CAppender();

// Public Methods

    public synchronized void doAppend(ch.qos.logback.classic.spi.LoggingEvent event);

    public String getEvents();

    public String getFields();

    public void setEvents(String events);

    public void setFields(String fields);

}
```

**Methods inherited from ch.qos.logback.core.FileAppender:** `getAppend`, `getBufferSize`, `getFile`, `isBufferedIO`, `setAppend`, `setBufferedIO`, `setBufferSize`, `setFile`, `start`

**Methods inherited from ch.qos.logback.core.WriterAppender:** `append`, `closeWriter`, `createWriter`, `getEncoding`, `getImmediateFlush`, `getLayout`, `setEncoding`, `setImmediateFlush`, `setLayout`, `setWriter`, `stop`, `subAppend`

**Methods inherited from ch.qos.logback.core.AppenderBase:** addFilter ,  
clearAllFilters , doAppend , getFilterChainDecision , getFirstFilter , getName , isStarted  
, setName , toString

**Methods inherited from ch.qos.logback.core.spi.ContextAwareBase:** addError ,  
addInfo , addStatus , addWarn , getContext , getStatusManager , setContext

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , wait

**Fields inherited from ch.qos.logback.core.FileAppender:** append , bufferedIO ,  
bufferSize , fileName

**Fields inherited from ch.qos.logback.core.WriterAppender:** encoding , immediateFlush ,  
layout , writer

**Fields inherited from ch.qos.logback.core.AppenderBase:** name , started

**Fields inherited from ch.qos.logback.core.spi.ContextAwareBase:** context

See Also

<http://www.w3.org/TR/WD-logfile.html>

# 1. Class SimpleClient

Sample class that uses the client mode of the RTMP library to connect to the "oflaDemo" application on the current server.

## 1.1. Synopsis

```
public class SimpleClient implements org.red5.server.net.rtmp.IRTMPHandler {  
    // Public Constructors  
  
    public SimpleClient();  
  
    // Public Static Methods  
  
    public static void main(String[] args);  
  
    // Public Methods  
  
    public void connectionClosed(org.red5.server.net.rtmp.RTMPConnection conn,  
                                org.red5.server.net.rtmp.codec.RTMP state);  
  
    public void connectionOpened(org.red5.server.net.rtmp.RTMPConnection conn,  
                                org.red5.server.net.rtmp.codec.RTMP state);  
  
    public void messageReceived(org.red5.server.net.rtmp.RTMPConnection conn,  
                               org.red5.server.net.protocol.ProtocolState state,  
                               Object message)  
        throws Exception;  
  
    public void messageSent(org.red5.server.net.rtmp.RTMPConnection conn,  
                           Object message);  
  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### See Also

<http://mirror1.cvsdude.com/trac/osflash/red5/ticket/94>

## 1.2. connectionClosed(RTMPConnection, RTMP)

```
public void connectionClosed(org.red5.server.net.rtmp.RTMPConnection conn,  
                            org.red5.server.net.rtmp.codec.RTMP state);
```

**Specified by:** Method connectionClosed in interface IRTMPHandler

Connection closed

## 1.3. connectionOpened(RTMPConnection, RTMP)

```
public void connectionOpened(org.red5.server.net.rtmp.RTMPConnection conn,  
                            org.red5.server.net.rtmp.codec.RTMP state);
```

**Specified by:** Method connectionOpened in interface IRTMPHandler

Connection open event

## 1.4. messageReceived(RTMPConnection, ProtocolState, Object)

```
public void messageReceived(org.red5.server.net.rtmp.RTMPConnection conn,  
                           org.red5.server.net.protocol.ProtocolState state,  
                           Object message)  
throws Exception;
```

**Specified by:** Method messageReceived in interface IRTMPHandler

Message received

## 1.5. messageSent(RTMPConnection, Object)

```
public void messageSent(org.red5.server.net.rtmp.RTMPConnection conn,  
                       Object message);
```

**Specified by:** Method messageSent in interface IRTMPHandler

Message sent

# 1. Class ClientManager

Class that keeps a list of client names in a SharedObject.

## 1.1. Synopsis

```
public class ClientManager {  
    // Public Constructors  
  
    public ClientManager(String name,  
                        boolean persistent);  
  
    // Public Methods  
  
    public void addClient(org.red5.server.api.IScope scope,  
                         String username,  
                         String uid);  
  
    public String removeClient(org.red5.server.api.IScope scope,  
                             String uid);  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. ClientManager(String, boolean)

```
public ClientManager(String name,  
                     boolean persistent);
```

Create a new instance of the client manager.

Parameters	
name	name of the shared object to use
persistent	should the shared object be persistent

## 1.3. addClient(IScope, String, String)

```
public void addClient(org.red5.server.api.IScope scope,  
                      String username,  
                      String uid);
```

A new client connected. This adds the username to the shared object of the passed scope.

Parameters	
scope	scope the client connected to
username	name of the user that connected
uid	the unique id of the user that connected

## 1.4. removeClient(IScope, String)

```
public String removeClient(org.red5.server.api.IScope scope,  
                           String uid);
```

A client disconnected. This removes the username from the shared object of the passed scope.

### Parameters

scope	scope the client disconnected from
uid	unique id of the user that disconnected
<i>return</i>	the username of the disconnected user

# 1. Class EchoService

The Echo service is used to test all of the different datatypes and to make sure that they are being returned properly.

## 1.1. Synopsis

```
public class EchoService implements org.red5.samples.services.IEchoService {  
    // Public Constructors  
  
    public EchoService();  
  
    // Public Methods  
  
    public Object echoAny(Object any);  
  
    public Object[] echoArray(Object[] array);  
  
    public boolean echoBoolean(boolean bool);  
  
    public java.util.Date echoDate(java.util.Date date);  
  
    public java.util.List echoList(java.util.List list);  
  
    public Object[] echoMultiParam(java.util.Map team,  
                                  java.util.List words,  
                                  String str);  
  
    public double echoNumber(double number);  
  
    public java.util.Map echoObject(java.util.Map obj);  
  
    public String echoString(String string);  
  
    public org.w3c.dom.Document echoXML(org.w3c.dom.Document xml);  
  
    public org.red5.server.api.IConnection returnConnection(Object any);  
  
    public java.util.List<java.lang.Object> returnDistinctObjects(Object any);  
  
    public java.util.List<java.lang.Object> returnSameObjects(Object any);  
  
    public void startUp();  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. echoArray(Object[])

```
public Object[] echoArray(Object[] array);
```

**Specified by:** Method echoArray in interface IEchoService

Verifies that a Flash simple Array that is passed in returns correctly. Flash simple Array = Object[]

#### Parameters

array	
<i>return</i>	input value

#### See Also

`echoArray(java.lang.Object[])`

**Description copied from interface: echoArray**

### 1.3. echoBoolean(boolean)

```
public boolean echoBoolean(boolean bool);
```

**Specified by:** Method echoBoolean in interface IEchoService

Verifies that a boolean that is passed in returns correctly.

#### Parameters

bool	
<i>return</i>	input value

#### See Also

`echoBoolean(boolean)`

**Description copied from interface: echoBoolean**

### 1.4. echoDate(Date)

```
public java.util.Date echoDate(java.util.Date date);
```

**Specified by:** Method echoDate in interface IEchoService

Verifies that a Date that is passed in returns correctly.

#### Parameters

date	
<i>return</i>	input value

#### See Also

`echoDate(java.util.Date)`

**Description copied from interface: echoDate**

### 1.5. echoList(List)

```
public java.util.List echoList(java.util.List list);
```

**See Also**

```
echoList(java.util.List<? extends T>)
```

**1.6. echoNumber(double)**

<pre>public double echoNumber(double number);</pre>
---

**Specified by:** Method echoNumber in interface IEchoService

Verifies that a Number that is passed in returns correctly. Flash Number = double

**Parameters**

num	
<i>return</i>	input value

**See Also**

```
echoNumber(double)
```

**Description copied from interface: echoNumber****1.7. echoObject(Map)**

<pre>public java.util.Map echoObject(java.util.Map obj);</pre>
--

**See Also**

```
echoObject(java.util.Map<? extends K, ? extends V>)
```

**1.8. echoString(String)**

<pre>public String echoString(String string);</pre>
---

**Specified by:** Method echoString in interface IEchoService

Verifies that a String that is passed in returns correctly.

**Parameters**

string	
<i>return</i>	input value

**See Also**

```
echoString(java.lang.String)
```

**Description copied from interface: echoString****1.9. echoXML(Document)**

<pre>public org.w3c.dom.Document echoXML(org.w3c.dom.Document xml);</pre>
---

**Specified by:** Method echoXML in interface IEchoService

Verifies that Flash XML that is passed in returns itself. Flash XML = org.w3c.dom.Document

Parameters	
xml	
<i>return</i>	input value

#### See Also

`echoXML(org.w3c.dom.Document)`

**Description copied from interface: echoXML**

### 1.10. returnConnection(Object)

```
public org.red5.server.api.IConnection returnConnection(Object any);
```

Test returning of internal objects.

Parameters	
any	
<i>return</i>	the current connection

### 1.11. returnDistinctObjects(Object)

```
public java.util.List<java.lang.Object> returnDistinctObjects(Object any);
```

Test serialization of arbitrary objects.

Parameters	
any	
<i>return</i>	list containing distinct objects

### 1.12. returnSameObjects(Object)

```
public java.util.List<java.lang.Object> returnSameObjects(Object any);
```

Test references.

Parameters	
any	
<i>return</i>	list containing same objects

### 1.13. startUp()

```
public void startUp();
```

**Specified by:** Method startUp in interface IEchoService

Used to verify that Spring has loaded the bean.

## 2. Class EchoService.SampleObject

Sample object that contains attributes with all access possibilities. This will test the serializer of arbitrary objects.

### 2.1. Synopsis

```
public class EchoService.SampleObject {
// Public Fields

    public String value1 ;

    public int value2 ;

// Protected Fields

    protected int value4 ;

// Public Constructors

    public EchoService.SampleObject();

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 3. Interface IEchoService

The Echo service is used to test all of the different datatypes and to make sure that they are being returned properly.

### 3.1. Synopsis

```
public interface IEchoService {
// Public Methods

    public Object[] echoArray(Object[] array);

    public boolean echoBoolean(boolean bool);

    public java.util.Date echoDate(java.util.Date date);

    public java.util.List<T> echoList(java.util.List<? extends T> list);

    public double echoNumber(double num);

    public java.util.Map<K, V> echoObject(java.util.Map<? extends K, ? extends V> obj);

    public String echoString(String string);

    public org.w3c.dom.Document echoXML(org.w3c.dom.Document xml);
```

```
public void startUp();
}
```

### 3.2. echoArray(Object[])

```
public Object[] echoArray(Object[] array);
```

Verifies that a Flash simple Array that is passed in returns correctly. Flash simple Array = Object[]

#### Parameters

array	
<i>return</i>	input value

### 3.3. echoBoolean(boolean)

```
public boolean echoBoolean(boolean bool);
```

Verifies that a boolean that is passed in returns correctly.

#### Parameters

bool	
<i>return</i>	input value

### 3.4. echoDate(Date)

```
public java.util.Date echoDate(java.util.Date date);
```

Verifies that a Date that is passed in returns correctly.

#### Parameters

date	
<i>return</i>	input value

### 3.5. echoList(List<? extends T>)

```
public java.util.List<T> echoList(java.util.List<? extends T> list);
```

Verifies that a Flash multi-dimensional Array that is passed in returns itself. Flash multi-dimensional Array = java.util.List

#### Parameters

list	
<i>return</i>	input value

### 3.6. echoNumber(double)

```
public double echoNumber(double num);
```

Verifies that a Number that is passed in returns correctly. Flash Number = double

#### Parameters

num	
<i>return</i>	input value

### 3.7. echoObject(Map<? extends K, ? extends V>)

```
public java.util.Map<K, V> echoObject(java.util.Map<? extends K, ? extends V> obj);
```

Verifies that a Flash Object that is passed in returns correctly. Flash Object = java.util.Map

#### Parameters

obj	
<i>return</i>	input value

### 3.8. echoString(String)

```
public String echoString(String string);
```

Verifies that a String that is passed in returns correctly.

#### Parameters

string	
<i>return</i>	input value

### 3.9. echoXML(Document)

```
public org.w3c.dom.Document echoXML(org.w3c.dom.Document xml);
```

Verifies that Flash XML that is passed in returns itself. Flash XML = org.w3c.dom.Document

#### Parameters

xml	
<i>return</i>	input value

### 3.10. startUp()

```
public void startUp();
```

Used to verify that Spring has loaded the bean.

# 1. Class AttributeStore

```
public class AttributeStore implements org.red5.server.api.ICastingAttributeStore {  
    // Protected Fields  
  
    protected java.util.concurrent.ConcurrentMap<java.lang.String, java.lang.Object> attributes;  
  
    // Public Constructors  
  
    public AttributeStore();  
  
    public AttributeStore(java.util.Map<java.lang.String, java.lang.Object> values);  
  
    public AttributeStore(org.red5.server.api.IAttributeStore values);  
  
    // Public Methods  
  
    public Object getAttribute(String name);  
  
    public Object getAttribute(String name,  
                               Object defaultValue);  
  
    public java.util.Set<java.lang.String> getAttributeNames();  
  
    public java.util.Map<java.lang.String, java.lang.Object> getAttributes();  
  
    public Boolean getBoolAttribute(String name);  
  
    public Byte getByteAttribute(String name);  
  
    public Double getDoubleAttribute(String name);  
  
    public Integer getIntAttribute(String name);  
  
    public java.util.List getListAttribute(String name);  
  
    public Long getLongAttribute(String name);  
  
    public java.util.Map getMapAttribute(String name);  
  
    public java.util.Set getSetAttribute(String name);  
  
    public Short getShortAttribute(String name);  
  
    public String getStringAttribute(String name);  
  
    public boolean hasAttribute(String name);  
  
    public boolean removeAttribute(String name);  
  
    public void removeAttributes();  
  
    public boolean setAttribute(String name,  
                               Object value);
```

```

public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);

public void setAttributes(org.red5.server.api.IAttributeStore values);

// Protected Methods

protected java.util.Map<java.lang.String, java.lang.Object> filterNull(java.util.Map<java.lang.String, java.lang.Object> values);

}

```

**Direct known subclasses:** org.?red5.?server.?BaseConnection , org.?red5.?server.?Client , org.?red5.?server.?PersistableAttributeStore , org.?red5.?server.?so.?SharedObject

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 1.1. AttributeStore()

```
public AttributeStore();
```

Creates empty attribute store. Object is not associated with a persistence storage.

## 1.2. AttributeStore(IAttributeStore)

```
public AttributeStore(org.red5.server.api.IAttributeStore values);
```

Creates attribute store with initial values. Object is not associated with a persistence storage.

## 1.3. AttributeStore(Map<String, Object>)

```
public AttributeStore(java.util.Map<java.lang.String, java.lang.Object> values);
```

Creates attribute store with initial values. Object is not associated with a persistence storage.

## 1.4. attributes

```
protected java.util.concurrent.ConcurrentMap<java.lang.String, java.lang.Object> attributes;
```

Map for attributes

## 1.5. filterNull(Map<String, Object>)

```
protected java.util.Map<java.lang.String, java.lang.Object> filterNull(java.util.Map<java.lang.String, java.lang.Object> values);
```

Filter null keys and values from given map.

### Parameters

values	the map to filter
--------	-------------------

<i>return</i>	filtered map
---------------	--------------

## 1.6. **getAttribute(String)**

```
public Object getAttribute(String name);
```

Return the value for a given attribute.

### Parameters

name	the name of the attribute to get
<i>return</i>	the attribute value or null if the attribute doesn't exist

## 1.7. **getAttribute(String, Object)**

```
public Object getAttribute(String name,
                           Object defaultValue);
```

Return the value for a given attribute and set it if it doesn't exist.

### Parameters

name	the name of the attribute to get
defaultValue	the value of the attribute to set if the attribute doesn't exist
<i>return</i>	the attribute value

## 1.8. **getAttributeNames()**

```
public java.util.Set<java.lang.String> getAttributeNames();
```

Get the attribute names. The resulting set will be read-only.

### Parameters

<i>return</i>	set containing all attribute names
---------------	------------------------------------

## 1.9. **getAttributes()**

```
public java.util.Map<java.lang.String, java.lang.Object> getAttributes();
```

Get the attributes. The resulting map will be read-only.

### Parameters

<i>return</i>	map containing all attributes
---------------	-------------------------------

## 1.10. **getBoolAttribute(String)**

```
public Boolean getBoolAttribute(String name);
```

**Specified by:** Method getBoolAttribute in interface ICastingAttributeStore

Get Boolean attribute by name

#### Parameters

name	Attribute name
<i>return</i>	Attribute

## 1.11. getByteAttribute(String)

```
public Byte getByteAttribute(String name);
```

**Specified by:** Method getByteAttribute in interface ICastingAttributeStore

Get Byte attribute by name

#### Parameters

name	Attribute name
<i>return</i>	Attribute

## 1.12. getDoubleAttribute(String)

```
public Double getDoubleAttribute(String name);
```

**Specified by:** Method getDoubleAttribute in interface ICastingAttributeStore

Get Double attribute by name

#### Parameters

name	Attribute name
<i>return</i>	Attribute

## 1.13. getIntAttribute(String)

```
public Integer getIntAttribute(String name);
```

**Specified by:** Method getIntAttribute in interface ICastingAttributeStore

Get Integer attribute by name

#### Parameters

name	Attribute name
<i>return</i>	Attribute

## 1.14. getListAttribute(String)

```
public java.util.List getListAttribute(String name);
```

**Specified by:** Method getListAttribute in interface ICastingAttributeStore

Get List attribute by name

### Parameters

name	Attribute name
<i>return</i>	Attribute

## 1.15. getLongAttribute(String)

```
public Long getLongAttribute(String name);
```

**Specified by:** Method getLongAttribute in interface ICastingAttributeStore

Get boolean attribute by name

### Parameters

name	Attribute name
<i>return</i>	Attribute

## 1.16. getMapAttribute(String)

```
public java.util.Map getMapAttribute(String name);
```

**Specified by:** Method getMapAttribute in interface ICastingAttributeStore

Get Long attribute by name

### Parameters

name	Attribute name
<i>return</i>	Attribute

## 1.17. getSetAttribute(String)

```
public java.util.Set getSetAttribute(String name);
```

**Specified by:** Method getSetAttribute in interface ICastingAttributeStore

Get Set attribute by name

### Parameters

name	Attribute name
------	----------------

<i>return</i>	Attribute
---------------	-----------

## 1.18. **getShortAttribute(String)**

```
public Short getShortAttribute(String name);
```

**Specified by:** Method `getShortAttribute` in interface `ICastingAttributeStore`

Get Short attribute by name

### Parameters

name	Attribute name
<i>return</i>	Attribute

## 1.19. **getStringAttribute(String)**

```
public String getStringAttribute(String name);
```

**Specified by:** Method `getStringAttribute` in interface `ICastingAttributeStore`

Get String attribute by name

### Parameters

name	Attribute name
<i>return</i>	Attribute

## 1.20. **hasAttribute(String)**

```
public boolean hasAttribute(String name);
```

Check the object has an attribute.

### Parameters

name	the name of the attribute to check
<i>return</i>	true if the attribute exists otherwise false

## 1.21. **removeAttribute(String)**

```
public boolean removeAttribute(String name);
```

Remove an attribute.

### Parameters

name	the name of the attribute to remove
------	-------------------------------------

<i>return</i>	true if the attribute was found and removed otherwise false
---------------	---

## 1.22. removeAttributes()

```
public void removeAttributes();
```

Remove all attributes.

## 1.23. setAttribute(String, Object)

```
public boolean setAttribute(String name,
                           Object value);
```

Set an attribute on this object.

### Parameters

<i>name</i>	the name of the attribute to change
<i>value</i>	the new value of the attribute
<i>return</i>	true if the attribute value changed otherwise false

## 1.24. setAttributes(IAttributeStore)

```
public void setAttributes(org.red5.server.api.IAttributeStore values);
```

Set multiple attributes on this object.

### Parameters

<i>values</i>	the attributes to set
---------------	-----------------------

## 1.25. setAttributes(Map<String, Object>)

```
public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);
```

Set multiple attributes on this object.

### Parameters

<i>values</i>	the attributes to set
---------------	-----------------------

## 2. Class BaseConnection

Base abstract class for connections. Adds connection specific functionality like work with clients to AttributeStore.

### 2.1. Synopsis

```
public abstract class BaseConnection extends, org.?red5.?server.?AttributeStore
implements, org.?red5.?server.?api.?IConnection {
```

## Package org.?red5.?server

```
// Protected Fields

protected java.util.Set<org.red5.server.api.IBasicScope> basicScopes ;

protected org.red5.server.api.IClient client ;

protected boolean closed ;

protected long droppedMessages ;

protected String host ;

protected static org.slf4j.Logger log ;

protected java.util.Map<java.lang.String, java.lang.Object> params ;

protected String path ;

protected long readMessages ;

protected String remoteAddress ;

protected java.util.List<java.lang.String> remoteAddresses ;

protected int remotePort ;

protected Scope scope ;

protected String sessionId ;

protected String type ;

protected long writtenMessages ;

// Public Constructors

public BaseConnection(String type,
                      String host,
                      String remoteAddress,
                      int remotePort,
                      String path,
                      String sessionId,
                      java.util.Map<java.lang.String, java.lang.Object> params);

// Public Methods

public void close();

public boolean connect(org.red5.server.api.IScope newScope);

public boolean connect(org.red5.server.api.IScope newScope,
                      Object[] params);

public void dispatchEvent(org.red5.server.api.event.IEvent event);

public java.util.Iterator<org.red5.server.api.IBasicScope> getBasicScopes();

public org.red5.server.api.IClient getClient();
```

```

public long getClientBytesRead();

public java.util.Map<java.lang.String, java.lang.Object> getConnectParams();

public long getDroppedMessages();

public String getHost();

public String getPath();

public long getPendingMessages();

public long getPendingVideoMessages(int streamId);

public abstract long getReadBytes();

public long getReadMessages();

public String getRemoteAddress();

public java.util.List<java.lang.String> getRemoteAddresses();

public int getRemotePort();

public org.red5.server.api.IScope getScope();

public String getSessionId();

public String getType();

public abstract long getWrittenBytes();

public long getWrittenMessages();

public boolean handleEvent(org.red5.server.api.event.IEvent event);

public void initialize(org.red5.server.api.IClient client);

public boolean isConnected();

public void notifyEvent(org.red5.server.api.event.IEvent event);

public void registerBasicScope(org.red5.server.api.IBasicScope basicScope);

public void unregisterBasicScope(org.red5.server.api.IBasicScope basicScope);

}

```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?RTMPConnection

**Methods inherited from org.red5.server.AttributeStore:** filterNull ,getAttribute ,getAttributeNames ,getAttributes ,getBoolAttribute ,getByteAttribute ,getDoubleAttribute ,getIntAttribute ,getListAttribute ,getLongAttribute ,getMapAttribute ,getSetAttribute ,getShortAttribute ,getStringAttribute ,hasAttribute ,removeAttribute ,removeAttributes ,setAttribute ,setAttributes

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 2.2. BaseConnection(String, String, String, int, String, String, Map<String, Object>)

```
public BaseConnection(String type,
                      String host,
                      String remoteAddress,
                      int remotePort,
                      String path,
                      String sessionId,
                      java.util.Map<java.lang.String, java.lang.Object> params);
```

Parameters	
type	Connection type
host	Host
remoteAddress	Remote address
remotePort	Remote port
path	Scope path on server
sessionId	Session id
params	Params passed from client

## 2.3. basicScopes

```
protected java.util.Set<org.red5.server.api.IBasicScope> basicScopes ;
```

Set of basic scopes.

## 2.4. client

```
protected org.red5.server.api.IClient client ;
```

Client bound to connection

## 2.5. closed

```
protected boolean closed ;
```

Is the connection closed?

## 2.6. droppedMessages

```
protected long droppedMessages ;
```

Number of dropped messages

## 2.7. host

```
protected String host ;
```

Connection host

## 2.8. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 2.9. params

```
protected java.util.Map<java.lang.String, java.lang.Object> params ;
```

See Also

NetConnection in Flash Media Server docs (external) [<http://livedocs.adobe.com/fms/2/docs/00000570.html>]

Connection params passed from client with NetConnection.connect call

## 2.10. path

```
protected String path ;
```

Path of scope client connected to

## 2.11. readMessages

```
protected long readMessages ;
```

Number of read messages

## 2.12. remoteAddress

```
protected String remoteAddress ;
```

Connection remote address

## 2.13. remoteAddresses

```
protected java.util.List<java.lang.String> remoteAddresses ;
```

Connection remote addresses

## 2.14. remotePort

```
protected int remotePort ;
```

Remote port

## 2.15. scope

```
protected Scope scope ;
```

Scope that connection belongs to

## 2.16. sessionId

```
protected String sessionId ;
```

Connection session identifier

## 2.17. type

```
protected String type ;
```

Connection type

## 2.18. writtenMessages

```
protected long writtenMessages ;
```

Number of written messages

## 2.19. close()

```
public void close();
```

**Specified by:** Method close in interface IConnection

Closes connection

## 2.20. connect(IScope)

```
public boolean connect(org.red5.server.api.IScope newScope);
```

**Specified by:** Method connect in interface IConnection

Connect to another scope on server

### Parameters

newScope	New scope
return	true on success, false otherwise

## 2.21. connect(IScope, Object[])

```
public boolean connect(org.red5.server.api.IScope newScope,
                      Object[] params);
```

**Specified by:** Method connect in interface IConnection

Connect to another scope on server with given parameters

### Parameters

newScope	New scope
----------	-----------

params	Parameters to connect with
<i>return</i>	true on success, false otherwise

## 2.22. dispatchEvent(IEvent)

```
public void dispatchEvent(org.red5.server.api.event.IEvent event);
```

Dispatches event

Parameters	
event	Event

## 2.23. getBasicScopes()

```
public java.util.Iterator<org.red5.server.api.IBasicScope> getBasicScopes();
```

**Specified by:** Method getBasicScopes in interface IConnection

Get the basic scopes this connection has subscribed. This list will contain the shared objects and broadcast streams the connection connected to.

Parameters	
<i>return</i>	List of basic scopes

**Description copied from interface: getBasicScopes**

## 2.24. getClient()

```
public org.red5.server.api.IClient getClient();
```

**Specified by:** Method getClient in interface IConnection

Get the client object associated with this connection.

Parameters	
<i>return</i>	Client object

**Description copied from interface: getClient**

## 2.25. getClientBytesRead()

```
public long getClientBytesRead();
```

**Specified by:** Method getClientBytesRead in interface IConnection

Return number of written bytes the client reports to have received. This is the last value of the BytesRead message received from a client.

## 2.26. getConnectParams()

```
public java.util.Map<java.lang.String, java.lang.Object> getConnectParams();
```

**Specified by:** Method `getConnectParams` in interface `IConnection`

Return connection parameters

Parameters

<i>return</i>	
---------------	--

## 2.27. `getDroppedMessages()`

<code>public long getDroppedMessages();</code>
--

**Specified by:** Method `getDroppedMessages` in interface `IConnection`

Total number of messages that have been dropped.

Parameters

<i>return</i>	Number of dropped messages
---------------	----------------------------

**Description copied from interface: `getDroppedMessages`**

## 2.28. `getHost()`

<code>public String getHost();</code>
---------------------------------------

**Specified by:** Method `getHost` in interface `IConnection`

Get the hostname that the client is connected to. If they are connected to an IP, the IP address will be returned as a String.

Parameters

<i>return</i>	String containing the hostname
---------------	--------------------------------

**Description copied from interface: `getHost`**

## 2.29. `getPath()`

<code>public String getPath();</code>
---------------------------------------

**Specified by:** Method `getPath` in interface `IConnection`

Get the path for this connection. This is not updated if you switch scope.

Parameters

<i>return</i>	path Connection path
---------------	----------------------

**Description copied from interface: `getPath`**

## 2.30. `getPendingMessages()`

<code>public long getPendingMessages();</code>
--

**Specified by:** Method `getPendingMessages` in interface `IConnection`

Total number of messages that are pending to be sent to the connection.

#### Parameters

<i>return</i>	Number of pending messages
---------------	----------------------------

**Description copied from interface: `getPendingMessages`**

### 2.31. `getPendingVideoMessages(int)`

```
public long getPendingVideoMessages(int streamId);
```

#### Parameters

streamId	
----------	--

<i>return</i>	
---------------	--

### 2.32. `getReadBytes()`

```
public abstract long getReadBytes();
```

**Specified by:** Method `getReadBytes` in interface `IConnection`

Total number of bytes read from the connection.

#### Parameters

<i>return</i>	Number of read bytes
---------------	----------------------

**Description copied from interface: `getReadBytes`**

### 2.33. `getReadMessages()`

```
public long getReadMessages();
```

**Specified by:** Method `getReadMessages` in interface `IConnection`

Total number of messages read from the connection.

#### Parameters

<i>return</i>	Number of read messages
---------------	-------------------------

**Description copied from interface: `getReadMessages`**

### 2.34. `getRemoteAddress()`

```
public String getRemoteAddress();
```

**Specified by:** Method `getRemoteAddress` in interface `IConnection`

Get the IP address the client is connected from.

**Parameters**

<i>return</i>	The IP address of the client
---------------	------------------------------

**Description copied from interface: getRemoteAddress****2.35. getRemoteAddresses()**

```
public java.util.List<java.lang.String> getRemoteAddresses();
```

**Specified by:** Method getRemoteAddresses in interface IConnection

Get the IP addresses the client is connected from. If a client is connected through RTMPT and uses a proxy to connect, this will contain all hosts the client used to connect to the server.

**Parameters**

<i>return</i>	The IP addresses of the client
---------------	--------------------------------

**Description copied from interface: getRemoteAddresses****2.36. getRemotePort()**

```
public int getRemotePort();
```

**Specified by:** Method getRemotePort in interface IConnection

Get the port the client is connected from.

**Parameters**

<i>return</i>	The port of the client
---------------	------------------------

**Description copied from interface: getRemotePort****2.37. getScope()**

```
public org.red5.server.api.IScope getScope();
```

**Specified by:** Method getScope in interface IConnection

Get the scope this is connected to.

**Parameters**

<i>return</i>	The connected scope
---------------	---------------------

**Description copied from interface: getScope****2.38. getSessionId()**

```
public String getSessionId();
```

**Specified by:** Method getSessionId in interface IConnection

Get the session id, this may be null.

#### Parameters

<i>return</i>	Session id
---------------	------------

#### Description copied from interface: **getSessionId**

### 2.39. **getType()**

```
public String getType();
```

**Specified by:** Method `getType` in interface `IConnection`

Get the connection type.

#### Parameters

<i>return</i>	string containing one of connection types
---------------	---

#### Description copied from interface: **getType**

### 2.40. **getWrittenBytes()**

```
public abstract long getWrittenBytes();
```

**Specified by:** Method `getWrittenBytes` in interface `IConnection`

Total number of bytes written to the connection.

#### Parameters

<i>return</i>	Number of written bytes
---------------	-------------------------

#### Description copied from interface: **getWrittenBytes**

### 2.41. **getWrittenMessages()**

```
public long getWrittenMessages();
```

**Specified by:** Method `getWrittenMessages` in interface `IConnection`

Total number of messages written to the connection.

#### Parameters

<i>return</i>	Number of written messages
---------------	----------------------------

#### Description copied from interface: **getWrittenMessages**

### 2.42. **handleEvent(IEvent)**

```
public boolean handleEvent(org.red5.server.api.event.IEvent event);
```

Handles event

**Parameters**

<i>event</i>	Event
<i>return</i>	true if associated scope was able to handle event, false otherwise

**2.43. initialize(IClient)**

```
public void initialize(org.red5.server.api.IClient client);
```

**Specified by:** Method initialize in interface IConnection

Initializes client

**Parameters**

<i>client</i>	Client bound to connection
---------------	----------------------------

**2.44. isConnected()**

```
public boolean isConnected();
```

**Specified by:** Method isConnected in interface IConnection

Check whether connection is alive

**Parameters**

<i>return</i>	true if connection is bound to scope, false otherwise
---------------	---

**2.45. notifyEvent(IEvent)**

```
public void notifyEvent(org.red5.server.api.event.IEvent event);
```

Notified on event

**Parameters**

<i>event</i>	Event
--------------	-------

**2.46. registerBasicScope(IBasicScope)**

```
public void registerBasicScope(org.red5.server.api.IBasicScope basicScope);
```

Registers basic scope

**Parameters**

<i>basicScope</i>	Basic scope to register
-------------------	-------------------------

**2.47. unregisterBasicScope(IBasicScope)**

```
public void unregisterBasicScope(org.red5.server.api.IBasicScope basicScope);
```

## Unregister basic scope

## Parameters

basicScope	Unregister basic scope
------------	------------------------

### 3. Class BasicScope

Generalizations of one of main Red5 object types, Scope. Basic scope is a persistable attribute store with event handling functionality

#### 3.1. Synopsis

```

public abstract class BasicScope extends, org.?red5.?server.?PersistableAttributeStore
    implements, org.?red5.?server.?api.?IBasicScope {
    // Protected Fields

        protected boolean keepOnDisconnect ;

        protected java.util.Set<org.red5.server.api.event.IEventListener> listeners ;

        protected org.red5.server.api.IScope parent ;

        protected String persistenceClass ;

    // Public Constructors

        public BasicScope(org.red5.server.api.IScope parent,
                         String type,
                         String name,
                         boolean persistent);

    // Public Methods

        public void addEventListener(org.red5.server.api.event.IEventListener listener);

        public void dispatchEvent(org.red5.server.api.event.IEvent event);

        public int getDepth();

        public java.util.Iterator<org.red5.server.api.event.IEventListener> getEventListeners();

        public org.red5.server.api.IScope getParent();

        public String getPath();

        public boolean handleEvent(org.red5.server.api.event.IEvent event);

        public boolean hasParent();

        public java.util.Iterator<org.red5.server.api.IBasicScope> iterator();

        public void notifyEvent(org.red5.server.api.event.IEvent event);

        public void removeEventListener(org.red5.server.api.event.IEventListener listener);

    }
}

```

**Direct known subclasses:** org.?red5.?server.?Scope , org.?red5.?server.?so.?SharedObjectScope , org.?red5.?server.?stream.?BroadcastScope

**Methods inherited from org.red5.server.PersistableAttributeStore:** deserialize , getAttribute , getLastModified , getName , getPath , getStore , getType , isPersistent , modified , removeAttribute , removeAttributes , serialize , setAttribute , setAttributes , setName , setPath , setPersistent , setStore

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute , getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute , getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute , hasAttribute

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.PersistableAttributeStore:** lastModified , name , path , persistent , store , type

**Fields inherited from org.red5.server.AttributeStore:** attributes

#### See Also

[org.red5.server.api.IScope](#) , [org.red5.server.Scope](#)

## 3.2. BasicScope(IScope, String, String, boolean)

```
public BasicScope(org.red5.server.api.IScope parent,
                  String type,
                  String name,
                  boolean persistent);
```

Constructor for basic scope

Parameters	
parent	Parent scope
type	Scope type
name	Scope name. Used to identify scopes in application, must be unique among scopes of one level
persistent	Whether scope is persistent

## 3.3. keepOnDisconnect

```
protected boolean keepOnDisconnect ;
```

Set to true to prevent the scope from being freed upon disconnect.

## 3.4. listeners

```
protected java.util.Set<org.red5.server.api.event.IEventListener> listeners ;
```

List of event listeners

### 3.5. parent

```
protected org.red5.server.api.IScope parent ;
```

*See Also*

[org.red5.server.api.IScope](#)

Parent scope. Scopes can be nested.

### 3.6. persistenceClass

```
protected String persistenceClass ;
```

Scope persistence storage type

### 3.7. addEventListener(IEventListener)

```
public void addEventListener(org.red5.server.api.event.IEventListener listener);
```

Add event listener to list of notified objects

**Parameters**

listener	Listening object
----------	------------------

### 3.8. dispatchEvent(IEvent)

```
public void dispatchEvent(org.red5.server.api.event.IEvent event);
```

Dispatches event (notifies all listeners)

**Parameters**

event	Event to dispatch
-------	-------------------

### 3.9. getDepth()

```
public int getDepth();
```

**Specified by:** Method `getDepth` in interface `IBasicScope`

Get the scopes depth, how far down the scope tree is it. The lowest depth is 0x00, the depth of Global scope. Application scope depth is 0x01. Room depth is 0x02, 0x03 and so forth.

### 3.10. getEventListeners()

```
public java.util.Iterator<org.red5.server.api.event.IEventListener> getEventListeners();
```

Return listeners list iterator

**Parameters**

<i>return</i>	Listeners list iterator
---------------	-------------------------

### 3.11. getParent()

```
public org.red5.server.api.IScope getParent();
```

**Specified by:** Method getParent in interface IBasicScope

Get this scopes parent.

### 3.12. getPath()

```
public String getPath();
```

**Specified by:** Method getPath in interface IBasicScope

Get the full absolute path. Eg. host/myapp/someroom.

### 3.13. handleEvent(IEvent)

```
public boolean handleEvent(org.red5.server.api.event.IEvent event);
```

Handles event. To be implemented in subclass realization

#### Parameters

event	Event context
<i>return</i>	Event handling result

### 3.14. hasParent()

```
public boolean hasParent();
```

**Specified by:** Method hasParent in interface IBasicScope

Does this scope have a parent? You can think of scopes as of tree items where scope may have a parent and children (child).

### 3.15. iterator()

```
public java.util.Iterator<org.red5.server.api.IBasicScope> iterator();
```

Getter for subscopes list iterator. Returns null because this is a base implementation

#### Parameters

<i>return</i>	Iterator for subscopes
---------------	------------------------

### 3.16. notifyEvent(IEvent)

```
public void notifyEvent(org.red5.server.api.event.IEvent event);
```

Notifies listeners on event. Current implementation is empty. To be implemented in subclass realization

**Parameters**

event	Event to broadcast
-------	--------------------

**3.17. removeEventListener(IEventListener)**

```
public void removeEventListener(org.red5.server.api.event.IEventListener listener);
```

Remove event listener from list of listeners

**Parameters**

listener	Listener to remove
----------	--------------------

**4. Class BasicScope.EmptyBasicScopeIterator**

Iterator for basic scope

**4.1. Synopsis**

```
public class BasicScope.EmptyBasicScopeIterator implements java.util.Iterator {
// Public Constructors

    public BasicScope.EmptyBasicScopeIterator();

// Public Methods

    public boolean hasNext();

    public org.red5.server.api.IBasicScope next();

    public void remove();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

**4.2. hasNext()**

```
public boolean hasNext();
```

**Specified by:** Method `hasNext` in interface `Iterator`

**4.3. next()**

```
public org.red5.server.api.IBasicScope next();
```

**Specified by:** Method `next` in interface `Iterator`

**4.4. remove()**

```
public void remove();
```

**Specified by:** Method `remove` in interface `Iterator`

## 5. Class Bootstrap

Boot-straps Red5 using the latest available jars found in `red5.home/lib` directory.

### 5.1. Synopsis

```
public class Bootstrap {
    // Public Constructors

    public Bootstrap();

    // Public Static Methods

    public static void launch(java.net.URLClassLoader loader);

    public static void main(String[] args)
        throws Exception;

}
```

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

## 6. Class Client

Client is an abstraction representing user connected to Red5 application. Clients are tied to connections and registered in ClientRegistry

### 6.1. Synopsis

```
public class Client extends org.?red5.?server.?AttributeStore
    implements org.?red5.?server.?api.?IClient, org.?red5.?server.?ClientMBean {
    // Protected Fields

    protected static final String PERMISSIONS = "_transient_red5_permissions";

    protected java.util.concurrent.ConcurrentMap<org.red5.server.api.IConnection, org.red5.server.api.IConnection> connections;
    protected long creationTime;
    protected String id;
    protected static org.slf4j.Logger log;
    protected ClientRegistry registry;

    // Public Constructors

    public Client();

    public Client(String id,
        ClientRegistry registry);
```

```

// Public Methods

    public void disconnect();

    public boolean equals(Object obj);

    public org.red5.server.api.IBandwidthConfigure getBandwidthConfigure();

    public java.util.Set<org.red5.server.api.IConnection> getConnections();

    public java.util.Set<org.red5.server.api.IConnection> getConnections(org.red5.server.api.IScope sc);

    public long getCreationTime();

    public String getId();

    public org.red5.server.api.IBWControllable getParentBWControllable();

    public java.util.Collection<java.lang.String> getPermissions(org.red5.server.api.IConnection conn);

    public java.util.Collection<org.red5.server.api.IScope> getScopes();

    public boolean hasPermission(org.red5.server.api.IConnection conn,
                                String permissionName);

    public int hashCode();

    public java.util.List<java.lang.String> iterateScopeNameList();

    public void setBandwidthConfigure(org.red5.server.api.IBandwidthConfigure config);

    public void setPermissions(org.red5.server.api.IConnection conn,
                             java.util.Collection<java.lang.String> permissions);

    public String toString();

// Protected Methods

    protected void register(org.red5.server.api.IConnection conn);

    protected void unregister(org.red5.server.api.IConnection conn);

}

```

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttribute ,  
getAttributeName , getAttributes , getBoolAttribute , getByteAttribute ,  
getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute ,  
getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute ,  
hasAttribute , removeAttribute , removeAttributes , setAttribute , setAttributes

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 6.2. Client(String, ClientRegistry)

```
public Client(String id,
              ClientRegistry registry);
```

Creates client, sets creation time and registers it in ClientRegistry

### Parameters

id	Client id
registry	ClientRegistry

## 6.3. connToScope

```
protected java.util.concurrent.ConcurrentMap<org.red5.server.api.IConnection, org.red5.server.api.IScope> connToScope;
```

Scopes this client connected to

## 6.4. creationTime

```
protected long creationTime ;
```

Creation time as Timestamp

## 6.5. id

```
protected String id ;
```

Clients identifier

## 6.6. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 6.7. PERMISSIONS

```
protected static final String PERMISSIONS = "_transient_red5_permissions";
```

Name of connection attribute holding the permissions.

## 6.8. registry

```
protected ClientRegistry registry ;
```

Client registry where Client is registered

## 6.9. disconnect()

```
public void disconnect();
```

**Specified by:** Method disconnect in interface IClient

Disconnects client from Red5 application

## 6.10. equals(Object)

```
public boolean equals(Object obj);
```

Check clients equality by id

### Parameters

obj	Object to check against
<i>return</i>	true if clients ids are the same, false otherwise

## 6.11. getBandwidthConfigure()

```
public org.red5.server.api.IBandwidthConfigure getBandwidthConfigure();
```

Return bandwidth configuration context, that is, broadcasting bandwidth and quality settings for this client

### Parameters

<i>return</i>	Bandwidth configuration context
---------------	---------------------------------

## 6.12. getConnections()

```
public java.util.Set<org.red5.server.api.IConnection> getConnections();
```

**Specified by:** Method getConnections in interface IClient

Return set of connections for this client

### Parameters

<i>return</i>	Set of connections
---------------	--------------------

## 6.13. getConnections(IScope)

```
public java.util.Set<org.red5.server.api.IConnection> getConnections(org.red5.server.api.IScope scope);
```

**Specified by:** Method getConnections in interface IClient

Return client connections to given scope

### Parameters

scope	Scope
<i>return</i>	Set of connections for that scope

## 6.14. getCreationTime()

```
public long getCreationTime();
```

**Specified by:** Method `getCreationTime` in interface `IClient`

Get the creation time for this client object.

#### Parameters

<code>return</code>	Creation time in milliseconds
---------------------	-------------------------------

**Description copied from interface: `getCreationTime`**

## 6.15. `getId()`

```
public String getId();
```

**Specified by:** Method `getId` in interface `IClient`

Get the unique ID for this client. This will be generated by the server if not passed upon connection from client-side Flex/Flash app. To assign a custom ID to the client use `params` object of `appConnect(org.red5.server.api.IConnection, java.lang.Object[])` method, that contains 2nd all the rest values you pass to `NetConnection.connect` method. Example:  
**At client side:** `NetConnection.connect( "http://localhost/killerapp/" , "user123" );`  
**then at server side:** `public boolean appConnect( IConnection connection, Object[] params ){ try { connection.getClient().setStreamId( params[0] ); } catch(Exception e){ log.error("{}", e); } }`

#### Parameters

<code>return</code>	client id
---------------------	-----------

**Description copied from interface: `getId`**

## 6.16. `getParentBWControllable()`

```
public org.red5.server.api.IBWControllable getParentBWControllable();
```

Parent flow controllable object, that is, parent object that is used to determine client broadcast bandwidth settings. In case of base Client class parent is host.

#### Parameters

<code>return</code>	IFlowControllable instance
---------------------	----------------------------

## 6.17. `getPermissions(IConnection)`

```
public java.util.Collection<java.lang.String> getPermissions(org.red5.server.api.IConnection conn);
```

**Specified by:** Method `getPermissions` in interface `IClient`

Return the permissions in a given context.

## 6.18. `getScopes()`

```
public java.util.Collection<org.red5.server.api.IScope> getScopes();
```

**Specified by:** Method getScopes in interface IClient

Get a set of scopes the client is connected to.

Parameters

<i>return</i>	Set of scopes
---------------	---------------

**Description copied from interface: getScopes**

## 6.19. hashCode()

```
public int hashCode();
```

if overriding equals then also do hashCode

Parameters

<i>return</i>	
---------------	--

## 6.20. hasPermission(IConnection, String)

```
public boolean hasPermission(org.red5.server.api.IConnection conn,
                           String permissionName);
```

**Specified by:** Method hasPermission in interface IClient

Check if the client has a permission in the given context.

## 6.21. iterateScopeNameList()

```
public java.util.List<java.lang.String> iterateScopeNameList();
```

**Specified by:** Method iterateScopeNameList in interface ClientMBean

Iterate through the scopes and their attributes. Used by JMX

Parameters

<i>return</i>	list of scope attributes
---------------	--------------------------

## 6.22. register(IConnection)

```
protected void register(org.red5.server.api.IConnection conn);
```

Associate connection with client

Parameters

conn	Connection object
------	-------------------

## 6.23. setBandwidthConfigure(IBandwidthConfigure)

```
public void setBandwidthConfigure(org.red5.server.api.IBandwidthConfigure config);
```

Set new bandwidth configuration context

#### Parameters

config	Bandwidth configuration context
--------	---------------------------------

## 6.24. setPermissions(IConnection, Collection<String>)

```
public void setPermissions(org.red5.server.api.IConnection conn,
                         java.util.Collection<java.lang.String> permissions);
```

**Specified by:** Method setPermissions in interface IClient

Set the permissions for this client in a given context.

## 6.25. toString()

```
public String toString();
```

#### Parameters

return
--------

## 6.26. unregister(IConnection)

```
protected void unregister(org.red5.server.api.IConnection conn);
```

Removes client-connection association for given connection

#### Parameters

conn	Connection object
------	-------------------

# 7. Class ClientList

```
public class ClientList<E> extends java.util.ArrayList
    implements org.red5.server.ListMBean {
// Public Constructors
    public ClientList();
}
```

**Methods inherited from java.util.ArrayList:** add , addAll , clear , clone , contains , ensureCapacity , get , indexOf , isEmpty , lastIndexOf , remove , removeRange , set , size , toArray , trimToSize

**Methods inherited from java.util.AbstractList:** equals , hashCode , iterator , listIterator , subList

**Methods inherited from java.util.AbstractCollection:** containsAll , removeAll , retainAll , toString

**Methods inherited from java.lang.Object:** finalize , getClass , notify , notifyAll , wait

Fields inherited from **java.util.AbstractList**: modCount

## 8. Interface ClientMBean

MBean for Client.

### 8.1. Synopsis

```
public interface ClientMBean {
    // Public Methods

    public void disconnect();

    public java.util.Set<org.red5.server.api.IConnection> getConnections();

    public long getCreationTime();

    public String getId();

    public java.util.List<java.lang.String> iterateScopeNameList();

}

}
```

## 9. Class ClientRegistry

Registry for clients. Associates client with it's id so it's possible to get client by id from whenever we need.

### 9.1. Synopsis

```
public class ClientRegistry implements, org.?red5.?server.?api.?IClientRegistry, org.?red5.?server.?Client
// Public Constructors

    public ClientRegistry();

    public ClientRegistry(String name);

// Public Methods

    public Client getClient(String id)
        throws ClientNotFoundException;

    public org.red5.server.ClientList<org.red5.server.Client> getClientList();

    public boolean hasClient(String id);

    public org.red5.server.api.IClient lookupClient(String id)
        throws ClientNotFoundException;

    public org.red5.server.api.IClient newClient(Object[] params)
        throws ClientNotFoundException, ClientRejectedException;
```

```

    public String nextId();

    public String previousId();

// Protected Methods

    protected void addClient(org.red5.server.api.IClient client);

    protected java.util.Collection<org.red5.server.api.IClient> getClients();

    protected boolean hasClients();

    protected void removeClient(org.red5.server.api.IClient client);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 9.2. addClient(IClient)

```
protected void addClient(org.red5.server.api.IClient client);
```

Add client to registry

### Parameters

client	Client to add
--------	---------------

## 9.3. getClientList()

```
public org.red5.server.ClientList<org.red5.server.Client> getClientList();
```

**Specified by:** Method getClientList in interface ClientRegistryMBean

Returns a list of Clients.

## 9.4. getClients()

```
protected java.util.Collection<org.red5.server.api.IClient> getClients();
```

Return collection of clients

### Parameters

return	Collection of clients
--------	-----------------------

## 9.5. hasClient(String)

```
public boolean hasClient(String id);
```

**Specified by:** Method hasClient in interface IClientRegistry

Check whether registry has client with given id

Parameters	
id	Client id
<i>return</i>	true if client with given id was register with this registry, false otherwise

## 9.6. hasClients()

```
protected boolean hasClients();
```

Check if client registry contains clients.

Parameters	
<i>return</i>	True if clients exist, otherwise False

## 9.7. lookupClient(String)

```
public org.red5.server.apiIClient lookupClient(String id)
throws ClientNotFoundException;
```

**Specified by:** Method lookupClient in interface IClientRegistry

Return client by id

Parameters	
id	Client id
<i>return</i>	Client object associated with given id

ClientNotFoundException

Client not found

## 9.8. newClient(Object[])

```
public org.red5.server.apiIClient newClient(Object[] params)
throws ClientNotFoundException, ClientRejectedException;
```

**Specified by:** Method newClient in interface IClientRegistry

Return client from next id with given params

Parameters	
params	Client params
<i>return</i>	Client object

ClientNotFoundException

Client not found

```
ClientRejectedException
The client is not allowed to connect.
```

## 9.9. nextId()

<code>public String nextId();</code>
--------------------------------------

**Specified by:** Method `nextId` in interface `ClientRegistryMBean`

Return next client id

### Parameters

<i>return</i>	Next client id
---------------	----------------

## 9.10. previousId()

<code>public String previousId();</code>
--

**Specified by:** Method `previousId` in interface `ClientRegistryMBean`

Return previous client id

### Parameters

<i>return</i>	Previous client id
---------------	--------------------

## 9.11. removeClient(IClient)

<code>protected void removeClient(org.red5.server.api.IClient client);</code>
---

Removes client from registry

### Parameters

<i>client</i>	Client to remove
---------------	------------------

# 10. Interface ClientRegistryMBean

An MBean interface for the client registry.

## 10.1. Synopsis

<code>public interface ClientRegistryMBean {</code>
<code>// Public Methods</code>
<code>    public Client getClient(String id)</code>
<code>    throws ClientNotFoundException;</code>
<code>    public java.util.List&lt;org.red5.server.Client&gt; getClientList();</code>
<code>    public boolean hasClient(String id);</code>
<code>    public String nextId();</code>

```
public String previousId();
}
```

## 11. Class Context

This is basic context implementation used by Red5.

### 11.1. Synopsis

```
public class Context implements, org.?red5.?server.?api.?IContext, org.?springframework.?context.?Appl
// Public Static Fields

    public static org.slf4j.Logger logger ;

// Public Constructors

    public Context();

    public Context(org.springframework.context.ApplicationContext context,
                  String contextPath);

// Public Methods

    public org.springframework.context.ApplicationContext getApplicationContext();

    public Object getBean(String beanId);

    public ClassLoader getClassLoader();

    public org.red5.server.api.IClientRegistry getClientRegistry();

    public Object getCoreService(String beanId);

    public org.red5.server.api.IGlobalScope getGlobalScope();

    public org.red5.server.api.IMappingStrategy getMappingStrategy();

    public org.red5.server.api.persistence.IPersistenceStore getPersistianceStore();

    public org.springframework.core.io.Resource getResource(String path);

    public org.springframework.core.io.Resource[] getResources(String pattern)
        throws IOException;

    public org.red5.server.api.IScope getScope();

    public org.red5.server.api.IScopeResolver getScopeResolver();

    public org.red5.server.api.service.IServiceInvoker getServiceInvoker();

    public org.red5.server.api.IScopeHandler lookupScopeHandler(String contextPath);

    public Object lookupService(String serviceName);
```

```

public org.red5.server.api.IScope resolveScope(String path);

public org.red5.server.api.IScope resolveScope(String host,
                                              String path);

public org.red5.server.api.IScope resolveScope(org.red5.server.api.IScope root,
                                               String path);

public void setApplicationContext(org.springframework.context.ApplicationContext context);

public void setClientRegistry(org.red5.server.api.IClientRegistry clientRegistry);

public void setContextPath(String contextPath);

public void setCoreBeanFactory(org.springframework.beans.factory.BeanFactory core);

public void setMappingStrategy(org.red5.server.api.IMappingStrategy mappingStrategy);

public void setPersistanceStore(org.red5.server.api.persistence.IPersistenceStore persistanceStore);

public void setScopeResolver(org.red5.server.api.IScopeResolver scopeResolver);

public void setServiceInvoker(org.red5.server.api.service.IServiceInvoker serviceInvoker);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

## 11.2. Context()

```
public Context();
```

Initializes core context bean factory using red5.core bean factory from red5.xml context

## 11.3. Context(ApplicationContext, String)

```
public Context(org.springframework.context.ApplicationContext context,
              String contextPath);
```

Initializes app context and context path from given parameters

### Parameters

context	Application context
contextPath	Context path

## 11.4. getApplicationContext()

```
public org.springframework.context.ApplicationContext getApplicationContext();
```

**Specified by:** Method getApplicationContext in interface IContext

Return application context

#### Parameters

<i>return</i>	App context
---------------	-------------

## 11.5. getBean(String)

```
public Object getBean(String beanId);
```

**Specified by:** Method getBean in interface IContext

Return bean instantiated by bean factory

#### Parameters

beanId	Bean name
<i>return</i>	Instantiated bean

See Also

[org.springframework.beans.factory.BeanFactory](#)

## 11.6. getClassLoader()

```
public ClassLoader getClassLoader();
```

Return current thread's context classloader

#### Parameters

<i>return</i>	Classloder context of current thread
---------------	--------------------------------------

## 11.7. getClientRegistry()

```
public org.red5.server.api.IClientRegistry getClientRegistry();
```

**Specified by:** Method getClientRegistry in interface IContext

Return client registry

#### Parameters

<i>return</i>	Client registry
---------------	-----------------

## 11.8. getCoreService(String)

```
public Object getCoreService(String beanId);
```

**Specified by:** Method getCoreService in interface IContext

Return core Red5 service instantiated by core context bean factory

#### Parameters

beanId	Bean name
<i>return</i>	Core Red5 service instantiated

See Also

[org.springframework.beans.factory.BeanFactory](#)

## 11.9. getGlobalScope()

```
public org.red5.server.api.IGlobalScope getGlobalScope();
```

**Specified by:** Method `getGlobalScope` in interface `IContext`

Return global scope

Parameters

<i>return</i>	Global scope
---------------	--------------

## 11.10. getMappingStrategy()

```
public org.red5.server.api.IMappingStrategy getMappingStrategy();
```

**Specified by:** Method `getMappingStrategy` in interface `IContext`

Return mapping strategy used by this context. Mapping strategy define naming rules (prefixes, postfixes, default application name, etc) for all named objects in context.

Parameters

<i>return</i>	Mapping strategy
---------------	------------------

## 11.11. getPersistanceStore()

```
public org.red5.server.api.persistence.IPersistenceStore getPersistanceStore();
```

**Specified by:** Method `getPersistanceStore` in interface `IContext`

Return persistence store

Parameters

<i>return</i>	Persistence store
---------------	-------------------

## 11.12. getResource(String)

```
public org.springframework.core.io.Resource getResource(String path);
```

**Specified by:** Method `getResource` in interface `ContextMBean`

Return resource by path

Parameters

path	Resource path
<i>return</i>	Resource

**See Also**[org.springframework.core.io.Resource](#)

## 11.13. getResources(String)

```
public org.springframework.core.io.Resource[] getResources(String pattern)
throws IOException;
```

**Specified by:** Method getResources in interface ContextMBean

Return array or resource that match given pattern

Parameters	
pattern	Pattern to check against
<i>return</i>	Array of Resource objects

IOException

On I/O exception

**See Also**[org.springframework.core.io.Resource](#)

## 11.14. getScope()

```
public org.red5.server.api.IScope getScope();
```

**Specified by:** Method getScope in interface ContextMBean

Return scope

Parameters	
<i>return</i>	null

## 11.15. getScopeResolver()

```
public org.red5.server.api.IScopeResolver getScopeResolver();
```

Return scope resolver

Parameters	
<i>return</i>	scope resolver

## 11.16. getServiceInvoker()

```
public org.red5.server.api.service.IServiceInvoker getServiceInvoker();
```

**Specified by:** Method getServiceInvoker in interface IContext

Return service invoker

#### Parameters

<i>return</i>	Service invoker
---------------	-----------------

## 11.17. lookupScopeHandler(String)

```
public org.red5.server.api.IScopeHandler lookupScopeHandler(String contextPath);
```

**Specified by:** Method lookupScopeHandler in interface IContext

Look up scope handler for context path

#### Parameters

contextPath	Context path
<i>return</i>	Scope handler

ScopeHandlerNotFoundException

If there's no handler for given context path

## 11.18. lookupService(String)

```
public Object lookupService(String serviceName);
```

**Specified by:** Method lookupService in interface IContext

Look up service by name

#### Parameters

serviceName	Service name
<i>return</i>	Service object

ServiceNotFoundException

When service found but null

NoSuchBeanDefinitionException

When bean with given name doesn't exist

## 11.19. resolveScope(IScope, String)

```
public org.red5.server.api.IScope resolveScope(org.red5.server.api.IScope root,
                                                String path);
```

**Specified by:** Method resolveScope in interface IContext

Resolves scope from given root using scope resolver.

#### Parameters

root	Scope to start from.
------	----------------------

path	Path to resolve.
<i>return</i>	Scope resolution result.

## 11.20. resolveScope(String)

```
public org.red5.server.api.IScope resolveScope(String path);
```

**Specified by:** Method resolveScope in interface IContext

Resolves scope using scope resolver collaborator

Parameters	
path	Path to resolve
<i>return</i>	Scope resolution result

## 11.21. resolveScope(String, String)

```
public org.red5.server.api.IScope resolveScope(String host,
                                              String path);
```

**Specified by:** Method resolveScope in interface ContextMBean

Resolve scope from host and path

Parameters	
host	Host
path	Path
<i>return</i>	Scope

See Also

[org.red5.server.api.IScope](#) , [org.red5.server.Scope](#)

## 11.22. setApplicationContext(ApplicationContext)

```
public void setApplicationContext(org.springframework.context.ApplicationContext context);
```

**Specified by:** Method setApplicationContext in interface ApplicationContextAware

Setter for application context

Parameters	
context	App context

## 11.23. setClientRegistry(IClientRegistry)

```
public void setClientRegistry(org.red5.server.api.IClientRegistry clientRegistry);
```

Setter for client registry

## Parameters

clientRegistry	Client registry
----------------	-----------------

**11.24. setContextPath(String)**

```
public void setContextPath(String contextPath);
```

**Specified by:** Method setContextPath in interface ContextMBean

Setter for context path. Adds slash at the end of path if there's no one

## Parameters

contextPath	Context path
-------------	--------------

**11.25. setMappingStrategy(IMappingStrategy)**

```
public void setMappingStrategy(org.red5.server.api.IMappingStrategy mappingStrategy);
```

Setter for mapping strategy

## Parameters

mappingStrategy	Mapping strategy
-----------------	------------------

**11.26. setPersistanceStore(IPersistenceStore)**

```
public void setPersistanceStore(org.red5.server.api.persistence.IPersistenceStore persistanceStore);
```

Setter for persistence store

## Parameters

persistanceStore	Persistence store
------------------	-------------------

**11.27. setScopeResolver(IScopeResolver)**

```
public void setScopeResolver(org.red5.server.api.IScopeResolver scopeResolver);
```

Setter for scope resolver

## Parameters

scopeResolver	Scope resolver used to resolve scopes
---------------	---------------------------------------

**11.28. setServiceInvoker(IServiceInvoker)**

```
public void setServiceInvoker(org.red5.server.api.service.IServiceInvoker serviceInvoker);
```

Setter for service invoker

## Parameters

serviceInvoker

Service invoker object

## 12. Class ContextLoader

Red5 applications loader

### 12.1. Synopsis

```

public class ContextLoader implements, org.?springframework.?context.?ApplicationContextAware, org.?re
// Protected Fields

    protected org.springframework.context.ApplicationContext applicationContext ;

    protected java.util.concurrent.ConcurrentMap<java.lang.String, org.springframework.context.Applica
    protected String contextsConfig ;

    protected static org.slf4j.Logger log ;

    protected org.springframework.context.ApplicationContext parentContext ;

// Public Constructors

    public ContextLoader();

// Public Methods

    public org.springframework.context.ApplicationContext getContext(String name);

    public String getContextsConfig();

    public org.springframework.context.ApplicationContext getParentContext();

    public void init()
        throws Exception;

    public void loadContext(String name,
                          String config);

    public void setApplicationContext(org.springframework.context.ApplicationContext applicationContext
        throws BeansException;

    public void setContextsConfig(String contextsConfig);

    public void setParentContext(org.springframework.context.ApplicationContext parentContext);

    public void uninit();

    public void unloadContext(String name);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 12.2. applicationContext

```
protected org.springframework.context.ApplicationContext applicationContext ;
```

Spring Application context

## 12.3. contextMap

```
protected java.util.concurrent.ConcurrentMap<java.lang.String, org.springframework.context.ApplicationContext> contextMap ;
```

Context map

## 12.4. contextsConfig

```
protected String contextsConfig ;
```

Context location files

## 12.5. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 12.6. parentContext

```
protected org.springframework.context.ApplicationContext parentContext ;
```

Spring parent app context

## 12.7. getContext(String)

```
public org.springframework.context.ApplicationContext getContext(String name);
```

**Specified by:** Method getContext in interface ContextLoaderMBean

Return context by name

### Parameters

name	Context name
return	Application context for given name

## 12.8. getParentContext()

```
public org.springframework.context.ApplicationContext getParentContext();
```

**Specified by:** Method getParentContext in interface ContextLoaderMBean

Return parent context

### Parameters

<code>return</code>	parent application context
---------------------	----------------------------

## 12.9. init()

```
public void init()
    throws Exception;
```

**Specified by:** Method init in interface ContextLoaderMBean

Loads context settings from ResourceBundle (.properties file)

`Exception`

I/O exception, casting exception and others

## 12.10. loadContext(String, String)

```
public void loadContext(String name,
    String config);
```

**Specified by:** Method loadContext in interface ContextLoaderMBean

Loads a context (Red5 application) and stores it in a context map, then adds its beans to parent (that is, Red5)

### Parameters

<code>name</code>	Context name
<code>config</code>	Filename

## 12.11. setApplicationContext(ApplicationContext)

```
public void setApplicationContext(org.springframework.context.ApplicationContext applicationContext)
    throws BeansException;
```

**Specified by:** Method setApplicationContext in interface ApplicationContextAware

### Parameters

<code>applicationContext</code>	Spring application context
---------------------------------	----------------------------

`BeansException`

Top level exception for app context (that is, in fact, beans factory)

## 12.12. setContextsConfig(String)

```
public void setContextsConfig(String contextsConfig);
```

**Specified by:** Method setContextsConfig in interface ContextLoaderMBean

Setter for context config name

### Parameters

<code>contextsConfig</code>	Context config name
-----------------------------	---------------------

## 12.13. setParentContext(ApplicationContext)

```
public void setParentContext(org.springframework.context.ApplicationContext parentContext);
```

Setter for parent application context

### Parameters

parentContext	Parent Spring application context
---------------	-----------------------------------

## 12.14. unloadContext(String)

```
public void unloadContext(String name);
```

**Specified by:** Method `unloadContext` in interface `ContextLoaderMBean`

Unloads a context (Red5 application) and removes it from the context map, then removes its beans from the parent (that is, Red5)

### Parameters

name	Context name
------	--------------

# 13. Interface ContextLoaderMBean

Red5 applications loader

## 13.1. Synopsis

```
public interface ContextLoaderMBean {
    // Public Methods

    public org.springframework.context.ApplicationContext getContext(String name);

    public String getContextsConfig();

    public org.springframework.context.ApplicationContext getParentContext();

    public void init()
        throws Exception;

    public void loadContext(String name,
                          String config);

    public void setContextsConfig(String contextsConfig);

    public void uninit();

    public void unloadContext(String name);
}
```

# 14. Interface ContextMBean

This is basic context implementation used by Red5.

## 14.1. Synopsis

```

public interface ContextMBean {
// Public Methods

    public org.springframework.context.ApplicationContext getApplicationContext();

    public Object getBean(String beanId);

    public org.red5.server.api.IClientRegistry getClientRegistry();

    public Object getCoreService(String beanId);

    public org.red5.server.api.IScope getGlobalScope();

    public org.red5.server.api.IMappingStrategy getMappingStrategy();

    public org.red5.server.api.persistence.IPersistenceStore getPersistenceStore();

    public org.springframework.core.io.Resource getResource(String path);

    public org.springframework.core.io.Resource[] getResources(String pattern)
        throws IOException;

    public org.red5.server.api.IScope getScope();

    public org.red5.server.api.service.IServiceInvoker getServiceInvoker();

    public org.red5.server.api.IScopeHandler lookupScopeHandler(String contextPath);

    public Object lookupService(String serviceName);

    public org.red5.server.api.IScope resolveScope(String path);

    public org.red5.server.api.IScope resolveScope(String host,
                                                String path);

    public org.red5.server.api.IScope resolveScope(org.red5.server.api.IScope root,
                                                String path);

    public void setContextPath(String contextPath);
}

```

## 15. Class CoreHandler

Base IScopeHandler implementation

### 15.1. Synopsis

```

public class CoreHandler implements org.?red5.?server.?api.?IScopeHandler, org.?red5.?server.?CoreHan
// Protected Fields

```

```

protected static org.slf4j.Logger log ;

// Public Constructors

public CoreHandler();

// Public Methods

public boolean addChildScope(org.red5.server.api.IBasicScope scope);

public boolean connect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope);

public boolean connect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope,
                      Object[] params);

public void disconnect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope);

public boolean handleEvent(org.red5.server.api.event.IEvent event);

public boolean join(org.red5.server.api.IClient client,
                   org.red5.server.api.IScope scope);

public void leave(org.red5.server.api.IClient client,
                  org.red5.server.api.IScope scope);

public void removeChildScope(org.red5.server.api.IBasicScope scope);

public boolean serviceCall(org.red5.server.api.IConnection conn,
                           org.red5.server.api.service.IServiceCall call);

public boolean start(org.red5.server.api.IScope scope);

public void stop(org.red5.server.api.IScope scope);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 15.2. addChildScope(IBasicScope)

```
public boolean addChildScope(org.red5.server.api.IBasicScope scope);
```

**Specified by:** Method addChildScope in interface IScopeHandler

Called just before a child scope is added.

## 15.3. connect(IConnection, IScope)

```
public boolean connect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope);
```

**Specified by:** Method connect in interface CoreHandlerMBean

Connects client to the scope

#### Parameters

<i>conn</i>	Client conneciton
<i>scope</i>	Scope
<i>return</i>	true if client was registered within scope, false otherwise

### 15.4. connect(IConnection, IScope, Object[])

```
public boolean connect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope,
                      Object[] params);
```

**Specified by:** Method connect in interface IScopeHandler

Connects client to the scope

#### Parameters

<i>conn</i>	Client conneciton
<i>scope</i>	Scope
<i>params</i>	Params passed from client side with connect call
<i>return</i>	true if client was registered within scope, false otherwise

### 15.5. disconnect(IConnection, IScope)

```
public void disconnect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope);
```

**Specified by:** Method disconnect in interface IScopeHandler

Called just after the a connection is disconnected.

### 15.6. handleEvent(IEvent)

```
public boolean handleEvent(org.red5.server.api.event.IEvent event);
```

**Specified by:** Method handleEvent in interface CoreHandlerMBean

### 15.7. join(IClient, IScope)

```
public boolean join(org.red5.server.api.IClient client,
                   org.red5.server.api.IScope scope);
```

**Specified by:** Method join in interface IScopeHandler

Called just before a client enters the scope.

### 15.8. leave(IClient, IScope)

```
public void leave(org.red5.server.api.IClient client,
```

```
org.red5.server.api.IScope scope);
```

**Specified by:** Method leave in interface IScopeHandler

Called just after the client leaves the scope.

## 15.9. removeChildScope(IBasicScope)

```
public void removeChildScope(org.red5.server.api.IBasicScope scope);
```

**Specified by:** Method removeChildScope in interface IScopeHandler

Called just after a child scope has been removed.

## 15.10. serviceCall(IConnection, IServiceCall)

```
public boolean serviceCall(org.red5.server.api.IConnection conn,
                           org.red5.server.api.service(IServiceCall call);
```

**Specified by:** Method serviceCall in interface IScopeHandler

Remote method invocation

### Parameters

conn	Connection to invoke method on
call	Service call context
<i>return</i>	true on success

## 15.11. start(IScope)

```
public boolean start(org.red5.server.api.IScope scope);
```

**Specified by:** Method start in interface IScopeHandler

Called when a scope is created for the first time.

## 15.12. stop(IScope)

```
public void stop(org.red5.server.api.IScope scope);
```

**Specified by:** Method stop in interface IScopeHandler

Called just before a scope is disposed.

# 16. Interface CoreHandlerMBean

Base IScopeHandler implementation

## 16.1. Synopsis

```
public interface CoreHandlerMBean {
```

```
// Public Methods

    public boolean connect(org.red5.server.api.IConnection conn,
                          org.red5.server.api.IScope scope);

    public boolean connect(org.red5.server.api.IConnection conn,
                          org.red5.server.api.IScope scope,
                          Object[] params);

    public void disconnect(org.red5.server.api.IConnection conn,
                          org.red5.server.api.IScope scope);

    public boolean handleEvent(org.red5.server.api.event.IEvent event);

    public boolean join(org.red5.server.api.IClient client,
                       org.red5.server.api.IScope scope);

    public void leave(org.red5.server.api.IClient client,
                     org.red5.server.api.IScope scope);

    public void removeChildScope(org.red5.server.api.IBasicScope scope);

    public boolean serviceCall(org.red5.server.api.IConnection conn,
                              org.red5.server.api.service.IServiceCall call);

    public boolean start(org.red5.server.api.IScope scope);

    public void stop(org.red5.server.api.IScope scope);

}
```

## 17. Class DebugPooledByteBufferAllocator

A org.apache.mina.common.ByteBufferAllocator which pools allocated buffers.

All buffers are allocated with the size of power of 2 (e.g. 16, 32, 64, ...) This means that you cannot simply assume that the actual capacity of the buffer and the capacity you requested are same.

This allocator releases the buffers which have not been in use for a certain period. You can adjust the period by calling `setTimeout(int)`. The default timeout is 1 minute (60 seconds). To release these buffers periodically, a daemon thread is started when a new instance of the allocator is created. You can stop the thread by calling `dispose()`.

### 17.1. Synopsis

```
public class DebugPooledByteBufferAllocator implements org.apache.mina.common.ByteBufferAllocator
// Protected Fields

    protected static ThreadLocal local ;

    protected static org.slf4j.Logger log ;

    protected boolean saveStacks ;
```

```

protected java.util.HashMap<org.red5.server.DebugPooledByteBufferAllocator.UnexpandableByteBuffer, Object> stacks = new HashMap<org.red5.server.DebugPooledByteBufferAllocator.UnexpandableByteBuffer, Object>();

// Public Constructors

public DebugPooledByteBufferAllocator() {
}

public DebugPooledByteBufferAllocator(boolean saveStacks) {
}

public DebugPooledByteBufferAllocator(int timeout) {
}

public DebugPooledByteBufferAllocator(int timeout,
                                      boolean saveStacks) {
}

// Public Static Methods

public static String getCodeSection() {
}

public static void setCodeSection(String section) {
}

// Public Methods

public org.apache.mina.common.ByteBuffer allocate(int capacity,
                                                    boolean direct) {
}

public void dispose() {
}

public int getTimeout() {
}

public long getTimeoutMillis() {
}

public void printStacks() {
}

public void resetStacks() {
}

public void setTimeout(int timeout) {
}

public org.apache.mina.common.ByteBuffer wrap(java.nio.ByteBuffer nioBuffer) {
}

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 17.2. DebugPooledByteBufferAllocator()

```
public DebugPooledByteBufferAllocator();
```

Creates a new instance with the default timeout.

## 17.3. DebugPooledByteBufferAllocator(boolean)

```
public DebugPooledByteBufferAllocator(boolean saveStacks);
```

### Parameters

saveStacks	
------------	--

## 17.4. DebugPooledByteBufferAllocator(int)

public DebugPooledByteBufferAllocator(int timeout);
---

Creates a new instance with the specified `timeout`.

Parameters	
------------	--

timeout	
---------	--

## 17.5. DebugPooledByteBufferAllocator(int, boolean)

public DebugPooledByteBufferAllocator(int timeout, boolean saveStacks);
--

Parameters	
------------	--

timeout	
---------	--

saveStacks	
------------	--

## 17.6. local

protected static ThreadLocal local ;
--------------------------------------

## 17.7. log

protected static org.slf4j.Logger log ;
---

Logger

## 17.8. saveStacks

protected boolean saveStacks ;
--------------------------------

Save a stack trace for every buffer allocated? Warning: This slows down the Red5 a lot!

## 17.9. stacks

protected java.util.HashMap<org.red5.server.DebugPooledByteBufferAllocator.UnexpandableByteBuffer,
--

Contains stack traces where buffers were allocated.

## 17.10. allocate(int, boolean)

public org.apache.mina.common.ByteBuffer allocate(int capacity, boolean direct);
---

**Specified by:** Method `allocate` in interface `ByteBufferAllocator`

Parameters	
------------	--

capacity	
direct	
<i>return</i>	

## 17.11. dispose()

```
public void dispose();
```

**Specified by:** Method `dispose` in interface `ByteBufferAllocator`

Stops the thread which releases unused buffers and make this allocator unusable from now on.

## 17.12. getCodeSection()

```
public static String getCodeSection();
```

Parameters

*return*

## 17.13. getTimeout()

```
public int getTimeout();
```

Returns the timeout value of this allocator in seconds.

Parameters

*return*

## 17.14. getTimeoutMillis()

```
public long getTimeoutMillis();
```

Returns the timeout value of this allocator in milliseconds.

Parameters

*return*

## 17.15. setCodeSection(String)

```
public static void setCodeSection(String section);
```

Parameters

*section*

## 17.16. setTimeout(int)

```
public void setTimeout(int timeout);
```

Sets the timeout value of this allocator in seconds.

Parameters	
timeout	0 or negative value to disable timeout.

## 17.17. wrap(ByteBuffer)

```
public org.apache.mina.common.ByteBuffer wrap(java.nio.ByteBuffer nioBuffer);
```

**Specified by:** Method `wrap` in interface `ByteBufferAllocator`

Parameters	
nioBuffer	
<i>return</i>	

# 18. Class GlobalScope

Global scope is a top level scope. Server instance is meant to be injected with Spring before initialization (otherwise NullPointerException is thrown).

## 18.1. Synopsis

```
public class GlobalScope extends, org.?red5.?server.?Scope
    implements, org.?red5.?server.?api.?IGlobalScope {
// Protected Fields

    protected org.red5.server.api.IServer server ;

// Public Constructors

    public GlobalScope();

// Public Methods

    public org.red5.server.api.IServer getServer();

    public org.red5.server.api.persistence.IPersistenceStore getStore();

    public void register();

    public void setPersistenceClass(String persistenceClass)
        throws Exception;

    public void setServer(org.red5.server.api.IServer server);

}
```

**Methods inherited from org.red5.server.Scope:** addChildScope , connect , createChildScope , destroy , disconnect , dispatchEvent , getActiveClients , getActiveConnections , getActiveSubscopes , getBasicScope , getBasicScopeNames , getClassLoader , getClients , getConnections , getContext , getContextPath , getCreationTime , getDepth , getEnabled , getHandler , getMaxClients , getMaxConnections , getMaxSubscopes , getParent , getPath , getResource ,

```
getResources , getRunning , getScope , getScopeNames , getServer , getServiceHandler
, getServiceHandlerNames , getServiceHandlers , getStatistics , getTotalClients ,
getTotalConnections , getTotalSubscopes , handleEvent , hasChildScope , hasContext ,
hasHandler , hasParent , init , isEnabled , isRunning , iterator , lookupConnections
, registerServiceHandler , removeChildScope , setAutoStart , setChildLoadPath
, setContext , setDepth , setEnabled , setHandler , setName , setParent ,
setPersistenceClass , start , stop , toString , uninit , unregisterServiceHandler
```

**Methods inherited from org.red5.server.BasicScope:** addEventListener ,  
getEventListeners , notifyEvent , removeEventListener

**Methods inherited from org.red5.server.PersistableAttributeStore:** deserialize  
, getAttribute , getLastModified , getName , getStore , getType , isPersistent  
, modified , removeAttribute , removeAttributes , serialize , setAttribute ,  
setAttributes , setPath , setPersistent , setStore

**Methods inherited from org.red5.server.AttributeStore:** filterNull ,  
getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute ,  
getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute ,  
getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute ,  
hasAttribute

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , wait

**Fields inherited from org.red5.server.Scope:** clientStats , connectionStats , log ,  
oName , subscopeStats

**Fields inherited from org.red5.server.BasicScope:** keepOnDisconnect , listeners ,  
parent , persistenceClass

**Fields inherited from org.red5.server.PersistableAttributeStore:** lastModified , name  
, path , persistent , store , type

**Fields inherited from org.red5.server.AttributeStore:** attributes

#### See Also

`org.red5.server.api.IGlobalScope` , `org.red5.server.api.IScope`

## 18.2. getServer()

```
public org.red5.server.api.IServer getServer();
```

**Specified by:** Method `getServer` in interface `IGlobalScope`

Return the server this global scope runs in.

## 18.3. getStore()

```
public org.red5.server.api.persistence.IPersistenceStore getStore();
```

Get persistence store for scope

**Parameters**

<i>return</i>	Persistence store
---------------	-------------------

**18.4. register()**

```
public void register();
```

**Specified by:** Method register in interface IGlobalScope

Register global scope in server instance, then call initialization

**18.5. setPersistenceClass(String)**

```
public void setPersistenceClass(String persistenceClass)
    throws Exception;
```

**Parameters**

persistenceClass	Persistent class name
------------------	-----------------------

Exception

Exception

**18.6. setServer(I Server)**

```
public void setServer(org.red5.server.api.I Server server);
```

Setter for server

**Parameters**

server	Server
--------	--------

**19. Interface ListMBean**

```
public interface ListMBean {
// Public Methods

    public int size();

}
```

**20. Class LoaderBase**

Base class for all J2EE application loaders.

**20.1. Synopsis**

```
public class LoaderBase implements, org.?springframework.?context.?ApplicationContextAware {
// Public Static Fields
```

```

public static java.util.Map<java.lang.String, org.red5.server.api.IApplicationContext> red5AppCtx

// Protected Fields

protected static org.springframework.context.ApplicationContext applicationContext ;

protected static ThreadLocal<org.red5.server.api.IApplicationLoader> loader ;

protected String webappFolder ;

// Public Constructors

public LoaderBase();

// Public Static Methods

public static org.springframework.context.ApplicationContext getApplicationContext();

public static org.red5.server.api.IApplicationLoader getApplicationLoader();

public static org.red5.server.api.IApplicationContext getRed5ApplicationContext(String path);

public static org.red5.server.api.IApplicationContext removeRed5ApplicationContext(String path);

public static void setApplicationLoader(org.red5.server.api.IApplicationLoader loader);

public static void setRed5ApplicationContext(String path,
                                         org.red5.server.api.IApplicationContext context);

// Public Methods

public void removeContext(String path);

public void setApplicationContext(org.springframework.context.ApplicationContext context)
    throws BeansException;

public void setWebappFolder(String webappFolder);

}

```

**Direct known subclasses:** org.?red5.?server.?jetty.?JettyLoader , org.?red5.?server.?tomcat.?TomcatLoader

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 20.2. applicationContext

```
protected static org.springframework.context.ApplicationContext applicationContext ;
```

We store the application context so we can access it later.

## 20.3. loader

```
protected static ThreadLocal<org.red5.server.api.IApplicationLoader> loader ;
```

Loader for new applications.

## 20.4. red5AppCtx

```
public static java.util.Map<java.lang.String, org.red5.server.api.IApplicationContext> red5AppCtx ;
```

Current Red5 application context, set by the different loaders.

## 20.5. webappFolder

```
protected String webappFolder ;
```

Folder containing the webapps.

## 20.6. getApplicationContext()

```
public static org.springframework.context.ApplicationContext getApplicationContext();
```

Getter for application context

### Parameters

<i>return</i>	Application context
---------------	---------------------

## 20.7. getApplicationLoader()

```
public static org.red5.server.api.IApplicationLoader getApplicationLoader();
```

Getter for the application loader.

### Parameters

<i>return</i>	Application loader
---------------	--------------------

## 20.8. getRed5ApplicationContext(String)

```
public static org.red5.server.api.IApplicationContext getRed5ApplicationContext(String path);
```

Getter for a Red5 application context.

### Parameters

<i>return</i>	Red5 application context
---------------	--------------------------

## 20.9. removeContext(String)

```
public void removeContext(String path);
```

Remove context from the current host.

### Parameters

<i>path</i>	Path
-------------	------

## 20.10. removeRed5ApplicationContext(String)

```
public static org.red5.server.api.IApplicationContext removeRed5ApplicationContext(String path);
```

Remover for a Red5 application context.

### Parameters

<i>return</i>	Red5 application context
---------------	--------------------------

## 20.11. setApplicationContext(ApplicationContext)

```
public void setApplicationContext(org.springframework.context.ApplicationContext context)
throws BeansException;
```

**Specified by:** Method `setApplicationContext` in interface `ApplicationContextAware`

Setter for application context.

### Parameters

<i>context</i>	Application context
----------------	---------------------

`BeansException`

Abstract superclass for all exceptions thrown in the beans package and subpackages

## 20.12. setApplicationLoader(IApplicationLoader)

```
public static void setApplicationLoader(org.red5.server.api.IApplicationLoader loader);
```

Setter for the application loader.

### Parameters

<i>loader</i>	Application loader
---------------	--------------------

## 20.13. setRed5ApplicationContext(String, IApplicationContext)

```
public static void setRed5ApplicationContext(String path,
org.red5.server.api.IApplicationContext context);
```

Setter for a Red5 application context.

### Parameters

<i>context</i>	Red5 application context
----------------	--------------------------

## 20.14. setWebappFolder(String)

```
public void setWebappFolder(String webappFolder);
```

Set the folder containing webapps.

### Parameters

webappFolder	
--------------	--

## 21. Interface LoaderMBean

Simple mbean interface for J2EE container loaders. Allows for init and shutdown.

### 21.1. Synopsis

```
public interface LoaderMBean {
    // Public Methods

    public void init();

    public void removeContext(String path);

    public void shutdown();

}
```

## 22. Class MappingStrategy

Basic mapping strategy implementation. This one uses slash as filesystem path separator, '.service' postfix for services naming, '.handler' for handlers naming and 'default' string as default application name.

### 22.1. Synopsis

```
public class MappingStrategy implements org.?red5.?server.?api.?IMappingStrategy {
    // Public Constructors

    public MappingStrategy();

    // Public Methods

    public String mapResourcePrefix(String path);

    public String mapScopeHandlerName(String path);

    public String mapServiceName(String name);

    public void setDefaultApp(String defaultApp);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 22.2. mapResourcePrefix(String)

public String mapResourcePrefix(String path);
---

**Specified by:** Method mapResourcePrefix in interface IMappingStrategy

Resolves resource prefix from path. Default application used as root when path is specified

#### Parameters

path	Path
<i>return</i>	Resource prefix according to this naming strategy

### 22.3. mapScopeHandlerName(String)

```
public String mapScopeHandlerName(String path);
```

**Specified by:** Method mapScopeHandlerName in interface IMappingStrategy

Resolves scope handler name for path& Default application used as root when path is specified

#### Parameters

path	Path
<i>return</i>	Scope handler name according to this naming strategy

### 22.4. mapServiceName(String)

```
public String mapServiceName(String name);
```

**Specified by:** Method mapServiceName in interface IMappingStrategy

Resolves service filename name from name

#### Parameters

name	Service name
<i>return</i>	Service filename according to this naming strategy

### 22.5. setDefaultApp(String)

```
public void setDefaultApp(String defaultApp);
```

Setter for default application name ('default' by default).

#### Parameters

defaultApp	Default application
------------	---------------------

## 23. Class PersistableAttributeStore

Persistable attributes store. Server-side SharedObjects feature based on this class.

## 23.1. Synopsis

```

public class PersistableAttributeStore extends, org.?red5.?server.?AttributeStore
    implements, org.?red5.?server.?api.?persistence.?IPersistable {
// Protected Fields

    protected long lastModified ;

    protected String name ;

    protected String path ;

    protected boolean persistent ;

    protected org.red5.server.api.persistence.IPersistenceStore store ;

    protected String type ;

// Public Constructors

    public PersistableAttributeStore(String type,
                                    String name,
                                    String path,
                                    boolean persistent);

// Public Methods

    public void deserialize(org.red5.io.object.Input input)
        throws IOException;

    public Object getAttribute(String name,
                           Object defaultValue);

    public long getLastModified();

    public String getName();

    public String getPath();

    public org.red5.server.api.persistence.IPersistenceStore getStore();

    public String getType();

    public boolean isPersistent();

    public boolean removeAttribute(String name);

    public void removeAttributes();

    public void serialize(org.red5.io.object.Output output)
        throws IOException;

    public boolean setAttribute(String name,
                           Object value);

    public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);

    public void setAttributes(org.red5.server.api.IAttributeStore values);

```

```

    public void setName(String name);

    public void setPath(String path);

    public void setPersistent(boolean persistent);

    public void setStore(org.red5.server.api.persistence.IPersistenceStore store);

// Protected Methods

    protected void modified();

}

```

**Direct known subclasses:** org.?red5.?server.?BasicScope

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttribute ,  
getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute ,  
getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute ,  
getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute ,  
hasAttribute , removeAttribute , removeAttributes , setAttribute , setAttributes

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 23.2. PersistableAttributeStore(String, String, String, boolean)

```

public PersistableAttributeStore(String type,
                                String name,
                                String path,
                                boolean persistent);

```

Creates persistable attribute store

Parameters	
type	Attribute store type
name	Attribute store name
path	Attribute store path
persistent	Whether store is persistent or not

## 23.3. lastModified

```

protected long lastModified ;

```

Last modified Timestamp

## 23.4. name

```
protected String name ;
```

Attribute store name

## 23.5. path

```
protected String path ;
```

Attribute store path (on local hard drive)

## 23.6. persistent

```
protected boolean persistent ;
```

Persistence flag

## 23.7. store

```
protected org.red5.server.api.persistence.IPersistenceStore store ;
```

Store object that deals with save/load routines

## 23.8. type

```
protected String type ;
```

Attribute store type

## 23.9. deserialize(Input)

```
public void deserialize(org.red5.io.object.Input input)
throws IOException;
```

**Specified by:** Method deserialize in interface IPersistable

Deserializes data from input to attributes

### Parameters

input	Input object
-------	--------------

IOException

I/O exception

## 23.10. getAttribute(String, Object)

```
public Object getAttribute(String name,
                           Object defaultValue);
```

## 23.11. getLastModified()

```
public long getLastModified();
```

**Specified by:** Method getLastModified in interface IPersistable

Returns last modification time as timestamp

Parameters

<i>return</i>	Timestamp of last attribute modification
---------------	--

## 23.12. getName()

```
public String getName();
```

**Specified by:** Method getName in interface IPersistable

Return store name

Parameters

<i>return</i>	Store name
---------------	------------

## 23.13. getPath()

```
public String getPath();
```

**Specified by:** Method getPath in interface IPersistable

Return scope path

Parameters

<i>return</i>	Path
---------------	------

## 23.14. getStore()

```
public org.red5.server.api.persistence.IPersistenceStore getStore();
```

**Specified by:** Method getStore in interface IPersistable

Return persistent store

Parameters

<i>return</i>	Persistence store
---------------	-------------------

## 23.15. getType()

```
public String getType();
```

**Specified by:** Method getType in interface IPersistable

Return scope type

**Parameters**

<i>return</i>	Scope type
---------------	------------

**23.16. isPersistent()**

```
public boolean isPersistent();
```

**Specified by:** Method isPersistent in interface IPersistable

Check whether object is persistent or not

**Parameters**

<i>return</i>	true if object is persistent, false otherwise
---------------	---

**23.17. modified()**

```
protected void modified();
```

Set last modified flag to current system time

**23.18. removeAttribute(String)**

```
public boolean removeAttribute(String name);
```

Removes attribute

**Parameters**

<i>name</i>	Attribute name
<i>return</i>	true if attribute was removed, false otherwise

**23.19. removeAttributes()**

```
public void removeAttributes();
```

Removes all attributes and sets modified flag

**23.20. serialize(Output)**

```
public void serialize(org.red5.io.object.Output output)
throws IOException;
```

**Specified by:** Method serialize in interface IPersistable

Serializes byte buffer output, storing them to attributes

**Parameters**

<i>output</i>	Output object
---------------	---------------

```
IOException
java.io.IOException
```

## 23.21. setAttribute(String, Object)

```
public boolean setAttribute(String name,
                           Object value);
```

Set attribute by name and return success as boolean

### Parameters

name	Attribute name
value	Attribute value
<i>return</i>	true if attribute was set, false otherwise

## 23.22. setAttributes(IAttributeStore)

```
public void setAttributes(org.red5.server.api.IAttributeStore values);
```

Bulk set of attributes from another attributes store

### Parameters

values	Attributes store
--------	------------------

## 23.23. setAttributes(Map<String, Object>)

```
public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);
```

Set attributes from Map

### Parameters

values	Attributes as Map
--------	-------------------

## 23.24. setName(String)

```
public void setName(String name);
```

**Specified by:** Method `setName` in interface `IPersistable`

Setter for name

### Parameters

name	Name
------	------

## 23.25. setPath(String)

```
public void setPath(String path);
```

**Specified by:** Method setPath in interface IPersistable

Setter for scope path

Parameters

path	Path
------	------

## 23.26. setPersistent(boolean)

```
public void setPersistent(boolean persistent);
```

**Specified by:** Method setPersistent in interface IPersistable

Set for persistence

Parameters

persistent	Persistence flag value
------------	------------------------

## 23.27. setStore(IPersistenceStore)

```
public void setStore(org.red5.server.api.persistence.IPersistenceStore store);
```

**Specified by:** Method setStore in interface IPersistable

Load data from another persistent store

Parameters

store	Persistent store
-------	------------------

## 24. Class Scope

The scope object. A statefull object shared between a group of clients connected to the same context path. Scopes are arranged in a hierarchical way, so its possible for a scope to have a parent. If a client is connect to a scope then they are also connected to its parent scope. The scope object is used to access resources, shared object, streams, etc. The following are all names for scopes: application, room, place, lobby.

### 24.1. Synopsis

```
public class Scope extends, org.?red5.?server.?BasicScope
    implements, org.?red5.?server.?api.?IScope, org.?red5.?server.?api.?statistics.?IScopeStatistics,
    // Protected Fields

    protected final org.red5.server.api.statistics.support.StatisticsCounter clientStats ;

    protected final org.red5.server.api.statistics.support.StatisticsCounter connectionStats ;

    protected static org.slf4j.Logger log ;

    protected javax.management.ObjectName oName ;
```

```
protected final org.red5.server.api.statistics.support.StatisticsCounter subscopeStats ;  
  
// Public Constructors  
  
public Scope();  
  
public Scope(String name);  
  
// Public Methods  
  
public boolean addChildScope(org.red5.server.api.IBasicScope scope);  
  
public boolean connect(org.red5.server.api.IConnection conn);  
  
public boolean connect(org.red5.server.api.IConnection conn,  
                      Object[] params);  
  
public boolean createChildScope(String name);  
  
public void destroy();  
  
public void disconnect(org.red5.server.api.IConnection conn);  
  
public void dispatchEvent(org.red5.server.api.event.IEvent event);  
  
public int getActiveClients();  
  
public int getActiveConnections();  
  
public int getActiveSubscopes();  
  
public org.red5.server.api.IBasicScope getBasicScope(String type,  
                                                    String name);  
  
public java.util.Iterator<java.lang.String> getBasicScopeNames(String type);  
  
public ClassLoader getClassLoader();  
  
public java.util.Set<org.red5.server.api.IClient> getClients();  
  
public java.util.Iterator<org.red5.server.api.IConnection> getConnections();  
  
public org.red5.server.api.IContext getContext();  
  
public String getContextPath();  
  
public long getCreationTime();  
  
public int getDepth();  
  
public boolean getEnabled();  
  
public org.red5.server.api.IScopeHandler getHandler();  
  
public int getMaxClients();
```

## Package org.?red5.?server

```
public int getMaxConnections();

public int getMaxSubscopes();

public org.red5.server.api.IScope getParent();

public String getPath();

public org.springframework.core.io.Resource getResource(String path);

public org.springframework.core.io.Resource[] getResources(String path)
throws IOException;

public boolean getRunning();

public org.red5.server.api.IScope getScope(String name);

public java.util.Iterator<java.lang.String> getScopeNames();

public org.red5.server.api.IServer getServer();

public Object getServiceHandler(String name);

public java.util.Set<java.lang.String> getServiceHandlerNames();

public org.red5.server.api.statistics.IScopeStatistics getStatistics();

public int getTotalClients();

public int getTotalConnections();

public int getTotalSubscopes();

public boolean handleEvent(org.red5.server.api.event.IEvent event);

public boolean hasChildScope(String name);

public boolean hasChildScope(String type,
                           String name);

public boolean hasContext();

public boolean hasHandler();

public boolean hasParent();

public void init();

public boolean isEnabled();

public boolean isRunning();

public java.util.Iterator<org.red5.server.api.IBasicScope> iterator();

public java.util.Set<org.red5.server.api.IConnection> lookupConnections(org.red5.server.api.IClien
```

```

public void registerServiceHandler(String name,
                                  Object handler);

public void removeChildScope(org.red5.server.api.IBasicScope scope);

public void setAutoStart(boolean autoStart);

public void setChildLoadPath(String pattern);

public void setContext(org.red5.server.api.IContext context);

public void setDepth(int depth);

public void setEnabled(boolean enabled);

public void setHandler(org.red5.server.api.IScopeHandler handler);

public void setName(String name);

public void setParent(org.red5.server.api.IScope parent);

public void setPersistenceClass(String persistenceClass)
    throws Exception;

public synchronized boolean start();

public synchronized void stop();

public String toString();

public void uninit();

public void unregisterServiceHandler(String name);

// Protected Methods

protected java.util.Map<java.lang.String, java.lang.Object> getServiceHandlers();

protected java.util.Map<java.lang.String, java.lang.Object> getServiceHandlers(boolean allowCreate);

}

```

**Direct known subclasses:** org.?red5.?server.?GlobalScope , org.?red5.?server.?WebScope

**Methods inherited from org.red5.server.BasicScope:** addEventListener , dispatchEvent , getDepth , getEventListeners , getParent , getPath , handleEvent , hasParent , iterator , notifyEvent , removeEventListener

**Methods inherited from org.red5.server.PersistableAttributeStore:** deserialize , getAttribute , getLastModified , getName , getStore , getType , isPersistent , modified , removeAttribute , removeAttributes , serialize , setAttribute , setAttributes , setName , setPath , setPersistent , setStore

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute ,

```
getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute ,
getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute ,
hasAttribute
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.BasicScope:** keepOnDisconnect , listeners ,  
parent , persistenceClass

**Fields inherited from org.red5.server.PersistableAttributeStore:** lastModified , name  
, path , persistent , store , type

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 24.2. Scope()

```
public Scope();
```

Creates unnamed scope

## 24.3. Scope(String)

```
public Scope(String name);
```

Creates scope with given name

### Parameters

name	Scope name
------	------------

## 24.4. clientStats

```
protected final org.red5.server.api.statistics.support.StatisticsCounter clientStats ;
```

Statistics about clients connected to the scope.

## 24.5. connectionStats

```
protected final org.red5.server.api.statistics.support.StatisticsCounter connectionStats ;
```

Statistics about connections to the scope.

## 24.6. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 24.7. oName

```
protected javax.management.ObjectName oName ;
```

Mbean object name.

## 24.8. subscopeStats

```
protected final org.red5.server.api.statistics.support.StatisticsCounter subscopeStats ;
```

Statistics about subscopes.

## 24.9. addChildScope(IBasicScope)

```
public boolean addChildScope(org.red5.server.api.IBasicScope scope);
```

**Specified by:** Method addChildScope in interface IScope

Add child scope to this scope

Parameters	
scope	Child scope
<i>return</i>	true on success (if scope has handler and it accepts child scope addition), false otherwise

## 24.10. connect(IConnection)

```
public boolean connect(org.red5.server.api.IConnection conn);
```

**Specified by:** Method connect in interface IScope

Connect to scope

Parameters	
conn	Connection object
<i>return</i>	true on success, false otherwise

## 24.11. connect(IConnection, Object[])

```
public boolean connect(org.red5.server.api.IConnection conn,
Object[] params);
```

**Specified by:** Method connect in interface IScope

Connect to scope with parameters. To successfully connect to scope it must have handler that will accept this connection with given set of parameters. Client associated with connection is added to scope clients set, connection is registered as scope event listener.

Parameters	
conn	Connection object
params	Parameters passed with connection

<i>return</i>	true on success, false otherwise
---------------	----------------------------------

## 24.12. createChildScope(String)

```
public boolean createChildScope(String name);
```

**Specified by:** Method createChildScope in interface IScope

Create child scope with given name

### Parameters

name	Child scope name
------	------------------

<i>return</i>	true on success, false otherwise
---------------	----------------------------------

## 24.13. destroy()

```
public void destroy();
```

**Specified by:** Method destroy in interface ScopeMBean

Destroys scope

## 24.14. disconnect(IConnection)

```
public void disconnect(org.red5.server.api.IConnection conn);
```

**Specified by:** Method disconnect in interface IScope

Disconnect connection from scope

### Parameters

conn	Connection object
------	-------------------

## 24.15. dispatchEvent(IEvent)

```
public void dispatchEvent(org.red5.server.api.event.IEvent event);
```

## 24.16. getActiveClients()

```
public int getActiveClients();
```

**Specified by:** Method getActiveClients in interface IScopeStatistics

Return current number of clients connected to the scope.

## 24.17. getActiveConnections()

```
public int getActiveConnections();
```

**Specified by:** Method getActiveConnections in interface IScopeStatistics

Return current number of connections to the scope.

## 24.18. **getActiveSubscopes()**

```
public int getActiveSubscopes();
```

**Specified by:** Method `getActiveSubscopes` in interface `IScopeStatistics`

Return number of currently existing subscopes.

## 24.19. **getBasicScope(String, String)**

```
public org.red5.server.api.IBasicScope getBasicScope(String type,
                                                    String name);
```

**Specified by:** Method `getBasicScope` in interface `IScope`

Return base scope of given type with given name

### Parameters

type	Scope type
name	Scope name
<i>return</i>	Basic scope object

## 24.20. **getBasicScopeNames(String)**

```
public java.util.Iterator<java.lang.String> getBasicScopeNames(String type);
```

**Specified by:** Method `getBasicScopeNames` in interface `IScope`

Return basic scope names iterator

### Parameters

type	Scope type
<i>return</i>	Iterator

## 24.21. **getClassLoader()**

```
public ClassLoader getClassLoader();
```

Return current thread context classloader

### Parameters

<i>return</i>	Current thread context classloader
---------------	------------------------------------

## 24.22. **getClients()**

```
public java.util.Set<org.red5.server.api.IClient> getClients();
```

**Specified by:** Method `getClients` in interface `IScope`

Return set of clients

## Parameters

<i>return</i>	Set of clients bound to scope
---------------	-------------------------------

**24.23. `getConnections()`**

```
public java.util.Iterator<org.red5.server.api.IConnection> getConnections();
```

**Specified by:** Method `getConnections` in interface `IScope`

Return connection iterator

## Parameters

<i>return</i>	Connections iterator
---------------	----------------------

**24.24. `getContext()`**

```
public org.red5.server.api.IContext getContext();
```

**Specified by:** Method `getContext` in interface `IScope`

Return scope context. If scope doesn't have context, parent's context is returns, and so forth.

## Parameters

<i>return</i>	Scope context or parent context
---------------	---------------------------------

**24.25. `getContextPath()`**

```
public String getContextPath();
```

**Specified by:** Method `getContextPath` in interface `IScope`

Return scope context path

## Parameters

<i>return</i>	Scope context path
---------------	--------------------

**24.26. `getCreationTime()`**

```
public long getCreationTime();
```

**24.27. `getDepth()`**

```
public int getDepth();
```

**Specified by:** Method `getDepth` in interface `IScopeStatistics`

return scope depth

#### Parameters

<i>return</i>	Scope depth
---------------	-------------

## 24.28. getEnabled()

```
public boolean getEnabled();
```

**Specified by:** Method getEnabled in interface ScopeMBean

Here for JMX only, uses isEnabled()

## 24.29. getHandler()

```
public org.red5.server.api.IScopeHandler getHandler();
```

**Specified by:** Method getHandler in interface IScope

Return scope handler or parent's scope handler if this scope doesn't have one

#### Parameters

<i>return</i>	Scope handler (or parent's one)
---------------	---------------------------------

## 24.30. getMaxClients()

```
public int getMaxClients();
```

**Specified by:** Method getMaxClients in interface IScopeStatistics

Return maximum number of clients concurrently connected to the scope.

## 24.31. getMaxConnections()

```
public int getMaxConnections();
```

**Specified by:** Method getMaxConnections in interface IScopeStatistics

Return maximum number of concurrent connections to the scope.

## 24.32. getMaxSubscopes()

```
public int getMaxSubscopes();
```

**Specified by:** Method getMaxSubscopes in interface IScopeStatistics

Return maximum number of concurrently existing subscopes.

## 24.33. getParent()

```
public org.red5.server.api.IScope getParent();
```

**Specified by:** Method getParent in interface ScopeMBean

Return parent scope

Parameters

<i>return</i>	Parent scope
---------------	--------------

## 24.34. getPath()

```
public String getPath();
```

**Specified by:** Method getPath in interface IScopeStatistics

Return scope path calculated from parent path and parent scope name

Parameters

<i>return</i>	Scope path
---------------	------------

## 24.35. getResource(String)

```
public org.springframework.core.io.Resource getResource(String path);
```

**Specified by:** Method getResource in interface ScopeMBean

Return resource located at given path

Parameters

path	Resource path
<i>return</i>	Resource

## 24.36. getResources(String)

```
public org.springframework.core.io.Resource[] getResources(String path)
throws IOException;
```

**Specified by:** Method getResources in interface ScopeMBean

Return array of resources from path string, usually used with pattern path

Parameters

path	Resources path
<i>return</i>	Resources

IOException

I/O exception

## 24.37. getRunning()

```
public boolean getRunning();
```

**Specified by:** Method getRunning in interface ScopeMBean

Here for JMX only, uses isEnabled()

## 24.38. getScope(String)

```
public org.red5.server.api.IScope getScope(String name);
```

**Specified by:** Method getScope in interface IScope

Return child scope by name

### Parameters

name	Scope name
<i>return</i>	Child scope with given name

## 24.39. getScopeNames()

```
public java.util.Iterator<java.lang.String> getScopeNames();
```

**Specified by:** Method getScopeNames in interface IScope

Return child scope names iterator

### Parameters

<i>return</i>	Child scope names iterator
---------------	----------------------------

## 24.40. getServer()

```
public org.red5.server.api.IServer getServer();
```

Return the server instance connected to this scope.

### Parameters

<i>return</i>	the server instance
---------------	---------------------

## 24.41. getServiceHandler(String)

```
public Object getServiceHandler(String name);
```

**Specified by:** Method getServiceHandler in interface ScopeMBean

Return service handler by name

### Parameters

name	Handler name
------	--------------

<i>return</i>	Service handler with given name
---------------	---------------------------------

## 24.42. getServiceHandlerNames()

```
public java.util.Set<java.lang.String> getServiceHandlerNames();
```

**Specified by:** Method `getServiceHandlerNames` in interface `ScopeMBean`

Return set of service handler names. Removing entries from the set unregisters the corresponding service handler.

### Parameters

<i>return</i>	Set of service handler names
---------------	------------------------------

## 24.43. getServiceHandlers()

```
protected java.util.Map<java.lang.String, java.lang.Object> getServiceHandlers();
```

Return map of service handlers. The map is created if it doesn't exist yet.

### Parameters

<i>return</i>	Map of service handlers
---------------	-------------------------

## 24.44. getServiceHandlers(boolean)

```
protected java.util.Map<java.lang.String, java.lang.Object> getServiceHandlers(boolean allowCreate);
```

Return map of service handlers and optionally created it if it doesn't exist.

### Parameters

allowCreate	Should the map be created if it doesn't exist?
-------------	--

<i>return</i>	Map of service handlers
---------------	-------------------------

## 24.45. getStatistics()

```
public org.red5.server.api.statistics.IScopeStatistics getStatistics();
```

**Specified by:** Method `getStatistics` in interface `IScope`

Return statistics informations about the scope.

## 24.46. getTotalClients()

```
public int getTotalClients();
```

**Specified by:** Method `getTotalClients` in interface `IScopeStatistics`

Return total number of clients connected to the scope.

## 24.47. getTotalConnections()

```
public int getTotalConnections();
```

**Specified by:** Method getTotalConnections in interface IScopeStatistics

Return total number of connections to the scope.

## 24.48. getTotalSubscopes()

```
public int getTotalSubscopes();
```

**Specified by:** Method getTotalSubscopes in interface IScopeStatistics

Return total number of subscopes created.

## 24.49. handleEvent(IEvent)

```
public boolean handleEvent(org.red5.server.api.event.IEvent event);
```

Handles event. To be implemented in subclasses.

Parameters	
event	Event to handle
<i>return</i>	true on success, false otherwise

## 24.50. hasChildScope(String)

```
public boolean hasChildScope(String name);
```

**Specified by:** Method hasChildScope in interface IScope

Check whether scope has child scope with given name

Parameters	
name	Child scope name
<i>return</i>	true if scope has child node with given name, false otherwise

## 24.51. hasChildScope(String, String)

```
public boolean hasChildScope(String type,
                             String name);
```

**Specified by:** Method hasChildScope in interface IScope

Check whether scope has child scope with given name and type

Parameters	
type	Child scope type

<i>name</i>	Child scope name
<i>return</i>	<code>true</code> if scope has child node with given name and type, <code>false</code> otherwise

## 24.52. hasContext()

```
public boolean hasContext();
```

**Specified by:** Method hasContext in interface ScopeMBean

Check if scope has a context

### Parameters

<i>return</i>	<code>true</code> if scope has context, <code>false</code> otherwise
---------------	--

## 24.53. hasHandler()

```
public boolean hasHandler();
```

**Specified by:** Method hasHandler in interface IScope

Check if scope or it's parent has handler

### Parameters

<i>return</i>	<code>true</code> if scope or it's parent scope has a handler, <code>false</code> otherwise
---------------	---

## 24.54. hasParent()

```
public boolean hasParent();
```

**Specified by:** Method hasParent in interface ScopeMBean

Check if scope has parent scope

### Parameters

<i>return</i>	<code>true</code> if scope has parent scope, <code>false</code> otherwise
---------------	---

## 24.55. init()

```
public void init();
```

**Specified by:** Method init in interface ScopeMBean

Initialization actions, start if autostart is set to `true`

## 24.56. isEnabled()

```
public boolean isEnabled();
```

Check if scope is enabled

#### Parameters

<i>return</i>	true if scope is enabled, false otherwise
---------------	---

## 24.57. isRunning()

```
public boolean isRunning();
```

Check if scope is in running state

#### Parameters

<i>return</i>	true if scope is in running state, false otherwise
---------------	--

## 24.58. iterator()

```
public java.util.Iterator<org.red5.server.api.IBasicScope> iterator();
```

Child scopes iterator

#### Parameters

<i>return</i>	Child scopes iterator
---------------	-----------------------

## 24.59. lookupConnections(IClient)

```
public java.util.Set<org.red5.server.api.IConnection> lookupConnections(org.red5.server.api.IClient
```

**Specified by:** Method lookupConnections in interface IScope

Looks up connections for client

#### Parameters

client	Client
<i>return</i>	Connection

## 24.60. registerServiceHandler(String, Object)

```
public void registerServiceHandler(String name,
                                  Object handler);
```

**Specified by:** Method registerServiceHandler in interface ScopeMBean

Register service handler by name

#### Parameters

name	Service handler name
------	----------------------

handler	Service handler
---------	-----------------

## 24.61. removeChildScope(IBasicScope)

```
public void removeChildScope(org.red5.server.api.IBasicScope scope);
```

**Specified by:** Method removeChildScope in interface IScope

Removes child scope

### Parameters

scope	Child scope to remove
-------	-----------------------

## 24.62. setAutoStart(boolean)

```
public void setAutoStart(boolean autoStart);
```

**Specified by:** Method setAutoStart in interface ScopeMBean

Setter for autostart flag

### Parameters

autoStart	Autostart flag value
-----------	----------------------

## 24.63. setChildLoadPath(String)

```
public void setChildLoadPath(String pattern);
```

**Specified by:** Method setChildLoadPath in interface ScopeMBean

Setter for child load path. Should be implemented in subclasses?

### Parameters

pattern	Load path pattern
---------	-------------------

## 24.64. setContext(IContext)

```
public void setContext(org.red5.server.api.IContext context);
```

Setter for context

### Parameters

context	Context object
---------	----------------

## 24.65. setDepth(int)

```
public void setDepth(int depth);
```

**Specified by:** Method setDepth in interface ScopeMBean

Set scope depth

## Parameters

depth	Scope depth
-------	-------------

**24.66. setEnabled(boolean)**

```
public void setEnabled(boolean enabled);
```

**Specified by:** Method setEnabled in interface ScopeMBean

Enable or disable scope by setting enable flag

## Parameters

enabled	Enable flag value
---------	-------------------

**24.67. setHandler(IScopeHandler)**

```
public void setHandler(org.red5.server.api.IScopeHandler handler);
```

Setter for scope event handler

## Parameters

handler	Event handler
---------	---------------

**24.68. setName(String)**

```
public void setName(String name);
```

**Specified by:** Method setName in interface ScopeMBean

Setter for scope name

## Parameters

name	Scope name
------	------------

**24.69. setParent(IScope)**

```
public void setParent(org.red5.server.api.IScope parent);
```

Setter for parent scope

## Parameters

parent	Parent scope
--------	--------------

## 24.70. setPersistenceClass(String)

```
public void setPersistenceClass(String persistenceClass)
    throws Exception;
```

**Specified by:** Method setPersistenceClass in interface ScopeMBean

Set scope persistence class

### Parameters

persistenceClass	Scope's persistence class
------------------	---------------------------

Exception

Exception

## 24.71. start()

```
public synchronized boolean start();
```

**Specified by:** Method start in interface ScopeMBean

Starts scope

### Parameters

return	true if scope has handler and it's start method returned true, false otherwise
--------	---

## 24.72. stop()

```
public synchronized void stop();
```

**Specified by:** Method stop in interface ScopeMBean

Stops scope

## 24.73. toString()

```
public String toString();
```

## 24.74. uninit()

```
public void uninit();
```

Uninitialize scope and unregister from parent.

## 24.75. unregisterServiceHandler(String)

```
public void unregisterServiceHandler(String name);
```

**Specified by:** Method unregisterServiceHandler in interface ScopeMBean

Registers service handler by name

**Parameters**

name	Service handler name
------	----------------------

## 25. Interface ScopeMBean

An MBean interface for the scope object.

### 25.1. Synopsis

```

public interface ScopeMBean {
    // Public Methods

    public boolean createChildScope(String name);

    public void destroy();

    public int getActiveClients();

    public int getActiveConnections();

    public int getActiveSubscopes();

    public org.red5.server.api.IBasicScope getBasicScope(String type,
                                                       String name);

    public java.util.Iterator<java.lang.String> getBasicScopeNames(String type);

    public java.util.Set<org.red5.server.api.IClient> getClients();

    public java.util.Iterator<org.red5.server.api.IConnection> getConnections();

    public org.red5.server.api.IContext getContext();

    public String getContextPath();

    public int getDepth();

    public boolean getEnabled();

    public org.red5.server.api.IScopeHandler getHandler();

    public int getMaxClients();

    public int getMaxConnections();

    public int getMaxSubscopes();

    public org.red5.server.api.IScope getParent();

    public String getPath();

    public org.springframework.core.io.Resource getResource(String path);

    public org.springframework.core.io.Resource[] getResources(String path)
}

```

## Package org.?red5.?server

```
throws IOException;

public boolean getRunning();

public org.red5.server.api.IScope getScope(String name);

public java.util.Iterator<java.lang.String> getScopeNames();

public Object getServiceHandler(String name);

public java.util.Set<java.lang.String> getServiceHandlerNames();

public int getTotalClients();

public int getTotalConnections();

public int getTotalSubscopes();

public boolean hasChildScope(String name);

public boolean hasChildScope(String type,
                           String name);

public boolean hasContext();

public boolean hasHandler();

public boolean hasParent();

public void init();

public void registerServiceHandler(String name,
                                   Object handler);

public void setAutoStart(boolean autoStart);

public void setChildLoadPath(String pattern);

public void setDepth(int depth);

public void setEnabled(boolean enabled);

public void setName(String name);

public void setPersistenceClass(String persistenceClass)
    throws Exception;

public boolean start();

public void stop();

public void unregisterServiceHandler(String name);

}
```

## 25.2. createChildScope(String)

```
public boolean createChildScope(String name);
```

Create child scope with given name

### Parameters

name	Child scope name
<i>return</i>	true on success, false otherwise

## 25.3. destroy()

```
public void destroy();
```

Destroys scope

## 25.4. getActiveClients()

```
public int getActiveClients();
```

Return current number of clients connected to the scope.

### Parameters

<i>return</i>	number of clients
---------------	-------------------

## 25.5. getActiveConnections()

```
public int getActiveConnections();
```

Return current number of connections to the scope.

### Parameters

<i>return</i>	number of connections
---------------	-----------------------

## 25.6. getActiveSubscopes()

```
public int getActiveSubscopes();
```

Return number of currently existing subscopes.

### Parameters

<i>return</i>	number of subscopes
---------------	---------------------

## 25.7. getBasicScope(String, String)

```
public org.red5.server.api.IBasicScope getBasicScope(String type,
                                                    String name);
```

Return base scope of given type with given name

**Parameters**

<code>type</code>	Scope type
<code>name</code>	Scope name
<code>return</code>	Basic scope object

**25.8. getBasicScopeNames(String)**

```
public java.util.Iterator<java.lang.String> getBasicScopeNames(String type);
```

Return basic scope names iterator

**Parameters**

<code>type</code>	Scope type
<code>return</code>	Iterator

**25.9. getClients()**

```
public java.util.Set<org.red5.server.api.IClient> getClients();
```

Return set of clients

**Parameters**

<code>return</code>	Set of clients bound to scope
---------------------	-------------------------------

**25.10. getConnections()**

```
public java.util.Iterator<org.red5.server.api.IConnection> getConnections();
```

Return connection iterator

**Parameters**

<code>return</code>	Connections iterator
---------------------	----------------------

**25.11. getContext()**

```
public org.red5.server.api.IContext getContext();
```

Return scope context. If scope doesn't have context, parent's context is returns, and so forth.

**Parameters**

<code>return</code>	Scope context or parent context
---------------------	---------------------------------

**25.12. getContextPath()**

```
public String getContextPath();
```

Return scope context path

#### Parameters

<i>return</i>	Scope context path
---------------	--------------------

### 25.13. getDepth()

```
public int getDepth();
```

return scope depth

#### Parameters

<i>return</i>	Scope depth
---------------	-------------

### 25.14. getEnabled()

```
public boolean getEnabled();
```

Check if scope is enabled

#### Parameters

<i>return</i>	true if scope is enabled, false otherwise
---------------	---

### 25.15. getHandler()

```
public org.red5.server.api.IScopeHandler getHandler();
```

Return scope handler or parent's scope handler if this scope doesn't have one

#### Parameters

<i>return</i>	Scope handler (or parent's one)
---------------	---------------------------------

### 25.16. getMaxClients()

```
public int getMaxClients();
```

Return maximum number of clients concurrently connected to the scope.

#### Parameters

<i>return</i>	number of clients
---------------	-------------------

### 25.17. getMaxConnections()

```
public int getMaxConnections();
```

Return maximum number of concurrent connections to the scope.

#### Parameters

<i>return</i>	number of connections
---------------	-----------------------

## 25.18. getMaxSubscopes()

```
public int getMaxSubscopes();
```

Return maximum number of concurrently existing subscopes.

### Parameters

<i>return</i>	number of subscopes
---------------	---------------------

## 25.19. getParent()

```
public org.red5.server.api.IScope getParent();
```

Return parent scope

### Parameters

<i>return</i>	Parent scope
---------------	--------------

## 25.20. getPath()

```
public String getPath();
```

Return scope path calculated from parent path and parent scope name

### Parameters

<i>return</i>	Scope path
---------------	------------

## 25.21. getResource(String)

```
public org.springframework.core.io.Resource getResource(String path);
```

Return resource located at given path

### Parameters

path	Resource path
<i>return</i>	Resource

## 25.22. getResources(String)

```
public org.springframework.core.io.Resource[] getResources(String path)
throws IOException;
```

Return array of resources from path string, usually used with pattern path

### Parameters

path	Resources path
return	Resources

IOException  
I/O exception

## 25.23. getRunning()

```
public boolean getRunning();
```

Check if scope is in running state

### Parameters

return	true if scope is in running state, false otherwise
--------	--

## 25.24. getScope(String)

```
public org.red5.server.api.IScope getScope(String name);
```

Return child scope by name

### Parameters

name	Scope name
return	Child scope with given name

## 25.25. getScopeNames()

```
public java.util.Iterator<java.lang.String> getScopeNames();
```

Return child scope names iterator

### Parameters

return	Child scope names iterator
--------	----------------------------

## 25.26. getServiceHandler(String)

```
public Object getServiceHandler(String name);
```

Return service handler by name

### Parameters

name	Handler name
return	Service handler with given name

## 25.27. getServiceHandlerNames()

```
public java.util.Set<java.lang.String> getServiceHandlerNames();
```

Return set of service handler names

#### Parameters

<i>return</i>	Set of service handler names
---------------	------------------------------

## 25.28. getTotalClients()

```
public int getTotalClients();
```

Return total number of clients connected to the scope.

#### Parameters

<i>return</i>	number of clients
---------------	-------------------

## 25.29. getTotalConnections()

```
public int getTotalConnections();
```

Return total number of connections to the scope.

#### Parameters

<i>return</i>	number of connections
---------------	-----------------------

## 25.30. getTotalSubscopes()

```
public int getTotalSubscopes();
```

Return total number of subscopes created.

#### Parameters

<i>return</i>	number of subscopes created
---------------	-----------------------------

## 25.31. hasChildScope(String)

```
public boolean hasChildScope(String name);
```

Check whether scope has child scope with given name

#### Parameters

<i>name</i>	Child scope name
-------------	------------------

<i>return</i>	true if scope has child node with given name, false otherwise
---------------	---

## 25.32. hasChildScope(String, String)

```
public boolean hasChildScope(String type,
                             String name);
```

Check whether scope has child scope with given name and type

**Parameters**

<code>type</code>	Child scope type
<code>name</code>	Child scope name
<code>return</code>	<code>true</code> if scope has child node with given name and type, <code>false</code> otherwise

**25.33. hasContext()**

```
public boolean hasContext();
```

Check if scope has a context

**Parameters**

<code>return</code>	<code>true</code> if scope has context, <code>false</code> otherwise
---------------------	--

**25.34. hasHandler()**

```
public boolean hasHandler();
```

Check if scope or it's parent has handler

**Parameters**

<code>return</code>	<code>true</code> if scope or it's parent scope has a handler, <code>false</code> otherwise
---------------------	---

**25.35. hasParent()**

```
public boolean hasParent();
```

Check if scope has parent scope

**Parameters**

<code>return</code>	<code>true</code> if scope has parent scope, <code>false</code> otherwise
---------------------	---

**25.36. init()**

```
public void init();
```

Initialization actions, start if autostart is set to `true`

**25.37. registerServiceHandler(String, Object)**

```
public void registerServiceHandler(String name,
                                  Object handler);
```

Register service handler by name

**Parameters**

name	Service handler name
handler	Service handler

## 25.38. setAutoStart(boolean)

```
public void setAutoStart(boolean autoStart);
```

Setter for autostart flag

Parameters	
autoStart	Autostart flag value

## 25.39. setChildLoadPath(String)

```
public void setChildLoadPath(String pattern);
```

Setter for child load path. Should be implemented in subclasses?

Parameters	
pattern	Load path pattern

## 25.40. setDepth(int)

```
public void setDepth(int depth);
```

Set scope depth

Parameters	
depth	Scope depth

## 25.41. setEnabled(boolean)

```
public void setEnabled(boolean enabled);
```

Enable or disable scope by setting enable flag

Parameters	
enabled	Enable flag value

## 25.42. setName(String)

```
public void setName(String name);
```

Setter for scope name

Parameters	

name	Scope name
------	------------

## 25.43. setPersistenceClass(String)

```
public void setPersistenceClass(String persistenceClass)
throws Exception;
```

Set scope persistence class

### Parameters

persistenceClass	Scope's persistence class
------------------	---------------------------

Exception

Exception

## 25.44. start()

```
public boolean start();
```

Starts scope

### Parameters

return	true if scope has handler and it's start method returned true, false otherwise
--------	---

## 25.45. stop()

```
public void stop();
```

Stops scope

## 25.46. unregisterServiceHandler(String)

```
public void unregisterServiceHandler(String name);
```

Unregisters service handler by name

### Parameters

name	Service handler name
------	----------------------

# 26. Class ScopeResolver

Resolves scopes from path

## 26.1. Synopsis

```
public class ScopeResolver implements, org.?red5.?server.?api.?IScopeResolver {
// Public Static Fields

    public static final String DEFAULT_HOST = "";
```

```
// Protected Fields

protected org.red5.server.api.IGlobalScope globalScope ;

// Public Constructors

public ScopeResolver();

// Public Methods

public org.red5.server.api.IGlobalScope getGlobalScope();

public org.red5.server.api.IScope resolveScope(String path);

public org.red5.server.api.IScope resolveScope(org.red5.server.api.IScope root,
String path);

public void setGlobalScope(org.red5.server.api.IGlobalScope root);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 26.2. DEFAULT\_HOST

```
public static final String DEFAULT_HOST = "";
```

Default host constant

## 26.3. globalScope

```
protected org.red5.server.api.IGlobalScope globalScope ;
```

Global scope

## 26.4. getGlobalScope()

```
public org.red5.server.api.IGlobalScope getGlobalScope();
```

**Specified by:** Method getGlobalScope in interface IScopeResolver

Getter for global scope

### Parameters

return	Global scope
--------	--------------

## 26.5. resolveScope(IScope, String)

```
public org.red5.server.api.IScope resolveScope(org.red5.server.api.IScope root,
String path);
```

**Specified by:** Method resolveScope in interface IScopeResolver

Return scope associated with given path from given root scope.

#### Parameters

root	Scope to start from
path	Scope path
<i>return</i>	Scope object

## 26.6. resolveScope(String)

```
public org.red5.server.api.IScope resolveScope(String path);
```

**Specified by:** Method resolveScope in interface IScopeResolver

Return scope associated with given path

#### Parameters

path	Scope path
<i>return</i>	Scope object

## 26.7. setGlobalScope(IGlobalScope)

```
public void setGlobalScope(org.red5.server.api.IGlobalScope root);
```

Setter for global scope

#### Parameters

root	Global scope
------	--------------

# 27. Class Server

Red5 server core class implementation.

## 27.1. Synopsis

```
public class Server implements, org.?red5.?server.?api.?IServer, org.?springframework.?context.?ApplicationListener {
    // Public Fields

    public java.util.Set<org.red5.server.api.listeners.IConnectionListener> connectionListeners;

    public java.util.Set<org.red5.server.api.listeners.IScopeListener> scopeListeners;

    // Protected Fields

    protected static final String EMPTY = "";
    protected static final String SLASH = "/";
    protected org.springframework.context.ApplicationContext applicationContext;
}
```

## Package org.?red5.?server

```
protected java.util.concurrent.ConcurrentMap<java.lang.String, org.red5.server.api.IGlobalScope> g

protected static org.slf4j.Logger log ;

protected java.util.concurrent.ConcurrentMap<java.lang.String, java.lang.String> mapping ;

// Public Constructors

public Server();

// Public Methods

public void addListener(org.red5.server.api.listeners.IConnectionListener listener);

public void addListener(org.red5.server.api.listeners.IScopeListener listener);

public boolean addMapping(String hostName,
                          String contextPath,
                          String globalName);

public org.red5.server.api.IGlobalScope getGlobal(String name);

public java.util.Iterator<java.lang.String> getGlobalNames();

public java.util.Iterator<org.red5.server.api.IGlobalScope> getGlobalScopes();

public java.util.Map<java.lang.String, java.lang.String> getMappingTable();

public org.red5.server.api.IGlobalScope lookupGlobal(String hostName,
                                                    String contextPath);

public void registerGlobal(org.red5.server.api.IGlobalScope scope);

public void removeListener(org.red5.server.api.listeners.IConnectionListener listener);

public void removeListener(org.red5.server.api.listeners.IScopeListener listener);

public boolean removeMapping(String hostName,
                            String contextPath);

public void setApplicationContext(org.springframework.context.ApplicationContext applicationContext);

public String toString();

// Protected Methods

protected String getKey(String hostName,
                      String contextPath);

protected void notifyConnected(org.red5.server.api.IConnection conn);

protected void notifyDisconnected(org.red5.server.api.IConnection conn);

protected void notifyScopeCreated(org.red5.server.api.IScope scope);

protected void notifyScopeRemoved(org.red5.server.api.IScope scope);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 27.2. applicationContext

```
protected org.springframework.context.ApplicationContext applicationContext ;
```

Spring application context

## 27.3. EMPTY

```
protected static final String EMPTY = "";
```

Constant for empty string

## 27.4. globals

```
protected java.util.concurrent.ConcurrentMap<java.lang.String, org.red5.server.api.IGlobalScope> glo
```

List of global scopes

## 27.5. mapping

```
protected java.util.concurrent.ConcurrentMap<java.lang.String, java.lang.String> mapping ;
```

Mappings

## 27.6. SLASH

```
protected static final String SLASH = "/";
```

Constant for slash

## 27.7. addListener(IConnectionListener)

```
public void addListener(org.red5.server.api.listeners.IConnectionListener listener);
```

**Specified by:** Method addListener in interface IServer

Add listener to get notified about connection events.

## 27.8. addListener(IScopeListener)

```
public void addListener(org.red5.server.api.listeners.IScopeListener listener);
```

**Specified by:** Method addListener in interface IServer

Add listener to get notified about scope events.

## 27.9. addMapping(String, String, String)

```
public boolean addMapping(String hostName,
```

```
String contextPath,
String globalName);
```

**Specified by:** Method addMapping in interface IServer

Map key (host + / + context path) and global scope name

#### Parameters

hostName	Host name
contextPath	Context path
globalName	Global scope name
<i>return</i>	true if mapping was added, false if already exist

## 27.10. getGlobal(String)

```
public org.red5.server.api.IGlobalScope getGlobal(String name);
```

**Specified by:** Method getGlobal in interface IServer

Return global scope by name

#### Parameters

name	Global scope name
<i>return</i>	Global scope

## 27.11. getGlobalNames()

```
public java.util.Iterator<java.lang.String> getGlobalNames();
```

**Specified by:** Method getGlobalNames in interface IServer

Return global scope names set iterator

#### Parameters

<i>return</i>	Iterator
---------------	----------

## 27.12. getGlobalScopes()

```
public java.util.Iterator<org.red5.server.api.IGlobalScope> getGlobalScopes();
```

**Specified by:** Method getGlobalScopes in interface IServer

Return global scopes set iterator

#### Parameters

<i>return</i>	Iterator
---------------	----------

## 27.13. getKey(String, String)

```
protected String getKey(String hostName,
                      String contextPath);
```

Return scope key. Scope key consists of host name concatenated with context path by slash symbol

### Parameters

hostName	Host name
contextPath	Context path
<i>return</i>	Scope key as string

## 27.14. getMappingTable()

```
public java.util.Map<java.lang.String, java.lang.String> getMappingTable();
```

**Specified by:** Method getMappingTable in interface IServer

Return mapping

### Parameters

<i>return</i>	Map of "scope key / scope name" pairs
---------------	---------------------------------------

## 27.15. lookupGlobal(String, String)

```
public org.red5.server.api.IGlobalScope lookupGlobal(String hostName,
                                                    String contextPath);
```

**Specified by:** Method lookupGlobal in interface IServer

Does global scope lookup for host name and context path

### Parameters

hostName	Host name
contextPath	Context path
<i>return</i>	Global scope

## 27.16. notifyConnected(IConnection)

```
protected void notifyConnected(org.red5.server.api.IConnection conn);
```

Notify listeners that a new connection was established.

### Parameters

conn	the new connection
------	--------------------

## 27.17. notifyDisconnected(IConnection)

```
protected void notifyDisconnected(org.red5.server.api.IConnection conn);
```

Notify listeners that a connection was disconnected.

### Parameters

conn	the disconnected connection
------	-----------------------------

## 27.18. notifyScopeCreated(IScope)

```
protected void notifyScopeCreated(org.red5.server.api.IScope scope);
```

Notify listeners about a newly created scope.

### Parameters

scope	the scope that was created
-------	----------------------------

## 27.19. notifyScopeRemoved(IScope)

```
protected void notifyScopeRemoved(org.red5.server.api.IScope scope);
```

Notify listeners that a scope was removed.

### Parameters

scope	the scope that was removed
-------	----------------------------

## 27.20. registerGlobal(IGlobalScope)

```
public void registerGlobal(org.red5.server.api.IGlobalScope scope);
```

**Specified by:** Method registerGlobal in interface IServer

Register global scope

### Parameters

scope	Global scope to register
-------	--------------------------

## 27.21. removeListener(IConnectionListener)

```
public void removeListener(org.red5.server.api.listeners.IConnectionListener listener);
```

**Specified by:** Method removeListener in interface IServer

Remove listener that got notified about connection events.

## 27.22. removeListener(IScopeListener)

```
public void removeListener(org.red5.server.api.listeners.IScopeListener listener);
```

**Specified by:** Method removeListener in interface IServer

Remove listener that got notified about scope events.

## 27.23. removeMapping(String, String)

```
public boolean removeMapping(String hostName,
                             String contextPath);
```

**Specified by:** Method removeMapping in interface IServer

Remove mapping with given key

Parameters	
hostName	Host name
contextPath	Context path
return	true if mapping was removed, false if key doesn't exist

## 27.24. setApplicationContext(ApplicationContext)

```
public void setApplicationContext(org.springframework.context.ApplicationContext applicationContext)
```

**Specified by:** Method setApplicationContext in interface ApplicationContextAware

Setter for Spring application context

Parameters	
applicationContext	Application context

## 27.25. toString()

```
public String toString();
```

String representation of server

Parameters	
return	String representation of server

# 28. Class Shutdown

Provides a means to cleanly shutdown an instance from the command line.

## 28.1. Synopsis

```
public class Shutdown {
// Public Constructors

    public Shutdown();

// Public Static Methods
```

```
public static void main(String[] args);
}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 28.2. main(String[])

```
public static void main(String[] args);
```

Connects to the given RMI port (default: 9999) and invokes shutdown on the loader.

### Parameters

args	The first parameter should be a port number
------	---

## 29. Class Standalone

Entry point from which the server config file is loaded.

### 29.1. Synopsis

```
public class Standalone {
    // Protected Fields

    protected static org.slf4j.Logger log ;

    protected static String red5Config ;

    // Public Constructors

    public Standalone();

    // Public Static Methods

    public static void main(String[] args)
        throws Throwable;

    public static void raiseOriginalException(Throwable e)
        throws Throwable;

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 29.2. log

```
protected static org.slf4j.Logger log ;
```

Initialize Logging.

## 29.3. main(String[])

```
public static void main(String[] args)
    throws Throwable;
```

Main entry point for the Red5 Server usage Java Standalone.

### Parameters

args	String passed in that points to a red5.xml config file
------	--

Throwable

Base type of all exceptions

## 29.4. raiseOriginalException(Throwable)

```
public static void raiseOriginalException(Throwable e)
    throws Throwable;
```

Re-throws exception

### Parameters

e	Exception
---	-----------

Throwable

Re-thrown exception

# 30. Class WebScope

Web scope is special scope that is aware of servlet context and represents scope of Red5 application in servlet container (or application server) like Tomcat, Jetty or JBoss. Web scope is aware of virtual hosts configuration for Red5 application and is the first scope that instantiated after Red5 application gets started. Then it loads virtual hosts configuration, adds mappings of paths to global scope that is injected thru Spring IoC context file and runs initialization process. Red5 server implementation instance and ServletContext are injected as well.

## 30.1. Synopsis

```
public class WebScope extends, org.?red5.?server.?Scope
    implements, org.?springframework.?web.?context.?ServletContextAware {
// Protected Fields

    protected org.red5.server.api.IApplicationContext appContext ;

    protected org.red5.server.api.IApplicationLoader appLoader ;

    protected String contextPath ;

    protected String[] hostnames ;

    protected static org.slf4j.Logger log ;
```

```

protected boolean registered ;

protected org.red5.server.api.IServer server ;

protected javax.servlet.ServletContext servletContext ;

protected boolean shuttingDown ;

protected String virtualHosts ;

// Public Constructors

public WebScope();

// Public Methods

public org.red5.server.api.IApplicationLoader getApplicationLoader();

public org.red5.server.api.IServer getServer();

public boolean isShuttingDown();

public synchronized void register();

public void setContextPath(String contextPath);

public void setGlobalScope(org.red5.server.api.IGlobalScope globalScope);

public void setName();

public void setParent();

public void setServer(org.red5.server.api.IServer server);

public void setServletContext(javax.servlet.ServletContext servletContext);

public void setVirtualHosts(String virtualHosts);

public synchronized void unregister();

}

```

**Methods inherited from org.red5.server.Scope:** addChildScope , connect , createChildScope , destroy , disconnect , dispatchEvent , getActiveClients , getActiveConnections , getActiveSubscopes , getBasicScope , getBasicScopeNames , getClassLoader , getClients , getConnections , getContext , getContextPath , getCreationTime , getDepth , getEnabled , getHandler , getMaxClients , getMaxConnections , getMaxSubscopes , getParent , getPath , getResource , getResources , getRunning , getScope , getScopeNames , getServer , getServiceHandler , getServiceHandlerNames , getServiceHandlers , getStatistics , getTotalClients , getTotalConnections , getTotalSubscopes , handleEvent , hasChildScope , hasContext , hasHandler , hasParent , init , isEnabled , isRunning , iterator , lookupConnections , registerServiceHandler , removeChildScope , setAutoStart , setChildLoadPath , setContext , setDepth , setEnabled , setHandler , setName , setParent , setPersistenceClass , start , stop , toString , uninit , unregisterServiceHandler

**Methods inherited from org.red5.server.BasicScope:** addEventListener ,  
getEventListeners , notifyEvent , removeEventListener

**Methods inherited from org.red5.server.PersistableAttributeStore:** deserialize  
, getAttribute , getLastModified , getName , getStore , getType , isPersistent  
, modified , removeAttribute , removeAttributes , serialize , setAttribute ,  
setAttributes , setPath , setPersistent , setStore

**Methods inherited from org.red5.server.AttributeStore:** filterNull ,  
getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute ,  
getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute ,  
getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute ,  
hasAttribute

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , wait

**Fields inherited from org.red5.server.Scope:** clientStats , connectionStats , log ,  
oName , subscopeStats

**Fields inherited from org.red5.server.BasicScope:** keepOnDisconnect , listeners ,  
parent , persistenceClass

**Fields inherited from org.red5.server.PersistableAttributeStore:** lastModified , name  
, path , persistent , store , type

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 30.2. appContext

```
protected org.red5.server.api.IApplicationContext appContext ;
```

The application context this webscope is running in.

## 30.3. appLoader

```
protected org.red5.server.api.IApplicationLoader appLoader ;
```

Loader for new applications.

## 30.4. contextPath

```
protected String contextPath ;
```

Context path

## 30.5. hostnames

```
protected String[] hostnames ;
```

Hostnames

## 30.6. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 30.7. registered

```
protected boolean registered ;
```

Has the web scope been registered?

## 30.8. server

```
protected org.red5.server.api.IServer server ;
```

Server instance

## 30.9. servletContext

```
protected javax.servlet.ServletContext servletContext ;
```

Servlet context

## 30.10. shuttingDown

```
protected boolean shuttingDown ;
```

Is the scope currently shutting down?

## 30.11. virtualHosts

```
protected String virtualHosts ;
```

Virtual hosts list as string

## 30.12. getApplicationLoader()

```
public org.red5.server.api.IApplicationLoader getApplicationLoader();
```

Return object that can be used to load new applications.

### Parameters

<i>return</i>	the application loader
---------------	------------------------

## 30.13. getServer()

```
public org.red5.server.api.IServer getServer();
```

## 30.14. isShuttingDown()

```
public boolean isShuttingDown();
```

Is the scope currently shutting down?

#### Parameters

<i>return</i>
---------------

### 30.15. register()

public synchronized void register();
--------------------------------------

Map all vhosts to global scope then initialize

### 30.16. setContextPath(String)

public void setContextPath(String contextPath);
---

Setter for context path

#### Parameters

contextPath	Context path
-------------	--------------

### 30.17. setGlobalScope(IGlobalScope)

public void setGlobalScope(org.red5.server.api.IGlobalScope globalScope);
---

Setter for global scope. Sets persistence class.

#### Parameters

globalScope	Red5 global scope
-------------	-------------------

### 30.18. setName()

public void setName();
------------------------

Web scope has no name

### 30.19. setParent()

public void setParent();
--------------------------

Can't set parent to Web scope. Web scope is top level.

### 30.20. setServer(I Server)

public void setServer(org.red5.server.api.I Server server);
---

Setter for server

#### Parameters

server	Server instance
--------	-----------------

### 30.21. setServletContext(ServletContext)

```
public void setServletContext(javax.servlet.ServletContext servletContext);
```

**Specified by:** Method `setServletContext` in interface `ServletContextAware`

Servlet context

Parameters

servletContext	Servlet context
----------------	-----------------

### 30.22. setVirtualHosts(String)

```
public void setVirtualHosts(String virtualHosts);
```

Setter for virtual hosts. Creates array of hostnames.

Parameters

virtualHosts	Virtual hosts list as string
--------------	------------------------------

### 30.23. unregister()

```
public synchronized void unregister();
```

Uninitialize and remove all vhosts from the global scope.

# 1. Class AbstractScopeAdapter

Base scope handler implementation. Meant to be subclassed.

## 1.1. Synopsis

```
public abstract class AbstractScopeAdapter implements org.red5.server.api.IScopeHandler {  
    // Public Constructors  
  
    public AbstractScopeAdapter();  
  
    // Public Methods  
  
    public boolean addChildScope(org.red5.server.api.IBasicScope scope);  
  
    public boolean connect(org.red5.server.api.IConnection conn,  
                          org.red5.server.api.IScope scope,  
                          Object[] params);  
  
    public void disconnect(org.red5.server.api.IConnection conn,  
                          org.red5.server.api.IScope scope);  
  
    public boolean handleEvent(org.red5.server.api.event.IEvent event);  
  
    public boolean join(org.red5.server.api.IClient client,  
                      org.red5.server.api.IScope scope);  
  
    public void leave(org.red5.server.api.IClient client,  
                     org.red5.server.api.IScope scope);  
  
    public void removeChildScope(org.red5.server.api.IBasicScope scope);  
  
    public boolean serviceCall(org.red5.server.api.IConnection conn,  
                             org.red5.server.api.service.IServiceCall call);  
  
    public void setCanCallService(boolean canCallService);  
  
    public void setCanConnect(boolean canConnect);  
  
    public void setCanStart(boolean canStart);  
  
    public void setJoin(boolean canJoin);  
  
    public boolean start(org.red5.server.api.IScope scope);  
  
    public void stop(org.red5.server.api.IScope scope);  
}
```

**Direct known subclasses:** org.red5.server.adapter.StatefulScopeWrappingAdapter

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. addChildScope(IBasicScope)

```
public boolean addChildScope(org.red5.server.api.IBasicScope scope);
```

**Specified by:** Method addChildScope in interface IScopeHandler

Called just before a child scope is added.

## 1.3. connect(IConnection, IScope, Object[])

```
public boolean connect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope,
                      Object[] params);
```

**Specified by:** Method connect in interface IScopeHandler

Called just before every connection to a scope. You can pass additional params from client using `NetConnection.connect` method (see below).

## 1.4. disconnect(IConnection, IScope)

```
public void disconnect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope);
```

**Specified by:** Method disconnect in interface IScopeHandler

Called just after the a connection is disconnected.

## 1.5. handleEvent(IEvent)

```
public boolean handleEvent(org.red5.server.api.event.IEvent event);
```

## 1.6. join(IClient, IScope)

```
public boolean join(org.red5.server.api.IClient client,
                   org.red5.server.api.IScope scope);
```

**Specified by:** Method join in interface IScopeHandler

Called just before a client enters the scope.

## 1.7. leave(IClient, IScope)

```
public void leave(org.red5.server.api.IClient client,
                  org.red5.server.api.IScope scope);
```

**Specified by:** Method leave in interface IScopeHandler

Called just after the client leaves the scope.

## 1.8. removeChildScope(IBasicScope)

```
public void removeChildScope(org.red5.server.api.IBasicScope scope);
```

**Specified by:** Method removeChildScope in interface IScopeHandler

Called just after a child scope has been removed.

## 1.9. serviceCall(IConnection, IServiceCall)

```
public boolean serviceCall(org.red5.server.api.IConnection conn,
                         org.red5.server.api.service(IServiceCall call);
```

**Specified by:** Method serviceCall in interface IScopeHandler

Called when a service is called.

## 1.10. setCanCallService(boolean)

```
public void setCanCallService(boolean canCallService);
```

Setter for can call service flag

### Parameters

canCallService	true if remote service calls are allowed for the scope, false otherwise
----------------	---

## 1.11. setCanConnect(boolean)

```
public void setCanConnect(boolean canConnect);
```

Setter for can connect flag

### Parameters

canConnect	true if connections to scope are allowed, false otherwise
------------	---

## 1.12. setCanStart(boolean)

```
public void setCanStart(boolean canStart);
```

Setter for can start flag.

### Parameters

canStart	true if scope is ready to be activated, false otherwise
----------	---

## 1.13. setJoin(boolean)

```
public void setJoin(boolean canJoin);
```

Setter for 'can join' flag

### Parameters

canJoin	true if scope may be joined by users, false otherwise
---------	---

## 1.14. start(IScope)

```
public boolean start(org.red5.server.api.IScope scope);
```

**Specified by:** Method start in interface IScopeHandler

Called when a scope is created for the first time.

## 1.15. stop(IScope)

```
public void stop(org.red5.server.api.IScope scope);
```

**Specified by:** Method stop in interface IScopeHandler

Called just before a scope is disposed.

## 2. Class ApplicationAdapter

Base class for applications, takes care that callbacks are executed single-threaded. If you want to have maximum performance, use `org.red5.server.adapter.MultiThreadedApplicationAdapter` instead. Using this class may lead to problems if accepting a client in the \*Connect or \*Join methods takes too long, so using the multi-threaded version is preferred.

### 2.1. Synopsis

```
public class ApplicationAdapter extends org.?red5.?server.?adapter.?MultiThreadedApplicationAdapter
// Protected Fields
protected static org.slf4j.Logger log ;
// Public Constructors
public ApplicationAdapter();
// Public Methods
public synchronized boolean connect(org.red5.server.api.IConnection conn,
                                    org.red5.server.api.IScope scope,
                                    Object[] params);
public synchronized void disconnect(org.red5.server.api.IConnection conn,
                                    org.red5.server.api.IScope scope);
public synchronized boolean join(org.red5.server.api.IClient client,
                               org.red5.server.api.IScope scope);
public synchronized void leave(org.red5.server.api.IClient client,
                             org.red5.server.api.IScope scope);
public synchronized boolean start(org.red5.server.api.IScope scope);
public synchronized void stop(org.red5.server.api.IScope scope);
}
```

**Methods inherited from org.red5.server.adapter.MultiThreadedApplicationAdapter:**

addListener , addScheduledJob , addScheduledJobAfterDelay , addScheduledOnceJob ,  
, appConnect , appDisconnect , appJoin , appLeave , appStart , appStop ,  
cancelGhostConnectionsCleanup , clearSharedObjects , connect , createSharedObject ,  
disconnect , FCPublish , FCUnpublish , getBroadcastStream , getBroadcastStreamNames  
, getClientTTL , getGhostConnsCleanupPeriod , getListeners , getOnDemandStream  
, getScheduledJobNames , getSharedObject , getSharedObjectName ,  
getSharedObjectSecurity , getStreamLength , getStreamPlaybackSecurity

```
, getStreamPublishSecurity , getSubscriberStream , hasBroadcastStream ,
hasOnDemandStream , hasSharedObject , join , killGhostConnections , leave ,
measureBandwidth , registerSharedObjectSecurity , registerStreamPlaybackSecurity ,
registerStreamPublishSecurity , rejectClient , removeListener , removeScheduledJob
, roomConnect , roomDisconnect , roomJoin , roomLeave , roomStart , roomStop ,
scheduleGhostConnectionsCleanup , setClientTTL , setGhostConnsCleanupPeriod , start
, stop , streamBroadcastClose , streamBroadcastStart , streamPlaylistItemPlay ,
streamPlaylistItemStop , streamPlaylistVODItemPause , streamPlaylistVODItemResume
, streamPlaylistVODItemSeek , streamPublishStart , streamRecordStart ,
streamSubscriberClose , streamSubscriberStart , unregisterSharedObjectSecurity ,
unregisterStreamPlaybackSecurity , unregisterStreamPublishSecurity
```

**Methods inherited from org.red5.server.adapter.StatefulScopeWrappingAdapter:**

```
createChildScope , getAttribute , getAttributeNames , getAttributes , getChildScope
, getChildScopeNames , getClients , getConnectionsIter , getContext , getDepth ,
getName , getParent , getPath , getResource , getResources , getScope , hasAttribute
, hasChildScope , hasParent , lookupConnections , removeAttribute , removeAttributes
, setAttribute , setAttributes , setScope
```

**Methods inherited from org.red5.server.adapter.AbstractScopeAdapter:**

```
addChildScope , handleEvent , removeChildScope , serviceCall , setCanCallService ,
setCanConnect , setCanStart , setJoin
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.adapter.MultiThreadedApplicationAdapter:** log  
, schedulingService

**Fields inherited from org.red5.server.adapter.StatefulScopeWrappingAdapter:** scope

## 2.2. log

```
protected static org.slf4j.Logger log ;
```

Logger object.

## 2.3. connect(IConnection, IScope, Object[])

```
public synchronized boolean connect(org.red5.server.api.IConnection conn,
org.red5.server.api.IScope scope,
Object[] params);
```

## 2.4. disconnect(IConnection, IScope)

```
public synchronized void disconnect(org.red5.server.api.IConnection conn,
org.red5.server.api.IScope scope);
```

## 2.5. join(IClient, IScope)

```
public synchronized boolean join(org.red5.server.api.IClient client,
org.red5.server.api.IScope scope);
```

## 2.6. leave(IClient, IScope)

```
public synchronized void leave(org.red5.server.api.IClient client,
                           org.red5.server.api.IScope scope);
```

## 2.7. start(IScope)

```
public synchronized boolean start(org.red5.server.api.IScope scope);
```

## 2.8. stop(IScope)

```
public synchronized void stop(org.red5.server.api.IScope scope);
```

# 3. Interface ApplicationMBean

JMX mbean for Application.

## 3.1. Synopsis

```
public interface ApplicationMBean {
    // Public Methods

    public boolean appConnect(org.red5.server.api.IConnection conn,
                           Object[] params);

    public void appDisconnect(org.red5.server.api.IConnection conn);

    public boolean appJoin(org.red5.server.api.IClient client,
                         org.red5.server.api.IScope app);

    public void appLeave(org.red5.server.api.IClient client,
                       org.red5.server.api.IScope app);

    public boolean appStart(org.red5.server.api.IScope app);

    public void appStop(org.red5.server.api.IScope app);

    public boolean roomConnect(org.red5.server.api.IConnection conn,
                           Object[] params);

    public void roomDisconnect(org.red5.server.api.IConnection conn);

    public boolean roomJoin(org.red5.server.api.IClient client,
                         org.red5.server.api.IScope room);

    public void roomLeave(org.red5.server.api.IClient client,
                         org.red5.server.api.IScope room);

    public boolean roomStart(org.red5.server.api.IScope room);

    public void roomStop(org.red5.server.api.IScope room);

}
```

## 4. Interface IApplication

IApplication provides lifecycle methods that most communication applications will use. This interface defines the methods that are called by Red5 through an applications life.

### 4.1. Synopsis

```
public interface IApplication {
// Public Methods

    public boolean appConnect(org.red5.server.api.IConnection conn,
                           Object[] params);

    public void appDisconnect(org.red5.server.api.IConnection conn);

    public boolean appJoin(org.red5.server.api.IClient client,
                          org.red5.server.api.IScope app);

    public void appLeave(org.red5.server.api.IClient client,
                        org.red5.server.api.IScope app);

    public boolean appStart(org.red5.server.api.IScope app);

    public void appStop(org.red5.server.api.IScope app);

    public boolean roomConnect(org.red5.server.api.IConnection conn,
                           Object[] params);

    public void roomDisconnect(org.red5.server.api.IConnection conn);

    public boolean roomJoin(org.red5.server.api.IClient client,
                          org.red5.server.api.IScope room);

    public void roomLeave(org.red5.server.api.IClient client,
                         org.red5.server.api.IScope room);

    public boolean roomStart(org.red5.server.api.IScope room);

    public void roomStop(org.red5.server.api.IScope room);
}
```

### 4.2. appConnect(IConnection, Object[])

```
public boolean appConnect(org.red5.server.api.IConnection conn,
                           Object[] params);
```

Called per each client connect

Parameters	
conn	Connection object used to provide basic connection methods. See <a href="#">org.red5.server.api.IConnection</a>
params	List of params sent from client with NetConnection.connect call

`return``true` accepts the connection, `false` rejects it

### 4.3. appDisconnect(IConnection)

```
public void appDisconnect(org.red5.server.api.IConnection conn);
```

Called every time client disconnects from the application

#### Parameters

`conn`Connection object See [org.red5.server.api.IConnection](#)

### 4.4. appJoin(IClient, IScope)

```
public boolean appJoin(org.red5.server.api.IClient client,
                      org.red5.server.api.IScope app);
```

Called every time client joins app level scope

#### Parameters

`client`

Client object

`app`

Scope object

`return``true` accepts the client, `false` rejects it

### 4.5. appLeave(IClient, IScope)

```
public void appLeave(org.red5.server.api.IClient client,
                     org.red5.server.api.IScope app);
```

Called every time client leaves the application scope

#### Parameters

`client`

Client object

`app`

Scope object

### 4.6. appStart(IScope)

```
public boolean appStart(org.red5.server.api.IScope app);
```

Called once when application or room starts

#### Parameters

`app`Application or room level scope. See [org.red5.server.api.IScope](#) for details`return``true` continues application run, `false` terminates

### 4.7. appStop(IScope)

```
public void appStop(org.red5.server.api.IScope app);
```

Called on application stop

#### Parameters

app	Scope object
-----	--------------

## 4.8. roomConnect(IConnection, Object[])

```
public boolean roomConnect(org.red5.server.api.IConnection conn,
                         Object[] params);
```

Called every time client connects to the room

#### Parameters

conn	Connection object
params	List of params sent from client with NetConnection.connect call
return	true accepts the connection, false rejects it

## 4.9. roomDisconnect(IConnection)

```
public void roomDisconnect(org.red5.server.api.IConnection conn);
```

Called when client disconnects from room scope

#### Parameters

conn	Connection object used to provide basic connection methods. See <a href="#">org.red5.server.api.IConnection</a>
------	--

## 4.10. roomJoin(IClient, IScope)

```
public boolean roomJoin(org.red5.server.api.IClient client,
                       org.red5.server.api.IScope room);
```

Called when user joins room scope

#### Parameters

client	Client object
room	Scope object
return	true accepts the client, false rejects it

## 4.11. roomLeave(IClient, IScope)

```
public void roomLeave(org.red5.server.api.IClient client,
                      org.red5.server.api.IScope room);
```

Called when user leaves room scope

#### Parameters

client	Client object
--------	---------------

room	Scope object
------	--------------

## 4.12. roomStart(IScope)

```
public boolean roomStart(org.red5.server.api.IScope room);
```

Called on application room start

### Parameters

room	Scope object
------	--------------

return	true if scope can be started, false otherwise
--------	---

## 4.13. roomStop(IScope)

```
public void roomStop(org.red5.server.api.IScope room);
```

Called on room scope stop

### Parameters

room	Scope object
------	--------------

## 5. Class MultiThreadedApplicationAdapter

ApplicationAdapter class serves as a base class for your Red5 applications. It provides methods to work with SharedObjects and streams, as well as connections and scheduling services. ApplicationAdapter is an application level IScope.

To handle streaming processes in your application you should implement

`org.red5.server.api.stream.IStreamAwareScopeHandler` interface and implement handling methods. Application adapter provides you with useful event handlers that can be used to intercept streams, authorize users, etc. Also, all methods added in subclasses can be called from client side with `NetConnection.call` method. Unlike to Flash Media server which requires you to keep methods on Client object at server side, Red5 offers much more convenient way to add methods for remote invocation to your applications.

### EXAMPLE:

```
public List getLiveStreams() { // Implementation goes here, say, use Red5 object to obtain scope and all it's streams }
```

This method added to ApplicationAdapter subclass can be called from client side with the following code:

```
var nc:NetConnection = new NetConnection();

nc.connect(...);

nc.call("getLiveStreams", resultHandlerObj);
```

If you want to build a server-side framework this is a place to start and wrap it around ApplicationAdapter subclass.

## 5.1. Synopsis

```

public class MultiThreadedApplicationAdapter extends, org.?red5.?server.?adapter.?StatefulScopeWrapping
    implements, org.?red5.?server.?api.?so.?ISharedObjectService, org.?red5.?server.?api.?stream.?IBro
// Protected Fields

    protected static org.slf4j.Logger log ;

    protected org.red5.server.api.scheduling.ISchedulingService schedulingService ;

// Public Constructors

    public MultiThreadedApplicationAdapter();

// Public Methods

    public void FCPublish(String streamName);

    public void FCUnpublish();

    public void addListener(IApplication listener);

    public String addScheduledJob(int interval,
                                  org.red5.server.api.scheduling.IScheduledJob job);

    public String addScheduledJobAfterDelay(int interval,
                                            org.red5.server.api.scheduling.IScheduledJob job,
                                            int delay);

    public String addScheduledOnceJob(java.util.Date date,
                                      org.red5.server.api.scheduling.IScheduledJob job);

    public String addScheduledOnceJob(long timeDelta,
                                      org.red5.server.api.scheduling.IScheduledJob job);

    public boolean appConnect(org.red5.server.api.IConnection conn,
                            Object[] params);

    public void appDisconnect(org.red5.server.api.IConnection conn);

    public boolean appJoin(org.red5.server.api.IClient client,
                          org.red5.server.api.IScope app);

    public void appLeave(org.red5.server.api.IClient client,
                        org.red5.server.api.IScope app);

    public boolean appStart(org.red5.server.api.IScope app);

    public void appStop(org.red5.server.api.IScope app);

    public void cancelGhostConnectionsCleanup();

    public boolean clearSharedObjects(org.red5.server.api.IScope scope,
                                    String name);

    public boolean connect(org.red5.server.api.IConnection conn,
                          org.red5.server.api.IScope scope,
                          Object[] params);

```

## Package org.red5.server.adapter

```
public boolean createSharedObject(org.red5.server.api.IScope scope,
                                String name,
                                boolean persistent);

public void disconnect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope);

public org.red5.server.api.stream.IBroadcastStream getBroadcastStream(org.red5.server.api.IScope scope,
                                                                     String name);

public java.util.List<java.lang.String> getBroadcastStreamNames(org.red5.server.api.IScope scope);

public long getClientTTL();

public int getGhostConnsCleanupPeriod();

public java.util.Set<org.red5.server.adapter.IApplication> getListeners();

public org.red5.server.api.stream.IOnDemandStream getOnDemandStream(org.red5.server.api.IScope scope,
                                                                    String name);

public java.util.List<java.lang.String> getScheduledJobNames();

public org.red5.server.api.so.ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                                             String name);

public org.red5.server.api.so.ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                                             String name,
                                                             boolean persistent);

public java.util.Set<java.lang.String> getSharedObjectNamees(org.red5.server.api.IScope scope);

public java.util.Set<org.red5.server.api.so.ISharedObjectSecurity> getSharedObjectSecurity();

public double getStreamLength(String name);

public java.util.Set<org.red5.server.api.stream.IStreamPlaybackSecurity> getStreamPlaybackSecurity();

public java.util.Set<org.red5.server.api.stream.IStreamPublishSecurity> getStreamPublishSecurity();

public org.red5.server.api.stream.ISubscriberStream getSubscriberStream(org.red5.server.api.IScope scope,
                                                                       String name);

public boolean hasBroadcastStream(org.red5.server.api.IScope scope,
                                 String name);

public boolean hasOnDemandStream(org.red5.server.api.IScope scope,
                                 String name);

public boolean hasSharedObject(org.red5.server.api.IScope scope,
                             String name);

public boolean join(org.red5.server.api.IClient client,
                   org.red5.server.api.IScope scope);

public void leave(org.red5.server.api.IClient client,
                  org.red5.server.api.IScope scope);
```

## Package org.?red5.?server.?adapter

```
public void measureBandwidth();

public void measureBandwidth(org.red5.server.api.IConnection conn);

public void registerSharedObjectSecurity(org.red5.server.api.so.ISharedObjectSecurity handler);

public void registerStreamPlaybackSecurity(org.red5.server.api.stream.IStreamPlaybackSecurity handl

public void registerStreamPublishSecurity(org.red5.server.api.stream.IStreamPublishSecurity handle

public void removeListener(IApplication listener);

public void removeScheduledJob(String name);

public boolean roomConnect(org.red5.server.api.IConnection conn,
                           Object[] params);

public void roomDisconnect(org.red5.server.api.IConnection conn);

public boolean roomJoin(org.red5.server.api.IClient client,
                       org.red5.server.api.IScope room);

public void roomLeave(org.red5.server.api.IClient client,
                      org.red5.server.api.IScope room);

public boolean roomStart(org.red5.server.api.IScope room);

public void roomStop(org.red5.server.api.IScope room);

public void scheduleGhostConnectionsCleanup();

public void setClientTTL(int clientTTL);

public void setGhostConnsCleanupPeriod(int ghostConnsCleanupPeriod);

public boolean start(org.red5.server.api.IScope scope);

public void stop(org.red5.server.api.IScope scope);

public void streamBroadcastClose(org.red5.server.api.stream.IBroadcastStream stream);

public void streamBroadcastStart(org.red5.server.api.stream.IBroadcastStream stream);

public void streamPlaylistItemPlay(org.red5.server.api.stream.IPlaylistSubscriberStream stream,
                                   org.red5.server.api.stream.IPlayItem item,
                                   boolean isLive);

public void streamPlaylistItemStop(org.red5.server.api.stream.IPlaylistSubscriberStream stream,
                                   org.red5.server.api.stream.IPlayItem item);

public void streamPlaylistVODItemPause(org.red5.server.api.stream.IPlaylistSubscriberStream stream,
                                       org.red5.server.api.stream.IPlayItem item,
                                       int position);

public void streamPlaylistVODItemResume(org.red5.server.api.stream.IPlaylistSubscriberStream stream,
                                       org.red5.server.api.stream.IPlayItem item,
```

## Package org.?red5.?server.?adapter

```
        int position);  
  
    public void streamPlaylistVODItemSeek(org.red5.server.api.stream.IPlaylistSubscriberStream stream,  
                                         org.red5.server.api.stream.IPlayItem item,  
                                         int position);  
  
    public void streamPublishStart(org.red5.server.api.stream.IBroadcastStream stream);  
  
    public void streamRecordStart(org.red5.server.api.stream.IBroadcastStream stream);  
  
    public void streamSubscriberClose(org.red5.server.api.stream.ISubscriberStream stream);  
  
    public void streamSubscriberStart(org.red5.server.api.stream.ISubscriberStream stream);  
  
    public void unregisterSharedObjectSecurity(org.red5.server.api.so.ISharedObjectSecurity handler);  
  
    public void unregisterStreamPlaybackSecurity(org.red5.server.api.stream.IStreamPlaybackSecurity han  
    public void unregisterStreamPublishSecurity(org.red5.server.api.stream.IStreamPublishSecurity hand  
  
// Protected Methods  
  
    protected void killGhostConnections();  
  
    protected boolean rejectClient()  
        throws ClientRejectedException;  
  
    protected boolean rejectClient(Object reason)  
        throws ClientRejectedException;  
}
```

**Direct known subclasses:** org.?red5.?server.?adapter.?ApplicationAdapter

**Methods inherited from org.red5.server.adapter.StatefulScopeWrappingAdapter:**

createChildScope , getAttribute , getAttributeNames , getAttributes , getChildScope ,  
getChildScopeNames , getClients , getConnectionsIter , getContext , getDepth ,  
getName , getParent , getPath , getResource , getResources , getScope , hasAttribute ,  
hasChildScope , hasParent , lookupConnections , removeAttribute , removeAttributes ,  
setAttribute , setAttributes , setScope

**Methods inherited from org.red5.server.adapter.AbstractScopeAdapter:**

addChildScope , connect , disconnect , handleEvent , join , leave , removeChildScope ,  
serviceCall , setCanCallService , setCanConnect , setCanStart , setJoin , start ,  
stop

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.adapter.StatefulScopeWrappingAdapter:** scope

## 5.2. log

```
protected static org.slf4j.Logger log ;
```

Logger object

### 5.3. schedulingService

```
protected org.red5.server.api.scheduling.ISchedulingService schedulingService ;
```

Scheduling service. Uses Quartz. Adds and removes scheduled jobs.

### 5.4. addListener(IApplication)

```
public void addListener(IApplication listener);
```

Register listener that will get notified about application events. Please note that return values (e.g. from `appStart(org.red5.server.api.IScope)`) will be ignored for listeners.

#### Parameters

listener	object to register
----------	--------------------

### 5.5. addScheduledJob(int, IScheduledJob)

```
public String addScheduledJob(int interval,
                               org.red5.server.api.scheduling.IScheduledJob job);
```

**Specified by:** Method `addScheduledJob` in interface `ISchedulingService`

Wrapper around `ISchedulingService`, adds a scheduled job to be run periodically. We store this service in the scope as it can be shared across all rooms of the applications.

#### Parameters

interval	Time interval to run the scheduled job
job	Scheduled job object
return	Name of the scheduled job

### 5.6. addScheduledJobAfterDelay(int, IScheduledJob, int)

```
public String addScheduledJobAfterDelay(int interval,
                                       org.red5.server.api.scheduling.IScheduledJob job,
                                       int delay);
```

**Specified by:** Method `addScheduledJobAfterDelay` in interface `ISchedulingService`

Adds a scheduled job which starts after the specified delay period and fires periodically.

#### Parameters

interval	time in milliseconds between two notifications of the job
job	the job to trigger periodically
delay	time in milliseconds to pass before first execution.

<i>return</i>	the name of the scheduled job
---------------	-------------------------------

## 5.7. addScheduledOnceJob(Date, IScheduledJob)

```
public String addScheduledOnceJob(java.util.Date date,
                                org.red5.server.api.scheduling.IScheduledJob job);
```

**Specified by:** Method addScheduledOnceJob in interface ISchedulingService

Adds a scheduled job that's gonna be executed once on given date. Please note that the jobs are not saved if Red5 is restarted in the meantime.

### Parameters

<i>date</i>	When to run scheduled job
<i>job</i>	Scheduled job object
<i>return</i>	Name of the scheduled job

## 5.8. addScheduledOnceJob(long, IScheduledJob)

```
public String addScheduledOnceJob(long timeDelta,
                                org.red5.server.api.scheduling.IScheduledJob job);
```

**Specified by:** Method addScheduledOnceJob in interface ISchedulingService

Adds a scheduled job that's gonna be executed once. Please note that the jobs are not saved if Red5 is restarted in the meantime.

### Parameters

<i>timeDelta</i>	Time offset in milliseconds from the current date when given job should be run
<i>job</i>	Scheduled job object
<i>return</i>	Name of the scheduled job

## 5.9. appConnect(IConnection, Object[])

```
public boolean appConnect(org.red5.server.api.IConnection conn,
                        Object[] params);
```

**Specified by:** Method appConnect in interface ApplicationMBean

Handler method. Called every time new client connects (that is, new IConnection object is created after call from a SWF movie) to the application. You override this method to pass additional data from client to server application using `NetConnection.connect` method.

### EXAMPLE:

In this simple example we pass user's skin of choice identifier from client to the server.

### Client-side:

```
NetConnection.connect("rtmp://localhost/killerred5app", "silver");
```

**Server-side:**

```
if (params.length > 0) System.out.println("Theme selected: " + params[0]);
```

**Parameters**

conn	Connection object
params	List of parameters after connection URL passed to <code>NetConnection.connect</code> method.
<i>return</i>	Boolean value

**5.10. appDisconnect(IConnection)**

```
public void appDisconnect(org.red5.server.api.IConnection conn);
```

**Specified by:** Method appDisconnect in interface ApplicationMBean

Handler method. Called every time client disconnects from the application.

**Parameters**

conn	Disconnected connection object
------	--------------------------------

**5.11. appLeave(IClient, IScope)**

```
public void appLeave(org.red5.server.api.IClient client,  
                    org.red5.server.api.IScope app);
```

**Specified by:** Method appLeave in interface ApplicationMBean

Handler method. Called every time client leaves application scope.

**Parameters**

client	Client object that left
app	Application scope

**5.12. appStart(IScope)**

```
public boolean appStart(org.red5.server.api.IScope app);
```

**Specified by:** Method appStart in interface ApplicationMBean

Called once on scope (that is, application or application room) start. You override  
`appStart(org.red5.server.api.IScope) or roomStart(org.red5.server.api.IScope) }`  
 in your application to make it act the way you want.

**Parameters**

app	Application scope object
-----	--------------------------

**return**

true if scope can be started, false otherwise

## 5.13. appStop(IScope)

```
public void appStop(org.red5.server.api.IScope app);
```

**Specified by:** Method appStop in interface ApplicationMBean

Handler method. Called when application is stopped.

### Parameters

app	Scope object
-----	--------------

## 5.14. cancelGhostConnectionsCleanup()

```
public void cancelGhostConnectionsCleanup();
```

Cancel ghost connections cleanup period

## 5.15. clearSharedObjects(IScope, String)

```
public boolean clearSharedObjects(org.red5.server.api.IScope scope,
                                 String name);
```

**Specified by:** Method clearSharedObjects in interface ISharedObjectService

Deletes persistent shared objects specified by name and clears all properties from active shared objects (persistent and nonpersistent). The name parameter specifies the name of a shared object, which can include a slash (/) as a delimiter between directories in the path. The last element in the path can contain wildcard patterns (for example, a question mark [?] and an asterisk [\*]) or a shared object name. The clearSharedObjects() method traverses the shared object hierarchy along the specified path and clears all the shared objects. Specifying a slash (/) clears all the shared objects associated with an application instance.

The following values are possible for the soPath parameter:

/ clears all local and persistent shared objects associated with the instance.

/foo/bar clears the shared object /foo/bar; if bar is a directory name, no shared objects are deleted.

/foo/bar/\* clears all shared objects stored under the instance directory /foo/bar. The bar directory is also deleted if no persistent shared objects are in use within this namespace.

/foo/bar/XX?? clears all shared objects that begin with XX, followed by any two characters. If a directory name matches this specification, all the shared objects within this directory are cleared.

If you call the clearSharedObjects() method and the specified path matches a shared object that is currently active, all its properties are deleted, and a "clear" event is sent to all subscribers of the shared object. If it is a persistent shared object, the persistent store is also cleared.

## 5.16. connect(IConnection, IScope, Object[])

```
public boolean connect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope,
                      Object[] params);
```

Returns connection result for given scope and parameters. Whether the scope is room or app level scope, this method distinguishes it and acts accordingly. You override `appConnect(org.red5.server.api.IConnection, java.lang.Object[])` or `roomConnect(org.red5.server.api.IConnection, java.lang.Object[])` in your application to make it act the way you want.

Parameters	
conn	Connection object
scope	Scope
params	List of params passed to connection handler
<i>return</i>	<code>true</code> if connect is successful, <code>false</code> otherwise

## 5.17. createSharedObject(IScope, String, boolean)

```
public boolean createSharedObject(org.red5.server.api.IScope scope,
                                 String name,
                                 boolean persistent);
```

**Specified by:** Method `createSharedObject` in interface `ISharedObjectService`

Creates a new shared object for given scope. Server-side shared objects (also known as Remote SO) are special kind of objects those variable are synchronized between clients. To get an instance of RSO at client-side, use `SharedObject.getRemote()`. SharedObjects can be persistent and transient. Persistent RSO are statuful, i.e. store their data between sessions. If you need to store some data on server while clients go back and forth use persistent SO (just use `true`), otherwise perfer usage of transient for extra performance.

Parameters	
scope	Scope that shared object belongs to
name	Name of SharedObject
persistent	Whether SharedObject instance should be persistent or not
<i>return</i>	<code>true</code> if SO was created, <code>false</code> otherwise

## 5.18. disconnect(IConnection, IScope)

```
public void disconnect(org.red5.server.api.IConnection conn,
                      org.red5.server.api.IScope scope);
```

Returns disconnection result for given scope and parameters. Whether the scope is room or app level scope, this method distinguishes it and acts accordingly.

Parameters	

conn	Connection object
scope	Scope

## 5.19. FCPublish(String)

```
public void FCPublish(String streamName);
```

Notification method that is sent by FME just before publishing starts.

### Parameters

streamName	Name of stream that is about to be published.
------------	---

## 5.20. FCUnpublish()

```
public void FCUnpublish();
```

Notification method that is sent by FME when publishing of a stream ends.

## 5.21. getBroadcastStream(IScope, String)

```
public org.red5.server.api.stream.IBroadcastStream getBroadcastStream(org.red5.server.api.IScope scope, String name);
```

**Specified by:** Method getBroadcastStream in interface IBroadcastStreamService

Get a broadcast stream by name

## 5.22. getBroadcastStreamNames(IScope)

```
public java.util.List<java.lang.String> getBroadcastStreamNames(org.red5.server.api.IScope scope);
```

**Specified by:** Method getBroadcastStreamNames in interface IBroadcastStreamService

Returns list of stream names broadcasted in

scope

. Broadcast stream name is somewhat different from server stream name. Server stream name is just an ID assigned by Red5 to every created stream. Broadcast stream name is the name that is being used to subscribe to the stream at client side, that is, in `NetStream.play` call.

### Parameters

scope	Scope to retrieve broadcasted stream names
return	List of broadcasted stream names.

## 5.23. getClientTTL()

```
public long getClientTTL();
```

Client time to live is max allowed connection ping return time in seconds

#### Parameters

<i>return</i>	TTL value used in seconds
---------------	---------------------------

## 5.24. getGhostConnsCleanupPeriod()

```
public int getGhostConnsCleanupPeriod();
```

Return period of ghost connections cleanup task call

#### Parameters

<i>return</i>	Ghost connections cleanup period
---------------	----------------------------------

## 5.25. getListeners()

```
public java.util.Set<org.red5.server.adapter.IApplication> getListeners();
```

Return handlers that get notified about application events.

#### Parameters

<i>return</i>	list of handlers
---------------	------------------

## 5.26. getOnDemandStream(IScope, String)

```
public org.red5.server.api.stream.IOnDemandStream getOnDemandStream(org.red5.server.api.IScope scope
String name);
```

**Specified by:** Method `getOnDemandStream` in interface `IOnDemandStreamService`

Returns VOD stream with given name from specified scope.

#### Parameters

scope	Scope object
name	VOD stream name
<i>return</i>	IOnDemandStream object that represents stream that can be played on demand, seekable and so forth. See <code>org.red5.server.api.stream.IOnDemandStream</code> for details.

## 5.27. getScheduledJobNames()

```
public java.util.List<java.lang.String> getScheduledJobNames();
```

**Specified by:** Method `getScheduledJobNames` in interface `ISchedulingService`

Retuns list of scheduled job names

#### Parameters

<i>return</i>	List of scheduled job names as list of Strings.
---------------	---

## 5.28. getSharedObject(IScope, String)

```
public org.red5.server.api.so.ISharedObject getSharedObject(org.red5.server.api.IScope scope,
String name);
```

**Specified by:** Method getSharedObject in interface ISharedObjectService

Returns shared object from given scope by name.

### Parameters

scope	Scope that shared object belongs to
name	Name of SharedObject
<i>return</i>	Shared object instance with name given

## 5.29. getSharedObject(IScope, String, boolean)

```
public org.red5.server.api.so.ISharedObject getSharedObject(org.red5.server.api.IScope scope,
String name,
boolean persistent);
```

**Specified by:** Method getSharedObject in interface ISharedObjectService

Returns shared object from given scope by name.

### Parameters

scope	Scope that shared object belongs to
name	Name of SharedObject
persistent	Whether SharedObject instance should be persistent or not
<i>return</i>	Shared object instance with name given

## 5.30. getSharedObjectNames(IScope)

```
public java.util.Set<java.lang.String> getSharedObjectNames(org.red5.server.api.IScope scope);
```

**Specified by:** Method getSharedObjectNames in interface ISharedObjectService

Returns available SharedObject names as List

### Parameters

scope	Scope that SO belong to
-------	-------------------------

## 5.31. getSharedObjectSecurity()

```
public java.util.Set<org.red5.server.api.so.ISharedObjectSecurity> getSharedObjectSecurity();
```

**Specified by:** Method getSharedObjectSecurity in interface ISharedObjectSecurityService

Get handlers that protect shared objects.

## 5.32. getStreamLength(String)

```
public double getStreamLength(String name);
```

Returns stream length. This is a hook so it may be removed.

### Parameters

name	Stream name
<i>return</i>	Stream length in seconds (?)

## 5.33. getStreamPlaybackSecurity()

```
public java.util.Set<org.red5.server.api.stream.IStreamPlaybackSecurity> getStreamPlaybackSecurity()
```

**Specified by:** Method getStreamPlaybackSecurity in interface IStreamSecurityService

Get handlers that protect stream playback.

## 5.34. getStreamPublishSecurity()

```
public java.util.Set<org.red5.server.api.stream.IStreamPublishSecurity> getStreamPublishSecurity();
```

**Specified by:** Method getStreamPublishSecurity in interface IStreamSecurityService

Get handlers that protect stream publishing.

## 5.35. getSubscriberStream(IScope, String)

```
public org.red5.server.api.stream.ISubscriberStream getSubscriberStream(org.red5.server.api.IScope scope,
String name);
```

**Specified by:** Method getSubscriberStream in interface ISubscriberStreamService

Returns subscriber stream with given name from specified scope. Subscriber stream is a stream that clients can subscribe to.

### Parameters

scope	Scope
name	Stream name
<i>return</i>	ISubscriberStream object

## 5.36. hasBroadcastStream(IScope, String)

```
public boolean hasBroadcastStream(org.red5.server.api.IScope scope,
String name);
```

**Specified by:** Method hasBroadcastStream in interface IBroadcastStreamService

Does the scope have a broadcast stream registered with a given name

### 5.37. hasOnDemandStream(IScope, String)

```
public boolean hasOnDemandStream(org.red5.server.api.IScope scope,
                                String name);
```

**Specified by:** Method hasOnDemandStream in interface IOnDemandStreamService

Check whether scope has VOD stream with given name or not

Parameters	
scope	Scope
name	VOD stream name
return	true if scope has VOD stream with given name, false otherwise.

### 5.38. hasSharedObject(IScope, String)

```
public boolean hasSharedObject(org.red5.server.api.IScope scope,
                             String name);
```

**Specified by:** Method hasSharedObject in interface ISharedObjectService

Checks whether there's a SO with given scope and name

Parameters	
scope	Scope that SO belong to
name	Name of SharedObject

### 5.39. join(IClient, IScope)

```
public boolean join(org.red5.server.api.IClient client,
                   org.red5.server.api.IScope scope);
```

Adds client to scope. Scope can be both application or room. Can be applied to both application scope and scopes of lower level. This method calls org.red5.server.adapter.MultiThreadedApplicationAdapter or org.red5.server.adapter.MultiThreadedApplicationAdapter handlers respectively.

Parameters	
client	Client object
scope	Scope object

### 5.40. killGhostConnections()

```
protected void killGhostConnections();
```

Cleans up ghost connections

## 5.41. leave(IClient, IScope)

```
public void leave(org.red5.server.api.IClient client,
                 org.red5.server.api.IScope scope);
```

Disconnects client from scope. Can be applied to both application scope and scopes of lower level. This method calls `appLeave(org.red5.server.api.IClient, org.red5.server.api.IScope)` or `roomLeave(org.red5.server.api.IClient, org.red5.server.api.IScope)` handlers respectively.

### Parameters

client	Client object
scope	Scope object

## 5.42. measureBandwidth()

```
public void measureBandwidth();
```

Try to measure bandwidth of current connection. This is required for some FLV player to work because they require the "onBWDone" method to be called on the connection.

## 5.43. measureBandwidth(IConnection)

```
public void measureBandwidth(org.red5.server.api.IConnection conn);
```

Try to measure bandwidth of given connection. This is required for some FLV player to work because they require the "onBWDone" method to be called on the connection.

### Parameters

conn	the connection to measure the bandwidth for
------	---

## 5.44. registerSharedObjectSecurity(ISharedObjectSecurity)

```
public void registerSharedObjectSecurity(org.red5.server.api.so.ISharedObjectSecurity handler);
```

**Specified by:** Method `registerSharedObjectSecurity` in interface `ISharedObjectSecurityService`

Add handler that protects shared objects.

## 5.45. registerStreamPlaybackSecurity(IStreamPlaybackSecurity)

```
public void registerStreamPlaybackSecurity(org.red5.server.api.stream.IStreamPlaybackSecurity handler);
```

**Specified by:** Method `registerStreamPlaybackSecurity` in interface `IStreamSecurityService`

Add handler that protects stream playback.

## 5.46. registerStreamPublishSecurity(IStreamPublishSecurity)

```
public void registerStreamPublishSecurity(org.red5.server.api.stream.IStreamPublishSecurity handler);
```

**Specified by:** Method registerStreamPublishSecurity in interface IStreamSecurityService

Add handler that protects stream publishing.

## 5.47. rejectClient()

```
protected boolean rejectClient()
    throws ClientRejectedException;
```

Reject the currently connecting client without a special error message. This method throws `org.red5.server.exception.ClientRejectedException` exception.

Parameters	
<i>return</i>	never returns

`org.red5.server.exception.ClientRejectedException`

Thrown when client connection must be rejected by application logic

## 5.48. rejectClient(Object)

```
protected boolean rejectClient(Object reason)
    throws ClientRejectedException;
```

Reject the currently connecting client with an error message. The passed object will be available as "application" property of the information object that is returned to the caller.

Parameters	
<i>reason</i>	Additional error message to return to client-side Flex/Flash application
<i>return</i>	never returns

`org.red5.server.exception.ClientRejectedException`

Thrown when client connection must be rejected by application logic

## 5.49. removeListener(IApplication)

```
public void removeListener(IApplication listener);
```

Unregister handler that will not get notified about application events any longer.

Parameters	
<i>listener</i>	object to unregister

## 5.50. removeScheduledJob(String)

```
public void removeScheduledJob(String name);
```

**Specified by:** Method removeScheduledJob in interface ISchedulingService

Removes scheduled job from scheduling service list

**Parameters**

name	Scheduled job name
------	--------------------

**5.51. roomConnect(IConnection, Object[])**

```
public boolean roomConnect(org.red5.server.api.IConnection conn,
                         Object[] params);
```

**Specified by:** Method roomConnect in interface ApplicationMBean

Handler method. Called every time new client connects (that is, new IConnection object is created after call from a SWF movie) to the application. You override this method to pass additional data from client to server application using `NetConnection.connect` method. See `appConnect(org.red5.server.api.IConnection, java.lang.Object[])` for code example.

**Parameters**

conn	Connection object
params	List of params passed to room scope
<i>return</i>	Boolean value

**5.52. roomDisconnect(IConnection)**

```
public void roomDisconnect(org.red5.server.api.IConnection conn);
```

**Specified by:** Method roomDisconnect in interface ApplicationMBean

Handler method. Called every time client disconnects from the room.

**Parameters**

conn	Disconnected connection object
------	--------------------------------

**5.53. roomLeave(IClient, IScope)**

```
public void roomLeave(org.red5.server.api.IClient client,
                      org.red5.server.api.IScope room);
```

**Specified by:** Method roomLeave in interface ApplicationMBean

Handler method. Called every time client leaves room scope.

**Parameters**

client	Disconnected client object
room	Room scope

**5.54. roomStart(IScope)**

```
public boolean roomStart(org.red5.server.api.IScope room);
```

**Specified by:** Method roomStart in interface ApplicationMBean

Handler method. Called when room scope is started.

#### Parameters

room	Room scope
<i>return</i>	Boolean value

### 5.55. roomStop(IScope)

```
public void roomStop(org.red5.server.api.IScope room);
```

**Specified by:** Method roomStop in interface ApplicationMBean

Handler method. Called when room scope is stopped.

#### Parameters

room	Room scope.
------	-------------

### 5.56. scheduleGhostConnectionsCleanup()

```
public void scheduleGhostConnectionsCleanup();
```

Schedules new ghost connections cleanup using current cleanup period

### 5.57. setClientTTL(int)

```
public void setClientTTL(int clientTTL);
```

Client time to live is max allowed connection ping return time in seconds

#### Parameters

clientTTL	New TTL value in seconds
-----------	--------------------------

### 5.58. setGhostConnsCleanupPeriod(int)

```
public void setGhostConnsCleanupPeriod(int ghostConnsCleanupPeriod);
```

Set new ghost connections cleanup period

#### Parameters

ghostConnsCleanupPeriod	New ghost connections cleanup period
-------------------------	--------------------------------------

### 5.59. start(IScope)

```
public boolean start(org.red5.server.api.IScope scope);
```

Starts scope. Scope can be both application or room level.

#### Parameters

scope	Scope object
-------	--------------

**return**

`true` if scope can be started, `false` otherwise. See  
`start(org.red5.server.api.IScope)` for details.

## 5.60. stop(IScope)

```
public void stop(org.red5.server.api.IScope scope);
```

Stops scope handling (that is, stops application if given scope is app level scope and stops room handling if given scope has lower scope level). This method calls `appStop(org.red5.server.api.IScope)` or `roomStop(org.red5.server.api.IScope)` handlers respectively.

### Parameters

scope	Scope to stop
-------	---------------

## 5.61. streamBroadcastClose(IBroadcastStream)

```
public void streamBroadcastClose(org.red5.server.api.stream.IBroadcastStream stream);
```

**Specified by:** Method `streamBroadcastClose` in interface `IStreamAwareScopeHandler`

Notified when a broadcaster closes.

### Parameters

stream
--------

### Description copied from interface: `streamBroadcastClose`

## 5.62. streamBroadcastStart(IBroadcastStream)

```
public void streamBroadcastStart(org.red5.server.api.stream.IBroadcastStream stream);
```

**Specified by:** Method `streamBroadcastStart` in interface `IStreamAwareScopeHandler`

Notified when a broadcaster starts.

### Parameters

stream
--------

### Description copied from interface: `streamBroadcastStart`

## 5.63. streamPlaylistItemPlay(IPlaylistSubscriberStream, IPlayItem, boolean)

```
public void streamPlaylistItemPlay(org.red5.server.api.stream.IPlaylistSubscriberStream stream,
                                  org.red5.server.api.stream.IPlayItem item,
                                  boolean isLive);
```

**Specified by:** Method `streamPlaylistItemPlay` in interface `IStreamAwareScopeHandler`

Notified when a playlist item plays.

**Parameters**

stream	
item	
isLive	TODO

**Description copied from interface: streamPlaylistItemPlay**

## 5.64. streamPlaylistItemStop(IPlaylistSubscriberStream, IPlayItem)

```
public void streamPlaylistItemStop(org.red5.server.api.stream.IPlaylistSubscriberStream stream,
                                org.red5.server.api.stream.IPlayItem item);
```

**Specified by:** Method streamPlaylistItemStop in interface IStreamAwareScopeHandler

Notified when a playlist item stops.

**Parameters**

stream	
item	

**Description copied from interface: streamPlaylistItemStop**

## 5.65. streamPlaylistVODItemPause(IPlaylistSubscriberStream, IPlayItem, int)

```
public void streamPlaylistVODItemPause(org.red5.server.api.stream.IPlaylistSubscriberStream stream,
                                       org.red5.server.api.stream.IPlayItem item,
                                       int position);
```

**Specified by:** Method streamPlaylistVODItemPause in interface IStreamAwareScopeHandler

Notified when a playlist vod item pauses.

**Parameters**

stream	
item	
position	

**Description copied from interface: streamPlaylistVODItemPause**

## 5.66. streamPlaylistVODItemResume(IPlaylistSubscriberStream, IPlayItem, int)

```
public void streamPlaylistVODItemResume(org.red5.server.api.stream.IPlaylistSubscriberStream stream,
                                       org.red5.server.api.stream.IPlayItem item,
                                       int position);
```

**Specified by:** Method streamPlaylistVODItemResume in interface IStreamAwareScopeHandler

Notified when a playlist vod item resumes.

#### Parameters

stream	
item	
position	

**Description copied from interface: streamPlaylistVODItemResume**

## 5.67. streamPlaylistVODItemSeek(IPlaylistSubscriberStream, IPlayItem, int)

```
public void streamPlaylistVODItemSeek(org.red5.server.api.stream.IPlaylistSubscriberStream stream,
                                      org.red5.server.api.stream.IPlayItem item,
                                      int position);
```

**Specified by:** Method streamPlaylistVODItemSeek in interface IStreamAwareScopeHandler

Notified when a playlist vod item seeks.

#### Parameters

stream	
item	
position	

**Description copied from interface: streamPlaylistVODItemSeek**

## 5.68. streamPublishStart(IBroadcastStream)

```
public void streamPublishStart(org.red5.server.api.stream.IBroadcastStream stream);
```

**Specified by:** Method streamPublishStart in interface IStreamAwareScopeHandler

A broadcast stream starts being published. This will be called when the first video packet has been received.

#### Parameters

stream	
--------	--

**Description copied from interface: streamPublishStart**

## 5.69. streamRecordStart(IBroadcastStream)

```
public void streamRecordStart(org.red5.server.api.stream.IBroadcastStream stream);
```

**Specified by:** Method streamRecordStart in interface IStreamAwareScopeHandler

A broadcast stream starts being recorded. This will be called when the first video packet has been received.

Parameters

stream	
--------	--

Description copied from interface: **streamRecordStart**

## 5.70. streamSubscriberClose(ISubscriberStream)

public void streamSubscriberClose(org.red5.server.api.stream.ISubscriberStream stream);
---

**Specified by:** Method streamSubscriberClose in interface IStreamAwareScopeHandler

Notified when a subscriber closes.

Parameters

stream	
--------	--

Description copied from interface: **streamSubscriberClose**

## 5.71. streamSubscriberStart(ISubscriberStream)

public void streamSubscriberStart(org.red5.server.api.stream.ISubscriberStream stream);
---

**Specified by:** Method streamSubscriberStart in interface IStreamAwareScopeHandler

Notified when a subscriber starts.

Parameters

stream	
--------	--

Description copied from interface: **streamSubscriberStart**

## 5.72. unregisterSharedObjectSecurity(ISHaredObjectSecurity)

public void unregisterSharedObjectSecurity(org.red5.server.api.so.ISharedObjectSecurity handler);
---

**Specified by:** Method unregisterSharedObjectSecurity in interface ISharedObjectSecurityService

Remove handler that protects shared objects.

## 5.73. unregisterStreamPlaybackSecurity(IStreamPlaybackSecurity)

public void unregisterStreamPlaybackSecurity(org.red5.server.api.stream.IStreamPlaybackSecurity hand
--

**Specified by:** Method unregisterStreamPlaybackSecurity in interface IStreamSecurityService

Remove handler that protects stream playback.

## 5.74. unregisterStreamPublishSecurity(IStreamPublishSecurity)

```
public void unregisterStreamPublishSecurity(org.red5.server.api.stream.IStreamPublishSecurity handle
```

**Specified by:** Method unregisterStreamPublishSecurity in interface IStreamSecurityService

Remove handler that protects stream publishing.

# 6. Class StatefulScopeWrappingAdapter

StatefulScopeWrappingAdapter class wraps stateful IScope functionality. That is, it has attributes that you can work with, subscopes, associated resources and connections.

## 6.1. Synopsis

```
public class StatefulScopeWrappingAdapter extends, org.?red5.?server.?adapter.?AbstractScopeAdapter
    implements, org.?red5.?server.?api.?IScopeAware, org.?red5.?server.?api.?IAttributeStore {
    // Protected Fields

    protected org.red5.server.api.IScope scope ;

    // Public Constructors

    public StatefulScopeWrappingAdapter();

    // Public Methods

    public boolean createChildScope(String name);

    public Object getAttribute(String name);

    public Object getAttribute(String name,
        Object defaultValue);

    public java.util.Set<java.lang.String> getAttributeNames();

    public java.util.Map<java.lang.String, java.lang.Object> getAttributes();

    public org.red5.server.api.IScope getChildScope(String name);

    public java.util.Iterator<java.lang.String> getChildScopeNames();

    public java.util.Set<org.red5.server.api.IClient> getClients();

    public java.util.Iterator<org.red5.server.api.IConnection> getConnectionsIter();

    public org.red5.server.api.IContext getContext();

    public int getDepth();

    public String getName();

    public org.red5.server.api.IScope getParent();
```

```

public String getPath();

public org.springframework.core.io.Resource getResource(String path);

public org.springframework.core.io.Resource[] getResources(String pattern)
throws IOException;

public org.red5.server.api.IScope getScope();

public boolean hasAttribute(String name);

public boolean hasChildScope(String name);

public boolean hasParent();

public java.util.Set<org.red5.server.api.IConnection> lookupConnections(org.red5.server.api.IClien

public boolean removeAttribute(String name);

public void removeAttributes();

public boolean setAttribute(String name,
                           Object value);

public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);

public void setAttributes(org.red5.server.api.IAttributeStore values);

public void setScope(org.red5.server.api.IScope scope);

}

}

```

**Direct known subclasses:** org.?red5.?server.?adapter.?

MultiThreadedApplicationAdapter

**Methods inherited from org.red5.server.adapter.AbstractScopeAdapter:**

addChildScope , connect , disconnect , handleEvent , join , leave , removeChildScope  
, serviceCall , setCanCallService , setCanConnect , setCanStart , setJoin , start ,  
stop

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 6.2. scope

```
protected org.red5.server.api.IScope scope ;
```

Wrapped scope

## 6.3. createChildScope(String)

```
public boolean createChildScope(String name);
```

Creates child scope

#### Parameters

<i>name</i>	Child scope name
<i>return</i>	true on success, false otherwise

## 6.4. getAttribute(String)

```
public Object getAttribute(String name);
```

**Specified by:** Method getAttribute in interface IAttributeStore

Return the value for a given attribute.

## 6.5. getAttribute(String, Object)

```
public Object getAttribute(String name,
                           Object defaultValue);
```

**Specified by:** Method getAttribute in interface IAttributeStore

Return the value for a given attribute and set it if it doesn't exist.

This is a utility function that internally performs the following code:

```
if (!hasAttribute(name)) setAttribute(name, defaultValue); return
getAttribute(name);
```

## 6.6. getAttributeNames()

```
public java.util.Set<java.lang.String> getAttributeNames();
```

**Specified by:** Method getAttributeNames in interface IAttributeStore

Get the attribute names. The resulting set will be read-only.

## 6.7. getAttributes()

```
public java.util.Map<java.lang.String, java.lang.Object> getAttributes();
```

**Specified by:** Method getAttributes in interface IAttributeStore

Wrapper for Scope#getAttributes

#### Parameters

<i>return</i>	Scope attributes map
---------------	----------------------

## 6.8. getChildScope(String)

```
public org.red5.server.api.IScope getChildScope(String name);
```

Return child scope

**Parameters**

<i>name</i>	Child scope name
<i>return</i>	Child scope with given name

**6.9. getChildScopeNames()**

```
public java.util.Iterator<java.lang.String> getChildScopeNames();
```

Iterator for child scope names

**Parameters**

<i>return</i>	Iterator for child scope names
---------------	--------------------------------

**6.10. getClients()**

```
public java.util.Set<org.red5.server.api.IClient> getClients();
```

Getter for set of clients

**Parameters**

<i>return</i>	Set of clients
---------------	----------------

**6.11. getConnectionsIter()**

```
public java.util.Iterator<org.red5.server.api.IConnection> getConnectionsIter();
```

Return for connections iterator

**Parameters**

<i>return</i>	Connections
---------------	-------------

**6.12. getContext()**

```
public org.red5.server.api.IContext getContext();
```

Getter for context

**Parameters**

<i>return</i>	Value for context
---------------	-------------------

**6.13. getDepth()**

```
public int getDepth();
```

Getter for depth

**Parameters**

<i>return</i>	Value for depth
---------------	-----------------

## 6.14. getName()

```
public String getName();
```

Getter for name

### Parameters

<i>return</i>	Value for name
---------------	----------------

## 6.15. getParent()

```
public org.red5.server.api.IScope getParent();
```

Return parent scope

### Parameters

<i>return</i>	Parent scope
---------------	--------------

## 6.16. getPath()

```
public String getPath();
```

Getter for stateful scope path

### Parameters

<i>return</i>	Value for path
---------------	----------------

## 6.17. getResource(String)

```
public org.springframework.core.io.Resource getResource(String path);
```

Return resource by name

### Parameters

path	Resource name
<i>return</i>	Resource with given name

## 6.18. getResources(String)

```
public org.springframework.core.io.Resource[] getResources(String pattern)
throws IOException;
```

Returns array of resources (as Spring core Resource class instances)

### Parameters

pattern	Resource pattern
return	Returns array of resources

IOException  
I/O exception

## 6.19. getScope()

```
public org.red5.server.api.IScope getScope();
```

Getter for wrapped scope

Parameters	
return	Wrapped scope

## 6.20. hasAttribute(String)

```
public boolean hasAttribute(String name);
```

**Specified by:** Method hasAttribute in interface IAttributeStore

Check the object has an attribute.

## 6.21. hasChildScope(String)

```
public boolean hasChildScope(String name);
```

Whether this scope has a child scope with given name

Parameters	
name	Child scope name
return	true if it does have it, false otherwise

## 6.22. hasParent()

```
public boolean hasParent();
```

If this scope has a parent

Parameters	
return	true if this scope has a parent scope, false otherwise

## 6.23. removeAttribute(String)

```
public boolean removeAttribute(String name);
```

**Specified by:** Method removeAttribute in interface IAttributeStore

Remove an attribute.

## 6.24. removeAttributes()

```
public void removeAttributes();
```

**Specified by:** Method removeAttributes in interface IAttributeStore

Remove all attributes.

## 6.25. setAttribute(String, Object)

```
public boolean setAttribute(String name,  
                           Object value);
```

**Specified by:** Method setAttribute in interface IAttributeStore

Set an attribute on this object.

## 6.26. setAttributes(IAttributeStore)

```
public void setAttributes(org.red5.server.api.IAttributeStore values);
```

**Specified by:** Method setAttributes in interface IAttributeStore

Set multiple attributes on this object.

## 6.27. setAttributes(Map<String, Object>)

```
public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);
```

**Specified by:** Method setAttributes in interface IAttributeStore

Set multiple attributes on this object.

## 6.28. setScope(IScope)

```
public void setScope(org.red5.server.api.IScope scope);
```

**Specified by:** Method setScope in interface IScopeAware

Set the scope the object is located in.

# 1. Interface AttributeStoreMBean

Base interface for all API objects with attributes

## 1.1. Synopsis

```
public interface AttributeStoreMBean {  
    // Public Methods  
  
    public Object getAttribute(String name);  
  
    public Object getAttribute(String name,  
                               Object defaultValue);  
  
    public java.util.Set<java.lang.String> getAttributeNames();  
  
    public java.util.Map<java.lang.String, java.lang.Object> getAttributes();  
  
    public boolean hasAttribute(String name);  
  
    public boolean removeAttribute(String name);  
  
    public void removeAttributes();  
  
    public boolean setAttribute(String name,  
                               Object value);  
  
}
```

# 2. Interface ClientMBean

The client object represents a single client. One client may have multiple connections to different scopes on the same host. In some ways the client object is like a HTTP session. You can create IClient objects with `newClient(java.lang.Object[])` NOTE: I removed session, since client serves the same purpose as a client with attributes

## 2.1. Synopsis

```
public interface ClientMBean {  
    // Public Methods  
  
    public void disconnect();  
  
    public java.util.Set<org.red5.server.api.IConnection> getConnections();  
  
    public long getCreationTime();  
  
    public String getId();  
  
    public java.util.Collection<org.red5.server.api.IScope> getScopes();  
  
}
```

## 3. Interface ClientRegistryMBean

An MBean interface for the client registry.

### 3.1. Synopsis

```
public interface ClientRegistryMBean {
// Public Methods

    public IClient getClient(String id)
        throws ClientNotFoundException;

    public java.util.List<org.red5.server.api.IClient> getClientList();

    public boolean hasClient(String id);

    public IClient lookupClient(String id)
        throws ClientNotFoundException;

    public IClient newClient(Object[] params);

    public String nextId();

}
```

## 4. Interface ConnectionMBean

The connection object. Each connection has an associated client and scope. Connections may be persistent, polling, or transient. The aim of this interface is to provide basic connection methods shared between different types of connections Future subclasses: RTMPConnection, RemotingConnection, AJAXConnection, HttpConnection, etc

### 4.1. Synopsis

```
public interface ConnectionMBean {
// Public Methods

    public void close();

    public boolean connect(IScope scope);

    public boolean connect(IScope scope,
        Object[] params);

    public java.util.Iterator<org.red5.server.api.IBasicScope> getBasicScopes();

    public IClient getClient();

    public java.util.Map<java.lang.String, java.lang.Object> getConnectParams();

    public long getDroppedMessages();
```

```

public String getHost();

public int getLastPingTime();

public String getPath();

public long getPendingMessages();

public long getReadBytes();

public long getReadMessages();

public String getRemoteAddress();

public java.util.List<java.lang.String> getRemoteAddresses();

public int getRemotePort();

public IScope getScope();

public String getSessionId();

public String getType();

public long getWrittenBytes();

public long getWrittenMessages();

public void initialize(IClient client);

public boolean isConnected();

public void ping();

}

```

## 5. Interface IApplicationContext

Interface that wraps the application server context.

### 5.1. Synopsis

```

public interface IApplicationContext {
// Public Methods

    public void stop();

}

```

### 5.2. stop()

```
public void stop();
```

Stop the web application.

## 6. Interface IApplicationLoader

Interface for servers that can load new applications.

### 6.1. Synopsis

```
public interface IApplicationLoader {
    // Public Methods

    public org.springframework.context.ApplicationContext getRootContext();

    public void loadApplication(String contextPath,
                               String virtualHosts,
                               String directory)
        throws Exception;

}
```

### 6.2. getRootContext()

```
public org.springframework.context.ApplicationContext getRootContext();
```

Return the root org.springframework.context.ApplicationContext.

#### Parameters

*return*

### 6.3. loadApplication(String, String, String)

```
public void loadApplication(String contextPath,
                           String virtualHosts,
                           String directory)
        throws Exception;
```

Load a new application for the given context path from a directory.

#### Parameters

contextPath

virtualHosts

directory

Exception

java.lang.Exception

## 7. Interface IAttributeStore

Base interface for all API objects with attributes

## 7.1. Synopsis

```

public interface IAttributeStore extends, org.?red5.?server.?api.?AttributeStoreMBean {
    // Public Methods

    public Object getAttribute(String name);

    public Object getAttribute(String name,
                               Object defaultValue);

    public java.util.Set<java.lang.String> getAttributeNames();

    public java.util.Map<java.lang.String, java.lang.Object> getAttributes();

    public boolean hasAttribute(String name);

    public boolean removeAttribute(String name);

    public void removeAttributes();

    public boolean setAttribute(String name,
                               Object value);

    public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);

    public void setAttributes(IAttributeStore values);
}

```

## 7.2. getAttribute(String)

```
public Object getAttribute(String name);
```

**Specified by:** Method getAttribute in interface AttributeStoreMBean

Return the value for a given attribute.

### Parameters

name	the name of the attribute to get
return	the attribute value or null if the attribute doesn't exist

## 7.3. getAttribute(String, Object)

```
public Object getAttribute(String name,
                           Object defaultValue);
```

**Specified by:** Method getAttribute in interface AttributeStoreMBean

Return the value for a given attribute and set it if it doesn't exist.

This is a utility function that internally performs the following code:

```
if (!hasAttribute(name)) setAttribute(name, defaultValue); return
getAttribute(name);
```

**Parameters**

<i>name</i>	the name of the attribute to get
<i>defaultValue</i>	the value of the attribute to set if the attribute doesn't exist
<i>return</i>	the attribute value

## 7.4. **getAttributeNames()**

```
public java.util.Set<java.lang.String> getAttributeNames();
```

**Specified by:** Method `getAttributeNames` in interface `AttributeStoreMBean`

Get the attribute names. The resulting set will be read-only.

**Parameters**

<i>return</i>	set containing all attribute names
---------------	------------------------------------

## 7.5. **getAttributes()**

```
public java.util.Map<java.lang.String, java.lang.Object> getAttributes();
```

**Specified by:** Method `getAttributes` in interface `AttributeStoreMBean`

Get the attributes. The resulting map will be read-only.

**Parameters**

<i>return</i>	map containing all attributes
---------------	-------------------------------

## 7.6. **hasAttribute(String)**

```
public boolean hasAttribute(String name);
```

**Specified by:** Method `hasAttribute` in interface `AttributeStoreMBean`

Check the object has an attribute.

**Parameters**

<i>name</i>	the name of the attribute to check
<i>return</i>	true if the attribute exists otherwise false

## 7.7. **removeAttribute(String)**

```
public boolean removeAttribute(String name);
```

**Specified by:** Method `removeAttribute` in interface `AttributeStoreMBean`

Remove an attribute.

#### Parameters

<i>name</i>	the name of the attribute to remove
<i>return</i>	true if the attribute was found and removed otherwise false

## 7.8. removeAttributes()

```
public void removeAttributes();
```

**Specified by:** Method removeAttributes in interface AttributeStoreMBean

Remove all attributes.

## 7.9. setAttribute(String, Object)

```
public boolean setAttribute(String name,
                           Object value);
```

**Specified by:** Method setAttribute in interface AttributeStoreMBean

Set an attribute on this object.

#### Parameters

<i>name</i>	the name of the attribute to change
<i>value</i>	the new value of the attribute
<i>return</i>	true if the attribute value changed otherwise false

## 7.10. setAttributes(IAttributeStore)

```
public void setAttributes(IAttributeStore values);
```

Set multiple attributes on this object.

#### Parameters

<i>values</i>	the attributes to set
---------------	-----------------------

## 7.11. setAttributes(Map<String, Object>)

```
public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);
```

Set multiple attributes on this object.

#### Parameters

<i>values</i>	the attributes to set
---------------	-----------------------

# 8. Interface IBWControllable

Mark an object that can be bandwidth controlled.

A bw-controlled object has the bandwidth config property and a link to the parent controllable object.

The parent controllable object acts as the bandwidth provider for this object, thus generates a tree structure, in which the `null` parent means the host. The next depth level is the `IClient`. The following is `IStreamCapableConnection`. The deepest level is `IClientStream`. That is, bandwidth can be separately configured for client stream or connection, or client or the whole application.

The summary of children's bandwidth can't exceed the parent's bandwidth even though the children's bandwidth could be configured larger than the parent's bandwidth.

## 8.1. Synopsis

```
public interface IBWControllable {
    // Public Methods

    public IBandwidthConfigure getBandwidthConfigure();

    public IBWControllable getParentBWControllable();

    public void setBandwidthConfigure(IBandwidthConfigure config);

}
```

## 8.2. getBandwidthConfigure()

<code>public IBandwidthConfigure getBandwidthConfigure();</code>
--

Return bandwidth configuration object. Bandwidth configuration allows you to set bandwidth size for audio, video and total amount.

### Parameters

<i>return</i>	Bandwidth configuration object
---------------	--------------------------------

## 8.3. getParentBWControllable()

<code>public IBWControllable getParentBWControllable();</code>
--

Return parent IFlowControllable object

### Parameters

<i>return</i>	parent Parent flow controllable
---------------	---------------------------------

## 8.4. setBandwidthConfigure(IBandwidthConfigure)

<code>public void setBandwidthConfigure(IBandwidthConfigure config);</code>
---

Setter for bandwidth configuration

#### Parameters

config	Value to set for bandwidth configuration
--------	--

## 9. Interface IBandwidthConfigure

Interface for setting/getting bandwidth configure. Two properties are provided for bandwidth configuration. The property "channelBandwidth" is used to configure the bandwidth of each channel. The property "channelInitialBurst" is used to configure the initial bytes that can be sent to client in each channel.

### 9.1. Synopsis

```
public interface IBandwidthConfigure extends, java.lang.Cloneable {
// Public Static Fields

    public static final int AUDIO_CHANNEL = 0;

    public static final int DATA_CHANNEL = 2;

    public static final int MAX_CHANNEL_CONFIG_COUNT = 4;

    public static final int OVERALL_CHANNEL = 3;

    public static final int VIDEO_CHANNEL = 1;

// Public Methods

    public long[] getChannelBandwidth();

    public long[] getChannelInitialBurst();

}
```

### 9.2. getChannelBandwidth()

```
public long[] getChannelBandwidth();
```

Return the bandwidth configure for 3 channels: audio, video, data and the overall bandwidth. The unit is bit per second. A value of -1 means "don't care" so that there's no limit on bandwidth for that channel. The last element is the overall bandwidth. If it's not -1, the value of the first three elements will be ignored.

#### Parameters

return	The 4-element array of bandwidth configure.
--------	---

### 9.3. getChannelInitialBurst()

```
public long[] getChannelInitialBurst();
```

Return the byte count of initial burst value for 3 channels: audio, video, data and the overall bandwidth. If the value is -1, the default will be used per the implementation of bandwidth controller.

#### Parameters

<i>return</i>	The 4-element array of byte count of initial burst value.
---------------	---

## 10. Interface IBasicScope

Base interface for all scope objects, including SharedObjects.

### 10.1. Synopsis

```
public interface IBasicScope extends, org.?red5.?server.?api.?ICoreObject, org.?red5.?server.?api.?eve
// Public Methods

    public int getDepth();

    public String getName();

    public IScope getParent();

    public String getPath();

    public String getType();

    public boolean hasParent();

}
```

### 10.2. getDepth()

```
public int getDepth();
```

Get the scopes depth, how far down the scope tree is it. The lowest depth is 0x00, the depth of Global scope. Application scope depth is 0x01. Room depth is 0x02, 0x03 and so forth.

#### Parameters

<i>return</i>	the depth
---------------	-----------

### 10.3. getName()

```
public String getName();
```

**Specified by:** Method getName in interface IPersistable

Get the name of this scope. Eg. someroom.

#### Parameters

<i>return</i>	the name
---------------	----------

## 10.4. getParent()

```
public IScope getParent();
```

Get this scopes parent.

### Parameters

<i>return</i>	parent scope, or <code>null</code> if this scope doesn't have a parent
---------------	--

## 10.5. getPath()

```
public String getPath();
```

**Specified by:** Method getPath in interface IPersistable

Get the full absolute path. Eg. `host/myapp/someroom`.

### Parameters

<i>return</i>	Absolute scope path
---------------	---------------------

## 10.6. getType()

```
public String getType();
```

**Specified by:** Method getType in interface IPersistable

Get the type of the scope.

### Parameters

<i>return</i>	Type of scope
---------------	---------------

## 10.7. hasParent()

```
public boolean hasParent();
```

Does this scope have a parent? You can think of scopes as of tree items where scope may have a parent and children (child).

### Parameters

<i>return</i>	<code>true</code> if this scope has a parent, otherwise <code>false</code>
---------------	--

# 11. Interface ICastingAttributeStore

Attribute storage with automatic object casting support.

## 11.1. Synopsis

```
public interface ICastingAttributeStore extends, org.?red5.?server.?api.?IAttributeStore {
```

```
// Public Methods

    public Boolean getBoolAttribute(String name);

    public Byte getByteAttribute(String name);

    public Double getDoubleAttribute(String name);

    public Integer getIntAttribute(String name);

    public java.util.List getListAttribute(String name);

    public Long getLongAttribute(String name);

    public java.util.Map getMapAttribute(String name);

    public java.util.Set getSetAttribute(String name);

    public Short getShortAttribute(String name);

    public String getStringAttribute(String name);

}
```

## 11.2. getBoolAttribute(String)

```
public Boolean getBoolAttribute(String name);
```

Get Boolean attribute by name

### Parameters

name	Attribute name
<i>return</i>	Attribute

## 11.3. getByteAttribute(String)

```
public Byte getByteAttribute(String name);
```

Get Byte attribute by name

### Parameters

name	Attribute name
<i>return</i>	Attribute

## 11.4. getDoubleAttribute(String)

```
public Double getDoubleAttribute(String name);
```

Get Double attribute by name

**Parameters**

<i>name</i>	Attribute name
<i>return</i>	Attribute

**11.5. getIntAttribute(String)**

```
public Integer getIntAttribute(String name);
```

Get Integer attribute by name

**Parameters**

<i>name</i>	Attribute name
<i>return</i>	Attribute

**11.6. getListAttribute(String)**

```
public java.util.List getListAttribute(String name);
```

Get List attribute by name

**Parameters**

<i>name</i>	Attribute name
<i>return</i>	Attribute

**11.7. getLongAttribute(String)**

```
public Long getLongAttribute(String name);
```

Get boolean attribute by name

**Parameters**

<i>name</i>	Attribute name
<i>return</i>	Attribute

**11.8. getMapAttribute(String)**

```
public java.util.Map getMapAttribute(String name);
```

Get Long attribute by name

**Parameters**

<i>name</i>	Attribute name
<i>return</i>	Attribute

## 11.9. **getSetAttribute(String)**

```
public java.util.Set getSetAttribute(String name);
```

Get Set attribute by name

### Parameters

<i>name</i>	Attribute name
<i>return</i>	Attribute

## 11.10. **getShortAttribute(String)**

```
public Short getShortAttribute(String name);
```

Get Short attribute by name

### Parameters

<i>name</i>	Attribute name
<i>return</i>	Attribute

## 11.11. **getStringAttribute(String)**

```
public String getStringAttribute(String name);
```

Get String attribute by name

### Parameters

<i>name</i>	Attribute name
<i>return</i>	Attribute

# 12. Interface IClient

The client object represents a single client. One client may have multiple connections to different scopes on the same host. In some ways the client object is like a HTTP session. You can create IClient objects with `newClient(java.lang.Object[])` NOTE: I removed session, since client serves the same purpose as a client with attributes

## 12.1. Synopsis

```
public interface IClient extends, org.?red5.?server.?api.?ClientMBean, org.?red5.?server.?api.?IAttri
// Public Static Fields

    public static final String ID = "red5.client";

// Public Methods

    public void disconnect();

    public java.util.Set<org.red5.server.api.IConnection> getConnections();
```

```

public java.util.Set<org.red5.server.api.IConnection> getConnections(IScope scope);

public long getCreationTime();

public String getId();

public java.util.Collection<java.lang.String> getPermissions(IConnection conn);

public java.util.Collection<org.red5.server.api.IScope> getScopes();

public boolean hasPermission(IConnection conn,
                           String permissionName);

public void setPermissions(IConnection conn,
                           java.util.Collection<java.lang.String> permissions);

}

```

## 12.2. ID

```
public static final String ID = "red5.client";
```

The key used to store the client object in a http session.

## 12.3. disconnect()

```
public void disconnect();
```

**Specified by:** Method disconnect in interface ClientMBean

Closes all the connections.

## 12.4. getConnections()

```
public java.util.Set<org.red5.server.api.IConnection> getConnections();
```

**Specified by:** Method getConnections in interface ClientMBean

Get a set of connections.

### Parameters

return	Set of connections
--------	--------------------

## 12.5. getConnections(IScope)

```
public java.util.Set<org.red5.server.api.IConnection> getConnections(IScope scope);
```

Get a set of connections of a given scope.

### Parameters

scope	scope to get connections for
<i>return</i>	Set of connections to the passed scope

## 12.6. getCreationTime()

```
public long getCreationTime();
```

**Specified by:** Method getCreationTime in interface ClientMBean

Get the creation time for this client object.

### Parameters

<i>return</i>	Creation time in milliseconds
---------------	-------------------------------

## 12.7. getId()

```
public String getId();
```

**Specified by:** Method getId in interface ClientMBean

Get the unique ID for this client. This will be generated by the server if not passed upon connection from client-side Flex/Flash app. To assign a custom ID to the client use `params` object of `appConnect(org.red5.server.api.IConnection, java.lang.Object[])` method, that contains 2nd all the rest values you pass to `NetConnection.connect` method. Example:  
**At client side:** `NetConnection.connect( "http://localhost/killerapp/", "user123" );`  
**then at server side:** `public boolean appConnect( IConnection connection, Object[] params ){ try { connection.getClient().setStreamId( params[0] ); } catch(Exception e){ log.error("{}", e); } }`

### Parameters

<i>return</i>	client id
---------------	-----------

## 12.8. getPermissions(IConnection)

```
public java.util.Collection<java.lang.String> getPermissions(IConnection conn);
```

Return the permissions in a given context.

### Parameters

conn	Connection specifying the context to get the permissions for.
<i>return</i>	Permission names.

## 12.9. getScopes()

```
public java.util.Collection<org.red5.server.api.IScope> getScopes();
```

**Specified by:** Method getScopes in interface ClientMBean

Get a set of scopes the client is connected to.

#### Parameters

<i>return</i>	Set of scopes
---------------	---------------

## 12.10. hasPermission(IConnection, String)

```
public boolean hasPermission(IConnection conn,
                           String permissionName);
```

Check if the client has a permission in the given context.

#### Parameters

conn	Connection specifying the context to check the permissions for.
permissionName	Name of the permission to check.
<i>return</i>	<code>true</code> if the client has the permission, otherwise <code>false</code>

## 12.11. setPermissions(IConnection, Collection<String>)

```
public void setPermissions(IConnection conn,
                           java.util.Collection<java.lang.String> permissions);
```

Set the permissions for this client in a given context.

#### Parameters

conn	Connection specifying the context to set the permissions for.
permissions	Permissions the client has in this context or <code>null</code> for no permissions.

## 13. Interface IClientRegistry

Provides a registry of client objects. You can lookup a client by its client id / session id using `lookupClient` method. This interface implementations also create new client objects from given params, usually passed from client-side Flex/Flash application upon initial connection.

### 13.1. Synopsis

```
public interface IClientRegistry {
    // Public Methods

    public boolean hasClient(String id);

    public IClient lookupClient(String id)
        throws ClientNotFoundException;

    public IClient newClient(Object[] params)
        throws ClientNotFoundException, ClientRejectedException;
```

}

## 13.2. hasClient(String)

```
public boolean hasClient(String id);
```

Check if a client with a given id exists.

### Parameters

<i>id</i>	the id of the client to check for
<i>return</i>	true if the client exists, false otherwise

## 13.3. lookupClient(String)

```
public IClient lookupClient(String id)
    throws ClientNotFoundException;
```

Return an existing client from a client id.

### Parameters

<i>id</i>	the id of the client to return
<i>return</i>	the client object

`ClientNotFoundException`  
no client with the passed id exists

## 13.4. newClient(Object[])

```
public IClient newClient(Object[] params)
    throws ClientNotFoundException, ClientRejectedException;
```

Create a new client client object from connection params.

### Parameters

<i>params</i>	the parameters the client passed during connection
<i>return</i>	the new client

`ClientNotFoundException`  
no client could be created from the passed parameters

`ClientRejectedException`  
the client is not allowed to connect

## 14. Interface IConnection

The connection object. Each connection has an associated client and scope. Connections may be persistent, polling, or transient. The aim of this interface is to provide basic

connection methods shared between different types of connections Future subclasses:  
RTMPConnection, RemotingConnection, AJAXConnection, HttpConnection, etc

## 14.1. Synopsis

```
public interface IConnection extends, org.?red5.?server.?api.?ConnectionMBean, org.?red5.?server.?api?  
// Public Static Fields  
  
    public static final String PERSISTENT = "persistent";  
  
    public static final String POLLING = "polling";  
  
    public static final String TRANSIENT = "transient";  
  
// Public Methods  
  
    public void close();  
  
    public boolean connect(IScope scope);  
  
    public boolean connect(IScope scope,  
                          Object[] params);  
  
    public java.util.Iterator<org.red5.server.api.IBasicScope> getBasicScopes();  
  
    public IClient getClient();  
  
    public long getClientBytesRead();  
  
    public java.util.Map<java.lang.String, java.lang.Object> getConnectParams();  
  
    public long getDroppedMessages();  
  
    public org.red5.server.api.IConnection.Encoding getEncoding();  
  
    public String getHost();  
  
    public int getLastPingTime();  
  
    public String getPath();  
  
    public long getPendingMessages();  
  
    public long getReadBytes();  
  
    public long getReadMessages();  
  
    public String getRemoteAddress();  
  
    public java.util.List<java.lang.String> getRemoteAddresses();  
  
    public int getRemotePort();  
  
    public IScope getScope();  
  
    public String getSessionId();
```

```

public String getType();

public long getWrittenBytes();

public long getWrittenMessages();

public void initialize(IClient client);

public boolean isConnected();

public void ping();

}

```

## 14.2. PERSISTENT

```
public static final String PERSISTENT = "persistent";
```

Persistent connection type, eg RTMP.

## 14.3. POLLING

```
public static final String POLLING = "polling";
```

Polling connection type, eg RTMPT.

## 14.4. TRANSIENT

```
public static final String TRANSIENT = "transient";
```

Transient connection type, eg Remoting, HTTP, etc.

## 14.5. close()

```
public void close();
```

**Specified by:** Method close in interface ConnectionMBean

Close this connection. This will disconnect the client from the associated scope.

## 14.6. connect(IScope)

```
public boolean connect(IScope scope);
```

**Specified by:** Method connect in interface ConnectionMBean

Try to connect to the scope.

### Parameters

scope	Scope object
-------	--------------

*return*

true on success, false otherwise

## 14.7. connect(IScope, Object[])

```
public boolean connect(IScope scope,
                      Object[] params);
```

**Specified by:** Method connect in interface ConnectionMBean

Try to connect to the scope with a list of connection parameters.

### Parameters

params	Connections parameters
--------	------------------------

scope	Scope object
-------	--------------

<i>return</i>	true on success, false otherwise
---------------	----------------------------------

## 14.8. getBasicScopes()

```
public java.util.Iterator<org.red5.server.api.IBasicScope> getBasicScopes();
```

**Specified by:** Method getBasicScopes in interface ConnectionMBean

Get the basic scopes this connection has subscribed. This list will contain the shared objects and broadcast streams the connection connected to.

### Parameters

<i>return</i>	List of basic scopes
---------------	----------------------

## 14.9. getClient()

```
public IClient getClient();
```

**Specified by:** Method getClient in interface ConnectionMBean

Get the client object associated with this connection.

### Parameters

<i>return</i>	Client object
---------------	---------------

## 14.10. getClientBytesRead()

```
public long getClientBytesRead();
```

Return number of written bytes the client reports to have received. This is the last value of the BytesRead message received from a client.

### Parameters

<i>return</i>	number of written bytes received by the client
---------------	--

**See Also**

[org.red5.server.net.rtmp.event.BytesRead](#)

## 14.11. getConnectParams()

```
public java.util.Map<java.lang.String, java.lang.Object> getConnectParams();
```

**Specified by:** Method `getConnectParams` in interface `ConnectionMBean`

Return the parameters that were given in the call to "connect".

**Parameters**

<i>return</i>	Connection parameters passed from client-side (Flex/Flash application)
---------------	--

## 14.12. getDroppedMessages()

```
public long getDroppedMessages();
```

**Specified by:** Method `getDroppedMessages` in interface `ConnectionMBean`

Total number of messages that have been dropped.

**Parameters**

<i>return</i>	Number of dropped messages
---------------	----------------------------

## 14.13. getEncoding()

```
public org.red5.server.api.IConnection.Encoding getEncoding();
```

Get the object encoding (AMF version) for this connection.

**Parameters**

<i>return</i>	the used encoding.
---------------	--------------------

## 14.14. getHost()

```
public String getHost();
```

**Specified by:** Method `getHost` in interface `ConnectionMBean`

Get the hostname that the client is connected to. If they are connected to an IP, the IP address will be returned as a String.

**Parameters**

<i>return</i>	String containing the hostname
---------------	--------------------------------

## 14.15. getLastPingTime()

```
public int getLastPingTime();
```

**Specified by:** Method getLastPingTime in interface ConnectionMBean

Return roundtrip time of last ping command.

### Parameters

<i>return</i>	roundtrip time in milliseconds
---------------	--------------------------------

## 14.16. getPath()

```
public String getPath();
```

**Specified by:** Method getPath in interface ConnectionMBean

Get the path for this connection. This is not updated if you switch scope.

### Parameters

<i>return</i>	path Connection path
---------------	----------------------

## 14.17. getPendingMessages()

```
public long getPendingMessages();
```

**Specified by:** Method getPendingMessages in interface ConnectionMBean

Total number of messages that are pending to be sent to the connection.

### Parameters

<i>return</i>	Number of pending messages
---------------	----------------------------

## 14.18. getReadBytes()

```
public long getReadBytes();
```

**Specified by:** Method getReadBytes in interface ConnectionMBean

Total number of bytes read from the connection.

### Parameters

<i>return</i>	Number of read bytes
---------------	----------------------

## 14.19. getReadMessages()

```
public long getReadMessages();
```

**Specified by:** Method getReadMessages in interface ConnectionMBean

Total number of messages read from the connection.

#### Parameters

<i>return</i>	Number of read messages
---------------	-------------------------

## 14.20. getRemoteAddress()

```
public String getRemoteAddress();
```

**Specified by:** Method getRemoteAddress in interface ConnectionMBean

Get the IP address the client is connected from.

#### Parameters

<i>return</i>	The IP address of the client
---------------	------------------------------

## 14.21. getRemoteAddresses()

```
public java.util.List<java.lang.String> getRemoteAddresses();
```

**Specified by:** Method getRemoteAddresses in interface ConnectionMBean

Get the IP addresses the client is connected from. If a client is connected through RTMPT and uses a proxy to connect, this will contain all hosts the client used to connect to the server.

#### Parameters

<i>return</i>	The IP addresses of the client
---------------	--------------------------------

## 14.22. getRemotePort()

```
public int getRemotePort();
```

**Specified by:** Method getRemotePort in interface ConnectionMBean

Get the port the client is connected from.

#### Parameters

<i>return</i>	The port of the client
---------------	------------------------

## 14.23. getScope()

```
public IScope getScope();
```

**Specified by:** Method getScope in interface ConnectionMBean

Get the scope this is connected to.

#### Parameters

<i>return</i>	The connected scope
---------------	---------------------

## 14.24. getSessionId()

```
public String getSessionId();
```

**Specified by:** Method getSessionId in interface ConnectionMBean

Get the session id, this may be null.

### Parameters

<i>return</i>	Session id
---------------	------------

## 14.25. getType()

```
public String getType();
```

**Specified by:** Method getType in interface ConnectionMBean

Get the connection type.

### Parameters

<i>return</i>	string containing one of connection types
---------------	---

## 14.26. getWrittenBytes()

```
public long getWrittenBytes();
```

**Specified by:** Method getWrittenBytes in interface ConnectionMBean

Total number of bytes written to the connection.

### Parameters

<i>return</i>	Number of written bytes
---------------	-------------------------

## 14.27. getWrittenMessages()

```
public long getWrittenMessages();
```

**Specified by:** Method getWrittenMessages in interface ConnectionMBean

Total number of messages written to the connection.

### Parameters

<i>return</i>	Number of written messages
---------------	----------------------------

## 14.28. initialize(IClient)

```
public void initialize(IClient client);
```

**Specified by:** Method initialize in interface ConnectionMBean

Initialize the connection.

## Parameters

client	Client object associated with connection
--------	--

**14.29. isConnected()**

```
public boolean isConnected();
```

**Specified by:** Method isConnected in interface ConnectionMBeanIs the client connected to the scope. Result depends on connection type, `true` for persistent and polling connections, `false` for transient.

## Parameters

return	<code>true</code> if the connection is persistent or polling, otherwise <code>false</code>
--------	--

**14.30. ping()**

```
public void ping();
```

**Specified by:** Method ping in interface ConnectionMBean

Start measuring the roundtrip time for a packet on the connection.

**15. Class IConnection.Encoding**

AMF version types, either AMF0 or AMF3.

**15.1. Synopsis**

```
public static final class IConnection.Encoding extends java.lang.Enum {
    // Public Static Fields

    public static final org.red5.server.api.IConnection.Encoding AMF0 ;
    public static final org.red5.server.api.IConnection.Encoding AMF3 ;

    // Public Static Methods

    public static org.red5.server.api.IConnection.Encoding valueOf(String name);
    public static org.red5.server.api.IConnection.Encoding[] values();
}
```

**Methods inherited from java.lang.Enum:** `clone`, `compareTo`, `equals`, `finalize`, `getDeclaringClass`, `hashCode`, `name`, `ordinal`, `toString`, `valueOf`**Methods inherited from java.lang.Object:** `getClass`, `notify`, `notifyAll`, `wait`

## 16. Interface IConnectionBWConfig

The bandwidth configure for connection that has an extra property "upstreamBandwidth" which is not used by Bandwidth Control Framework in Red5.

### 16.1. Synopsis

```
public interface IConnectionBWConfig extends, org.?red5.?server.?api.?IBandwidthConfigure {
    // Public Methods

    public long getDownstreamBandwidth();

    public long getUpstreamBandwidth();

    public void setUpstreamBandwidth(long bw);

}
```

### 16.2. getDownstreamBandwidth()

<code>public long getDownstreamBandwidth();</code>
--

Getter for downstream bandwidth

#### Parameters

<i>return</i>	Downstream bandwidth, from server to client
---------------	---

### 16.3. getUpstreamBandwidth()

<code>public long getUpstreamBandwidth();</code>
--

Get the upstream bandwidth to be notified to the client. Upstream is the data that is sent from the client to the server.

#### Parameters

<i>return</i>	Upstream (from client to server) bandwidth configuration
---------------	--

### 16.4. setUpstreamBandwidth(long)

<code>public void setUpstreamBandwidth(long bw);</code>
---

Set the upstream bandwidth to be notified to the client. Upstream is the data that is sent from the client to the server.

#### Parameters

<i>bw</i>	Bandwidth
-----------	-----------

## 17. Interface IContext

The current context, this object basically wraps the Spring context or in the case of the .Net version, any similar system.

## 17.1. Synopsis

```
public interface IContext extends org.springframework.core.io.ResourcePatternResolver {
    // Public Static Fields

    public static final String ID = "red5.context";

    // Public Methods

    public org.springframework.context.ApplicationContext getApplicationContext();

    public Object getBean(String beanId);

    public IClientRegistry getClientRegistry();

    public Object getCoreService(String beanId);

    public IGlobalScope getGlobalScope();

    public IMappingStrategy getMappingStrategy();

    public org.red5.server.api.persistence.IPersistenceStore getPersistenceStore();

    public org.red5.server.api.service.IServiceInvoker getServiceInvoker();

    public IScopeHandler lookupScopeHandler(String path);

    public Object lookupService(String serviceName);

    public IScope resolveScope(String path);

    public IScope resolveScope(IScope root,
                               String path);
}
```

## 17.2. getApplicationContext()

```
public org.springframework.context.ApplicationContext getApplicationContext();
```

Getter for application context

### Parameters

<i>return</i>	Application context
---------------	---------------------

## 17.3. getBean(String)

```
public Object getBean(String beanId);
```

Returns bean by ID

**Parameters**

<i>beanId</i>	Bean ID
<i>return</i>	Given bean instance

**17.4. getClientRegistry()**

```
public IClientRegistry getClientRegistry();
```

Get client registry. Client registry is a place where all clients are registered.

**Parameters**

<i>return</i>	Client registry object
---------------	------------------------

**17.5. getCoreService(String)**

```
public Object getCoreService(String beanId);
```

Returns core service by bean id

**Parameters**

<i>beanId</i>	Bean ID
<i>return</i>	Core service

**17.6. getGlobalScope()**

```
public IGlobalScope getGlobalScope();
```

Returns global scope reference

**Parameters**

<i>return</i>	global scope reference
---------------	------------------------

**17.7. getMappingStrategy()**

```
public IMappingStrategy getMappingStrategy();
```

Returns IMappingStrategy object

**Parameters**

<i>return</i>	IMappingStrategy object
---------------	-------------------------

**17.8. getPersistenceStore()**

```
public org.red5.server.api.persistence.IPersistenceStore getPersistenceStore();
```

Returns persistence store object, a storage for persistent objects like persistent SharedObjects.

**Parameters**

<i>return</i>	Persistence store object
---------------	--------------------------

**17.9. getServiceInvoker()**

```
public org.red5.server.api.service.IServiceInvoker getServiceInvoker();
```

Returns service invoker object. Service invokers are objects that make service calls to client side NetConnection objects.

**Parameters**

<i>return</i>	Service invoker object
---------------	------------------------

**17.10. lookupScopeHandler(String)**

```
public IScopeHandler lookupScopeHandler(String path);
```

Returns scope handler (object that handle all actions related to the scope) by path. See `org.red5.server.api.IScopeHandler` for details.

**Parameters**

path	Path of scope handler
<i>return</i>	Scope handler

**17.11. lookupService(String)**

```
public Object lookupService(String serviceName);
```

Returns service by name.

**Parameters**

serviceName	Name of service
<i>return</i>	Service object

**17.12. resolveScope(IScope, String)**

```
public IScope resolveScope(IScope root,
                           String path);
```

Returns scope by path from given root. You can think of IScope as of tree items, used to separate context and resources between users. See `org.red5.server.api.IScope` for more details.

**Parameters**

root	Root to start from
path	Path of scope

<i>return</i>	IScope object
---------------	---------------

## 17.13. resolveScope(String)

```
public IScope resolveScope(String path);
```

Returns scope by path. You can think of IScope as of tree items, used to separate context and resources between users. See [org.red5.server.api.IScope](#) for more details.

### Parameters

path	Path of scope
<i>return</i>	IScope object

## 18. Interface ICoreObject

Base marker interface for all core objects.

### 18.1. Synopsis

```
public interface ICoreObject extends, org.?red5.?server.?api.?ICastingAttributeStore, org.?red5.?serv
```

## 19. Interface IGlobalScope

The global scope that acts as root for all applications in a host.

### 19.1. Synopsis

```
public interface IGlobalScope extends, org.?red5.?server.?api.?IScope {
// Public Methods

    public IServer getServer();

    public void register();

}
```

### 19.2. getServer()

```
public IServer getServer();
```

Return the server this global scope runs in.

### Parameters

<i>return</i>	the server
---------------	------------

## 19.3. register()

```
public void register();
```

Register the global scope in the server and initialize it.

# 20. Interface IMappingStrategy

This interface encapsulates the mapping strategy used by the context.

## 20.1. Synopsis

```
public interface IMappingStrategy {
    // Public Methods

    public String mapResourcePrefix(String contextPath);

    public String mapScopeHandlerName(String contextPath);

    public String mapServiceName(String name);

}
```

## 20.2. mapResourcePrefix(String)

```
public String mapResourcePrefix(String contextPath);
```

Map a context path to a path prefix for resources.

### Parameters

contextPath	context path to map
<i>return</i>	The path prefix for resources with the given name

## 20.3. mapScopeHandlerName(String)

```
public String mapScopeHandlerName(String contextPath);
```

Map a context path to the name of a scope handler.

### Parameters

contextPath	context path to map
<i>return</i>	The name of a scope handler

## 20.4. mapServiceName(String)

```
public String mapServiceName(String name);
```

Map a name to the name of a service.

### Parameters

name	name to map
return	The name of the service with the passed name

## 21. Interface IScope

The scope object. A statefull object shared between a group of clients connected to the same context path. Scopes are arranged in hierarchical way, so its possible for a scope to have a parent and children scopes. If a client connects to a scope then they are also connected to its parent scope. The scope object is used to access resources, shared object, streams, etc. That is, scope are general option for grouping things in application. The following are all names for scopes: application, room, place, lobby.

### 21.1. Synopsis

```
public interface IScope extends, org.?red5.?server.?api.?IBasicScope, org.?springframework.?core.?io...
// Public Static Fields

    public static final String ID = "red5.scope";

    public static final String SEPARATOR = ":";

    public static final String TYPE = "scope";

// Public Methods

    public boolean addChildScope(IBasicScope scope);

    public boolean connect(IConnection conn);

    public boolean connect(IConnection conn,
                          Object[] params);

    public boolean createChildScope(String name);

    public void disconnect(IConnection conn);

    public IBasicScope getBasicScope(String type,
                                    String name);

    public java.util.Iterator<java.lang.String> getBasicScopeNames(String type);

    public java.util.Set<org.red5.server.api.IClient> getClients();

    public java.util.Iterator<org.red5.server.api.IConnection> getConnections();

    public IContext getContext();

    public String getContextPath();

    public IScopeHandler getHandler();

    public IScope getScope(String name);
```

```

public java.util.Iterator<java.lang.String> getScopeNames();

public org.red5.server.api.statistics.IScopeStatistics getStatistics();

public boolean hasChildScope(String name);

public boolean hasChildScope(String type,
                           String name);

public boolean hasHandler();

public java.util.Set<org.red5.server.api.IConnection> lookupConnections(IClient client);

public void removeChildScope(IBasicScope scope);

}

```

## 21.2. ID

```
public static final String ID = "red5.scope";
```

ID constant

## 21.3. SEPARATOR

```
public static final String SEPARATOR = ":";
```

Scope separator

## 21.4. TYPE

```
public static final String TYPE = "scope";
```

Type constant

## 21.5. addChildScope(IBasicScope)

```
public boolean addChildScope(IBasicScope scope);
```

Adds scope as a child scope. Returns `true` on success, `false` if given scope is already a child of current.

### Parameters

scope	Scope given
return	<code>true</code> if child scope was successfully added, <code>false</code> otherwise

## 21.6. connect(IConnection)

```
public boolean connect(IConnection conn);
```

Adds given connection to the scope

**Parameters**

<code>conn</code>	Given connection
<code>return</code>	<code>true</code> on success, <code>false</code> if given connection already belongs to this scope

**21.7. connect(IConnection, Object[])**

```
public boolean connect(IConnection conn,
                      Object[] params);
```

Add given connection to the scope, overloaded for parameters pass case.

**Parameters**

<code>conn</code>	Given connection
<code>params</code>	Parameters passed
<code>return</code>	<code>true</code> on success, <code>false</code> if given connection already belongs to this scope

**21.8. createChildScope(String)**

```
public boolean createChildScope(String name);
```

Creates child scope with name given and returns success value. Returns `true` on success, `false` if given scope already exists among children.

**Parameters**

<code>name</code>	New child scope name
<code>return</code>	<code>true</code> if child scope was successfully creates, <code>false</code> otherwise

**21.9. disconnect(IConnection)**

```
public void disconnect(IConnection conn);
```

Removes given connection from list of scope connections. This disconnects all clients of given connection from the scope.

**Parameters**

<code>conn</code>	Connection given
-------------------	------------------

**21.10. getBasicScope(String, String)**

```
public IBasicScope getBasicScope(String type,
                                 String name);
```

Get a child scope by name.

**Parameters**

<i>name</i>	Name of the child scope
<i>type</i>	Child scope type
<i>return</i>	the child scope, or null if no scope is found

## 21.11. getClients()

```
public java.util.Set<org.red5.server.api.IClient> getClients();
```

Get a set of connected clients. You can get the connections by passing the scope to the clients `getConnections()` method.

### Parameters

<i>return</i>	Set containing all connected clients
---------------	--------------------------------------

### See Also

`getConnections(org.red5.server.api.IScope)`

## 21.12. getConnections()

```
public java.util.Iterator<org.red5.server.api.IConnection> getConnections();
```

Get a connection iterator. You can call remove, and the connection will be closed.

### Parameters

<i>return</i>	Iterator holding all connections
---------------	----------------------------------

## 21.13. getContext()

```
public IContext getContext();
```

Returns scope context

### Parameters

<i>return</i>	Scope context
---------------	---------------

## 21.14. getContextPath()

```
public String getContextPath();
```

Return context path.

### Parameters

<i>return</i>	Context path
---------------	--------------

## 21.15. getHandler()

```
public IScopeHandler getHandler();
```

Return handler of the scope

#### Parameters

<i>return</i>	Scope handler
---------------	---------------

## 21.16. getScope(String)

```
public IScope getScope(String name);
```

Return scope by name

#### Parameters

name	Scope name
------	------------

<i>return</i>	Scope with given name
---------------	-----------------------

## 21.17. getScopeNames()

```
public java.util.Iterator<java.lang.String> getScopeNames();
```

Get a set of the child scope names.

#### Parameters

<i>return</i>	set containing child scope names
---------------	----------------------------------

## 21.18. getStatistics()

```
public org.red5.server.api.statistics.IScopeStatistics getStatistics();
```

Return statistics informations about the scope.

#### Parameters

<i>return</i>	the statistics
---------------	----------------

## 21.19. hasChildScope(String)

```
public boolean hasChildScope(String name);
```

Check to see if this scope has a child scope matching a given name.

#### Parameters

name	the name of the child scope
------	-----------------------------

<i>return</i>	true if a child scope exists, otherwise false
---------------	---

## 21.20. hasChildScope(String, String)

```
public boolean hasChildScope(String type,
```

```
String name);
```

Checks whether scope has a child scope with given name and type

#### Parameters

type	Child scope type
name	Child scope name
<i>return</i>	<code>true</code> if a child scope exists, otherwise <code>false</code>

## 21.21. hasHandler()

```
public boolean hasHandler();
```

Checks whether scope has handler or not.

#### Parameters

<i>return</i>	<code>true</code> if scope has a handler, <code>false</code> otherwise
---------------	--

## 21.22. lookupConnections(IClient)

```
public java.util.Set<org.red5.server.api.IConnection> lookupConnections(IClient client);
```

Lookup connections.

#### Parameters

client	object
<i>return</i>	Set of connection objects (readonly)

## 21.23. removeChildScope(IBasicScope)

```
public void removeChildScope(IBasicScope scope);
```

Removes scope from the children scope list. Returns `false` if given scope isn't a child of the current scope.

#### Parameters

scope	Scope given
-------	-------------

# 22. Interface IScopeAware

Marker interface for all objects that are aware of the scope they are located in.

## 22.1. Synopsis

```
public interface IScopeAware {
    // Public Methods
}
```

```
public void setScope(IScope scope);

}
```

## 22.2. setScope(IScope)

```
public void setScope(IScope scope);
```

Set the scope the object is located in.

### Parameters

scope	Scope for this object
-------	-----------------------

## 23. Interface IScopeHandler

The scope handler controls actions performed against a scope object, and also is notified of all events. Gives fine grained control over what actions can be performed with the can\* methods. Allows for detailed reporting on what is happening within the scope with the on\* methods. This is the core interface users implement to create applications. The thread local connection is always available via the Red5 object within these methods

### 23.1. Synopsis

```
public interface IScopeHandler extends, org.?red5.?server.?api.?event.?IEventHandler {
// Public Methods

    public boolean addChildScope(IBasicScope scope);

    public boolean connect(IConnection conn,
                          IScope scope,
                          Object[] params);

    public void disconnect(IConnection conn,
                          IScope scope);

    public boolean join(IClient client,
                      IScope scope);

    public void leave(IClient client,
                     IScope scope);

    public void removeChildScope(IBasicScope scope);

    public boolean serviceCall(IConnection conn,
                             org.red5.server.api.service.IServiceCall call);

    public boolean start(IScope scope);

    public void stop(IScope scope);

}
```

## 23.2. addChildScope(IBasicScope)

```
public boolean addChildScope(IBasicScope scope);
```

Called just before a child scope is added.

### Parameters

scope	Scope that will be added
<i>return</i>	<code>true</code> to allow, <code>false</code> to deny

## 23.3. connect(IConnection, IScope, Object[])

```
public boolean connect(IConnection conn,
                      IScope scope,
                      Object[] params);
```

Called just before every connection to a scope. You can pass additional params from client using `NetConnection.connect` method (see below).

### Parameters

conn	Connection object
params	List of params passed from client via <code>NetConnection.connect</code> method. All parameters but the first one passed to <code>NetConnection.connect</code> method are available as params array.
scope	Scope object
<i>return</i>	<code>true</code> to allow, <code>false</code> to deny

## 23.4. disconnect(IConnection, IScope)

```
public void disconnect(IConnection conn,
                      IScope scope);
```

Called just after the a connection is disconnected.

### Parameters

conn	Connection object
scope	Scope object

## 23.5. join(IClient, IScope)

```
public boolean join(IClient client,
                   IScope scope);
```

Called just before a client enters the scope.

### Parameters

client	Client object
scope	Scope that is joined by client
<i>return</i>	<code>true</code> to allow, <code>false</code> to deny connection

## 23.6. leave(IClient, IScope)

```
public void leave(IClient client,
                  IScope scope);
```

Called just after the client leaves the scope.

Parameters	
client	Client object
scope	Scope object

## 23.7. removeChildScope(IBasicScope)

```
public void removeChildScope(IBasicScope scope);
```

Called just after a child scope has been removed.

Parameters	
scope	Scope that has been removed

## 23.8. serviceCall(IConnection, IServiceCall)

```
public boolean serviceCall(IConnection conn,
                           org.red5.server.api.service(IServiceCall call);
```

Called when a service is called.

Parameters	
conn	The connection object
call	The call object.
<i>return</i>	<code>true</code> to allow, <code>false</code> to deny

## 23.9. start(IScope)

```
public boolean start(IScope scope);
```

Called when a scope is created for the first time.

Parameters	
scope	the new scope object
<i>return</i>	<code>true</code> to allow, <code>false</code> to deny

## 23.10. stop(IScope)

```
public void stop(IScope scope);
```

Called just before a scope is disposed.

### Parameters

scope	Scope that is disposed
-------	------------------------

# 24. Interface IScopeResolver

Resolve the scope from given a host and path. Resolver implementations depend on context naming strategy and so forth.

## 24.1. Synopsis

```
public interface IScopeResolver {
    // Public Methods

    public IGlobalScope getGlobalScope();

    public IScope resolveScope(String path);

    public IScope resolveScope(IScope root,
                           String path);

}
```

## 24.2. getGlobalScope()

```
public IGlobalScope getGlobalScope();
```

Return the global scope.

### Parameters

return	Global scope
--------	--------------

## 24.3. resolveScope(IScope, String)

```
public IScope resolveScope(IScope root,
                           String path);
```

Get the scope for a given path from a root scope.

### Parameters

root	The scope to start traversing from.
path	Path to return the scope for.

<i>return</i>	Scope for passed path.
---------------	------------------------

## 24.4. resolveScope(String)

```
public IScope resolveScope(String path);
```

Get the scope for a given path.

### Parameters

path	Path to return the scope for
<i>return</i>	Scope for passed path

ScopeNotFoundException

If scope doesn't exist an can't be created

## 25. Interface IScopeService

Base marker interface for all scope services. Used by the ScopeUtils to lookup services defined as beans in Spring application context. A scope service usually can perform various tasks on a scope like managing shared objects, streams, etc.

### 25.1. Synopsis

```
public interface IScopeService {
    // Public Static Fields

    public static final String BEAN_NAME ;
}
```

### 25.2. BEAN\_NAME

```
public static final String BEAN_NAME ;
```

Name of a bean defining that scope service. Override in subinterfaces.

## 26. Interface IServer

The interface that represents the Red5 server.

### 26.1. Synopsis

```
public interface IServer {
    // Public Static Fields

    public static final String ID = "red5.server";
}

// Public Methods

public void addListener(org.red5.server.api.listeners.IConnectionListener listener);
```

```

public void addListener(org.red5.server.api.listeners.IScopeListener listener);

public boolean addMapping(String hostName,
                          String contextPath,
                          String globalName);

public IGlobalScope getGlobal(String name);

public java.util.Iterator<java.lang.String> getGlobalNames();

public java.util.Iterator<org.red5.server.api.IGlobalScope> getGlobalScopes();

public java.util.Map<java.lang.String, java.lang.String> getMappingTable();

public IGlobalScope lookupGlobal(String hostName,
                                 String contextPath);

public void registerGlobal(IGlobalScope scope);

public void removeListener(org.red5.server.api.listeners.IConnectionListener listener);

public void removeListener(org.red5.server.api.listeners.IScopeListener listener);

public boolean removeMapping(String hostName,
                            String contextPath);

}

```

## 26.2. ID

```
public static final String ID = "red5.server";
```

Server ID

## 26.3. addListener(IConnectionListener)

```
public void addListener(org.red5.server.api.listeners.IConnectionListener listener);
```

Add listener to get notified about connection events.

### Parameters

listener	the listener to add
----------	---------------------

## 26.4. addListener(IScopeListener)

```
public void addListener(org.red5.server.api.listeners.IScopeListener listener);
```

Add listener to get notified about scope events.

### Parameters

listener	the listener to add
----------	---------------------

## 26.5. addMapping(String, String, String)

```
public boolean addMapping(String hostName,
                         String contextPath,
                         String globalName);
```

Map a virtual hostname and a path to the name of a global scope.

Parameters	
hostName	The name of the host to map
contextPath	The path to map
globalName	The name of the global scope to map to
<i>return</i>	<code>true</code> if the name was mapped, otherwise <code>false</code>

## 26.6. getGlobal(String)

```
public IGlobalScope getGlobal(String name);
```

Get the global scope with given name.

Parameters	
name	Name of the global scope
<i>return</i>	the global scope

## 26.7. getGlobalNames()

```
public java.util.Iterator<java.lang.String> getGlobalNames();
```

Get list of global scope names.

Parameters	
<i>return</i>	Iterator for names of global scopes

## 26.8. getGlobalScopes()

```
public java.util.Iterator<org.red5.server.api.IGlobalScope> getGlobalScopes();
```

Get list of global scopes.

Parameters	
<i>return</i>	Iterator for global scopes objects

## 26.9. getMappingTable()

```
public java.util.Map<java.lang.String, java.lang.String> getMappingTable();
```

Query informations about the global scope mappings.

**Parameters**

<i>return</i>	Map containing informations about the mappings
---------------	--

**26.10. lookupGlobal(String, String)**

```
public IGlobalScope lookupGlobal(String hostName,
                                String contextPath);
```

Lookup the global scope for a host.

**Parameters**

hostName	The name of the host
contextPath	The path in the host
<i>return</i>	The found global scope or <code>null</code>

**26.11. registerGlobal(IGlobalScope)**

```
public void registerGlobal(IGlobalScope scope);
```

Register a global scope.

**Parameters**

scope	The global scope to register
-------	------------------------------

**26.12. removeListener(IConnectionListener)**

```
public void removeListener(org.red5.server.api.listeners.IConnectionListener listener);
```

Remove listener that got notified about connection events.

**Parameters**

listener	the listener to remove
----------	------------------------

**26.13. removeListener(IScopeListener)**

```
public void removeListener(org.red5.server.api.listeners.IScopeListener listener);
```

Remove listener that got notified about scope events.

**Parameters**

listener	the listener to remove
----------	------------------------

**26.14. removeMapping(String, String)**

```
public boolean removeMapping(String hostName,
                            String contextPath);
```

Unregister a previously mapped global scope.

### Parameters

hostName	The name of the host to unmap
contextPath	The path for this host to unmap
<i>return</i>	<code>true</code> if the global scope was unmapped, otherwise <code>false</code>

## 27. Class Red5

Utility class for accessing Red5 API objects. This class uses a thread local, and will be setup by the service invoker. The code below shows various uses.

```
IConnection conn = Red5.getConnectionLocal(); Red5 r5 = new Red5(); IScope scope = r5.getScope(); conn = r5.getConnection(); r5 = new Red5(conn); IClient client = r5.getClient();
```

### 27.1. Synopsis

```
public final class Red5 implements, org.?red5.?server.?api.?Red5MBean {
    // Public Static Fields

    public static final String VERSION = "Red5 Server 0.7.1-dev $Revision: 2887 $";

    // Public Fields

    public IConnection conn;

    // Public Constructors

    public Red5();

    public Red5(IConnection conn);

    // Public Static Methods

    public static IConnection getConnectionLocal();

    public static long getUpTime();

    public static String getVersion();

    public static void setConnectionLocal(IConnection connection);

    // Public Methods

    public IClient getClient();

    public IConnection getConnection();

    public IContext getContext();

    public IScope getScope();

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 27.2. Red5()

```
public Red5();
```

Create a new Red5 object using the connection local to the current thread A bit of magic that lets you access the red5 scope from anywhere

## 27.3. Red5(IConnection)

```
public Red5(IConnection conn);
```

Create a new Red5 object using given connection.

### Parameters

conn	Connection object.
------	--------------------

## 27.4. conn

```
public IConnection conn;
```

Connection local to the current thread

## 27.5. VERSION

```
public static final String VERSION = "Red5 Server 0.7.1-dev $Revision: 2887 $";
```

Current server version with revision

## 27.6. getClient()

```
public IClient getClient();
```

**Specified by:** Method getClient in interface Red5MBean

Get the client

### Parameters

return	Client object
--------	---------------

## 27.7. getConnection()

```
public IConnection getConnection();
```

**Specified by:** Method getConnection in interface Red5MBean

Get the connection object.

**Parameters**

<i>return</i>	Connection object
---------------	-------------------

**27.8. getConnectionLocal()**

```
public static IConnection getConnectionLocal();
```

Get the connection associated with the current thread. This method allows you to get connection object local to current thread. When you need to get a connection associated with event handler and so forth, this method provides you with it.

**Parameters**

<i>return</i>	Connection object
---------------	-------------------

**27.9. getContext()**

```
public IContext getContext();
```

**Specified by:** Method getContext in interface Red5MBean

Get the spring application context

**Parameters**

<i>return</i>	Application context
---------------	---------------------

**27.10. getScope()**

```
public IScope getScope();
```

**Specified by:** Method getScope in interface Red5MBean

Get the scope

**Parameters**

<i>return</i>	Scope object
---------------	--------------

**27.11. getUpTime()**

```
public static long getUpTime();
```

Returns server uptime in milliseconds.

**Parameters**

<i>return</i>	String version
---------------	----------------

**27.12. getVersion()**

```
public static String getVersion();
```

Returns the current version with revision number

#### Parameters

<i>return</i>	String version
---------------	----------------

## 27.13. setConnectionLocal(IConnection)

```
public static void setConnectionLocal(IConnection connection);
```

Setter for connection

#### Parameters

<i>connection</i>	Thread local connection
-------------------	-------------------------

## 28. Interface Red5MBean

JMX interface for accessing Red5 API objects

### 28.1. Synopsis

```
public interface Red5MBean {
// Public Methods
```

```
    public IClient getClient();
```

```
    public IConnection getConnection();
```

```
    public IContext getContext();
```

```
    public IScope getScope();
```

```
}
```

## 29. Interface ScopeMBean

The scope object. A statefull object shared between a group of clients connected to the same `context path`. Scopes are arranged in hierarchical way, so its possible for a scope to have a parent and children scopes. If a client connects to a scope then they are also connected to its parent scope. The scope object is used to access resources, shared object, streams, etc. That is, scope are general option for grouping things in application. The following are all names for scopes: application, room, place, lobby.

### 29.1. Synopsis

```
public interface ScopeMBean {
// Public Static Fields
```

```
    public static final String ID = "red5.scope";
```

```
    public static final String SEPARATOR = ":";
```

```

public static final String TYPE = "scope";

// Public Methods

public boolean addChildScope(IBasicScope scope);

public boolean connect(IConnection conn);

public boolean connect(IConnection conn,
                      Object[] params);

public boolean createChildScope(String name);

public void disconnect(IConnection conn);

public IBasicScope getBasicScope(String type,
                                 String name);

public java.util.Iterator<java.lang.String> getBasicScopeNames(String type);

public java.util.Set<org.red5.server.api.IClient> getClients();

public java.util.Iterator<org.red5.server.api.IConnection> getConnections();

public IContext getContext();

public String getContextPath();

public IScopeHandler getHandler();

public IScope getScope(String name);

public java.util.Iterator<java.lang.String> getScopeNames();

public boolean hasChildScope(String name);

public boolean hasChildScope(String type,
                            String name);

public boolean hasHandler();

public java.util.Set<org.red5.server.api.IConnection> lookupConnections(IClient client);

public void removeChildScope(IBasicScope scope);

}

```

## 29.2. ID

```

public static final String ID = "red5.scope";

```

ID constant

## 29.3. SEPARATOR

```
public static final String SEPARATOR = ":";
```

Scope separator

## 29.4. TYPE

```
public static final String TYPE = "scope";
```

Type constant

## 29.5. addChildScope(IBasicScope)

```
public boolean addChildScope(IBasicScope scope);
```

Adds scope as a child scope. Returns `true` on success, `false` if given scope is already a child of current.

### Parameters

<code>scope</code>	Scope given
<code>return</code>	<code>true</code> if child scope was successfully added, <code>false</code> otherwise

## 29.6. connect(IConnection)

```
public boolean connect(IConnection conn);
```

Adds given connection to the scope

### Parameters

<code>conn</code>	Given connection
<code>return</code>	<code>true</code> on success, <code>false</code> if given connection already belongs to this scope

## 29.7. connect(IConnection, Object[])

```
public boolean connect(IConnection conn,
                      Object[] params);
```

Add given connection to the scope, overloaded for parameters pass case.

### Parameters

<code>conn</code>	Given connection
<code>params</code>	Parameters passed
<code>return</code>	<code>true</code> on success, <code>false</code> if given connection already belongs to this scope

## 29.8. createChildScope(String)

```
public boolean createChildScope(String name);
```

Creates child scope with name given and returns success value. Returns `true` on success, `false` if given scope already exists among children.

#### Parameters

<code>name</code>	New child scope name
<code>return</code>	<code>true</code> if child scope was successfully creates, <code>false</code> otherwise

## 29.9. disconnect(IConnection)

```
public void disconnect(IConnection conn);
```

Removes given connection from list of scope connections. This disconnects all clients of given connection from the scope.

#### Parameters

<code>conn</code>	Connection given
-------------------	------------------

## 29.10. getBasicScope(String, String)

```
public IBasicScope getBasicScope(String type,
                                 String name);
```

Get a child scope by name.

#### Parameters

<code>name</code>	Name of the child scope
<code>type</code>	Child scope type
<code>return</code>	the child scope, or null if no scope is found

## 29.11. getClients()

```
public java.util.Set<org.red5.server.api.IClient> getClients();
```

Get a set of connected clients. You can get the connections by passing the scope to the `clients.getConnections()` method.

#### Parameters

<code>return</code>	Set containing all connected clients
---------------------	--------------------------------------

#### See Also

`getConnections(org.red5.server.api.IScope)`

## 29.12. getConnections()

```
public java.util.Iterator<org.red5.server.api.IConnection> getConnections();
```

Get a connection iterator. You can call `remove`, and the connection will be closed.

**Parameters**

<i>return</i>	Iterator holding all connections
---------------	----------------------------------

**29.13. getContext()**

```
public IContext getContext();
```

Returns scope context

**Parameters**

<i>return</i>	Scope context
---------------	---------------

**29.14. getContextPath()**

```
public String getContextPath();
```

Return context path.

**Parameters**

<i>return</i>	Context path
---------------	--------------

**29.15. getHandler()**

```
public IScopeHandler getHandler();
```

Return handler of the scope

**Parameters**

<i>return</i>	Scope handler
---------------	---------------

**29.16. getScope(String)**

```
public IScope getScope(String name);
```

Return scope by name

**Parameters**

name	Scope name
<i>return</i>	Scope with given name

**29.17. getScopeNames()**

```
public java.util.Iterator<java.lang.String> getScopeNames();
```

Get a set of the child scope names.

**Parameters**

<i>return</i>	set containing child scope names
---------------	----------------------------------

## 29.18. hasChildScope(String)

```
public boolean hasChildScope(String name);
```

Check to see if this scope has a child scope matching a given name.

### Parameters

name	the name of the child scope
<i>return</i>	<code>true</code> if a child scope exists, otherwise <code>false</code>

## 29.19. hasChildScope(String, String)

```
public boolean hasChildScope(String type,
                             String name);
```

Checks whether scope has a child scope with given name and type

### Parameters

type	Child scope type
name	Child scope name
<i>return</i>	<code>true</code> if a child scope exists, otherwise <code>false</code>

## 29.20. hasHandler()

```
public boolean hasHandler();
```

Checks whether scope has handler or not.

### Parameters

<i>return</i>	<code>true</code> if scope has a handler, <code>false</code> otherwise
---------------	--

## 29.21. lookupConnections(IClient)

```
public java.util.Set<org.red5.server.api.IConnection> lookupConnections(IClient client);
```

Lookup connections.

### Parameters

client	object
<i>return</i>	Set of connection objects (readonly)

## 29.22. removeChildScope(IBasicScope)

```
public void removeChildScope(IBasicScope scope);
```

Removes scope from the children scope list. Returns `false` if given scope isn't a child of the current scope.

#### Parameters

scope	Scope given
-------	-------------

## 30. Class ScopeUtils

Collection of utils for working with scopes

### 30.1. Synopsis

```
public class ScopeUtils {
// Public Constructors

    public ScopeUtils();

// Public Static Methods

    public static IScope findApplication(IScope from);

    public static IScope findRoot(IScope from);

    public static Object getScopeService(IScope scope,
                                         Class<?> intf);

    public static Object getScopeService(IScope scope,
                                         Class<?> intf,
                                         boolean checkHandler);

    public static Object getScopeService(IScope scope,
                                         Class<?> intf,
                                         Class<?> defaultClass);

    public static Object getScopeService(IScope scope,
                                         Class<?> intf,
                                         Class<?> defaultClass,
                                         boolean checkHandler);

    public static boolean isAncestor(IBasicScope from,
                                    IBasicScope ancestor);

    public static boolean isApp(IBasicScope scope);

    public static boolean isGlobal(IBasicScope scope);

    public static boolean isRoom(IBasicScope scope);

    public static boolean isRoot(IBasicScope scope);

    public static IScope resolveScope(IScope from,
                                    String path);

// Protected Methods

    protected static Object getScopeService(IScope scope,
                                         String name);
}
```

```

protected static Object getScopeService(IScope scope,
                                      String name,
                                      Class<?> defaultClass);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 30.2. findApplication(IScope)

```
public static IScope findApplication(IScope from);
```

Returns the application scope for specified scope. Application scope has depth of 1 and has no parent. See `isApp` method for details.

### Parameters

from	Scope to find application for
<i>return</i>	Application scope.

## 30.3. findRoot(IScope)

```
public static IScope findRoot(IScope from);
```

Finds root scope for specified scope object. Root scope is the top level scope among scope's parents.

### Parameters

from	Scope to find root for
<i>return</i>	Root scope object

## 30.4. getScopeService(IScope, Class<?>)

```
public static Object getScopeService(IScope scope,
                                    Class<?> intf);
```

Returns scope service that implements a given interface.

### Parameters

scope	The scope service belongs to
intf	The interface the service must implement
<i>return</i>	Service object

## 30.5. getScopeService(IScope, Class<?>, Class<?>)

```
public static Object getScopeService(IScope scope,
                                    Class<?> intf,
```

```
Class<?> defaultClass);
```

Returns scope service that implements a given interface.

Parameters	
scope	The scope service belongs to
intf	The interface the service must implement
defaultClass	Class that should be used to create a new service if no service was found.
<i>return</i>	Service object

### 30.6. getScopeService(IScope, String)

```
protected static Object getScopeService(IScope scope,
                                         String name);
```

Returns scope service by bean name. See overloaded method for details.

Parameters	
scope	
name	
<i>return</i>	

### 30.7. getScopeService(IScope, String, Class<?>)

```
protected static Object getScopeService(IScope scope,
                                         String name,
                                         Class<?> defaultClass);
```

Returns scope services (e.g. SharedObject, etc) for the scope. Method uses either bean name passes as a string or class object.

Parameters	
scope	The scope service belongs to
name	Bean name
defaultClass	Class of service
<i>return</i>	Service object

### 30.8. isAncestor(IBasicScope, IBasicScope)

```
public static boolean isAncestor(IBasicScope from,
                                 IBasicScope ancestor);
```

Check whether one scope is an ancestor of another

Parameters	

from	Scope
ancestor	Scope to check
<i>return</i>	<code>true</code> if ancestor scope is really an ancestor of scope passed as from parameter, <code>false</code> otherwise.

## 30.9. isApp(IBasicScope)

```
public static boolean isApp(IBasicScope scope);
```

Check whether scope is an application scope (level 1 leaf in scope tree) or not

Parameters	
scope	Scope to check
<i>return</i>	<code>true</code> if scope is an application scope, <code>false</code> otherwise.

## 30.10. isGlobal(IBasicScope)

```
public static boolean isGlobal(IBasicScope scope);
```

Check whether scope is the global scope (level 0 leaf in scope tree) or not When user connects the following URL: rtmp://localhost/myapp/foo/bar then / is the global level scope, myapp is app level, foo is room level and bar is room level as well (but with higher depth level)

Parameters	
scope	Scope to check
<i>return</i>	<code>true</code> if scope is the global scope, <code>false</code> otherwise.

## 30.11. isRoom(IBasicScope)

```
public static boolean isRoom(IBasicScope scope);
```

Check whether scope is a room scope (level 2 leaf in scope tree or lower, e.g. 3, 4, ...) or not

Parameters	
scope	Scope to check
<i>return</i>	<code>true</code> if scope is a room scope, <code>false</code> otherwise.

## 30.12. isRoot(IBasicScope)

```
public static boolean isRoot(IBasicScope scope);
```

Checks whether scope is root or not

Parameters	
------------	--

scope	Scope to check
<i>return</i>	true if scope is root scope (top level scope), false otherwise.

### 30.13. resolveScope(IScope, String)

```
public static IScope resolveScope(IScope from,  
                                 String path);
```

Resolves scope for specified scope and path.

Parameters	
from	Scope to use as context (to start from)
path	Path to resolve
<i>return</i>	Resolved scope

# 1. Interface ICacheStore

Storage for cacheable objects. Selected cache engines must implement this interface.

## 1.1. Synopsis

```
public interface ICacheStore {  
    // Public Methods  
  
    public void destroy();  
  
    public ICacheable get(String name);  
  
    public java.util.Iterator<java.lang.String> getObjectNames();  
  
    public java.util.Iterator<java.lang.ref.SoftReference<? extends org.red5.server.api.cache.ICacheable>> offer(String name,  
        Object obj);  
  
    public void put(String name,  
        Object obj);  
  
    public boolean remove(String name);  
  
    public boolean remove(ICacheable obj);  
  
    public void setMaxEntries(int max);  
}
```

### See Also

Soft references provide for quick-and-dirty caching [<http://www-128.ibm.com/developerworks/java/library/j-jtp01246.html>] , Reference Objects and Garbage Collection [<http://java.sun.com/developer/technicalArticles/ALT/RefObj/>] , Top Ten New Things You Can Do with NIO [<http://www.onjava.com/pub/a/onjava/2002/10/02/javanio.html?page=3>] , [http://csci.csusb.edu/turner/archive/courses/aiit2004/proxy\\_cache\\_solution.html](http://csci.csusb.edu/turner/archive/courses/aiit2004/proxy_cache_solution.html)

## 1.2. destroy()

```
public void destroy();
```

Allows for cleanup of a cache implementation.

## 1.3. get(String)

```
public ICacheable get(String name);
```

Return a cached object with the given name.

### Parameters

<i>name</i>	the name of the object to return
<i>return</i>	the object or <code>null</code> if no such object was found

## 1.4. `getObjectNames()`

```
public java.util.Iterator<java.lang.String> getObjectNames();
```

Return iterator over the names of all already loaded objects in the storage.

Parameters	
<i>return</i>	iterator over all objects names

## 1.5. `getObjects()`

```
public java.util.Iterator<java.lang.ref.SoftReference<? extends org.red5.server.api.cache.ICacheable>> getObjects();
```

Return iterator over the already loaded objects in the storage.

Parameters	
<i>return</i>	iterator over all objects

## 1.6. `offer(String, Object)`

```
public boolean offer(String name,  
                     Object obj);
```

Offer an object to the cache with an associated key. If the named object exists in cache, it will not be accepted.

Parameters	
<i>name</i>	string name representing the object
<i>obj</i>	cacheable object
<i>return</i>	true if accepted, false otherwise

## 1.7. `put(String, Object)`

```
public void put(String name,  
                Object obj);
```

Puts an object in the cache with the associated key.

Parameters	
<i>name</i>	string name representing the object
<i>obj</i>	cacheable object

## 1.8. remove(ICacheable)

```
public boolean remove(ICacheable obj);
```

Delete the passed cached object.

### Parameters

obj	the object to delete
<i>return</i>	

## 1.9. remove(String)

```
public boolean remove(String name);
```

Delete the cached object with the given name.

### Parameters

name	the name of the object to delete
<i>return</i>	

## 1.10. setMaxEntries(int)

```
public void setMaxEntries(int max);
```

Sets the maximum number of entries for the cache.

### Parameters

max	upper-limit of the cache
-----	--------------------------

## 2. Interface ICacheable

Base interface for objects that can be made cacheable.

### 2.1. Synopsis

```
public interface ICacheable {
// Public Methods

    public org.apache.mina.common.ByteBuffer getByteBuffer();

    public byte[] getBytes();

    public String getName();

    public boolean isCached();

    public void setCached(boolean cached);

    public void setName(String name);
}
```

```
}
```

**See Also**

[org.red5.server.api.cache.ICacheStore](#)

**2.2. getByteBuffer()**

```
public org.apache.mina.common.ByteBuffer getByteBuffer();
```

Returns a readonly byte buffer.

**Parameters**

<i>return</i>	Read-only byte buffer with cached data
---------------	--

**2.3. getBytes()**

```
public byte[] getBytes();
```

Returns the object contained within the cacheable reference.

**Parameters**

<i>return</i>	Cached representation of object
---------------	---------------------------------

**2.4. getName()**

```
public String getName();
```

Returns the name of the cached object.

**Parameters**

<i>return</i>	Object name
---------------	-------------

**2.5. isCached()**

```
public boolean isCached();
```

Returns `true` if the object is cached, `false` otherwise.

**Parameters**

<i>return</i>	<code>true</code> if object is cached, <code>false</code> otherwise
---------------	---

**2.6. setCached(boolean)**

```
public void setCached(boolean cached);
```

Sets a flag to represent the cached status of a cacheable object.

**Parameters**

cached

`true` if object is cached, `false` otherwise

## 2.7. setName(String)

```
public void setName(String name);
```

Set the name of the cached object.

### Parameters

name

New object name

# 1. Interface IEvent

IEvent interfaces is the essential interface every Event should implement

## 1.1. Synopsis

```
public interface IEvent {  
    // Public Methods  
  
    public Object getObject();  
  
    public IEventListener getSource();  
  
    public org.red5.server.api.event.IEvent.Type getType();  
  
    public boolean hasSource();  
}
```

## 1.2. getObject()

```
public Object getObject();
```

Returns event context object

### Parameters

<i>return</i>	Event context object
---------------	----------------------

## 1.3. getSource()

```
public IEventListener getSource();
```

Returns event listener

### Parameters

<i>return</i>	Event listener object
---------------	-----------------------

## 1.4. getType()

```
public org.red5.server.api.event.IEvent.Type getType();
```

Returns even type

### Parameters

<i>return</i>	Event type enumeration
---------------	------------------------

## 1.5. hasSource()

```
public boolean hasSource();
```

Whether event has source (event listener(s))

#### Parameters

<i>return</i>	true if so, false otherwise
---------------	-----------------------------

## 2. Class IEvent.Type

```
public static final class IEvent.Type extends java.lang.Enum {
// Public Static Fields

    public static final org.red5.server.api.event.IEvent.Type CLIENT ;

    public static final org.red5.server.api.event.IEvent.Type SERVER ;

    public static final org.red5.server.api.event.IEvent.Type SERVICE_CALL ;

    public static final org.red5.server.api.event.IEvent.Type SHARED_OBJECT ;

    public static final org.red5.server.api.event.IEvent.Type STATUS ;

    public static final org.red5.server.api.event.IEvent.Type STREAM_CONTROL ;

    public static final org.red5.server.api.event.IEvent.Type STREAM_DATA ;

    public static final org.red5.server.api.event.IEvent.Type SYSTEM ;

// Public Static Methods

    public static org.red5.server.api.event.IEvent.Type valueOf(String name);

    public static org.red5.server.api.event.IEvent.Type[] values();

}
```

**Methods inherited from java.lang.Enum:** clone, compareTo, equals, finalize, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

**Methods inherited from java.lang.Object:** getClass, notify, notifyAll, wait

## 3. Interface IEventDispatcher

IEventDispatcher interface implementations dispatch events

### 3.1. Synopsis

```
public interface IEventDispatcher {
// Public Methods

    public void dispatchEvent(IEvent event);

}
```

### 3.2. dispatchEvent(IEvent)

```
public void dispatchEvent(IEvent event);
```

Dispatches event

#### Parameters

event	Event object
-------	--------------

## 4. Interface IEventHandler

```
public interface IEventHandler {
// Public Methods

    public boolean handleEvent(IEvent event);

}
```

### 4.1. handleEvent(IEvent)

```
public boolean handleEvent(IEvent event);
```

Handle an event

#### Parameters

event	event to handle
<i>return</i>	true if event was handled, false if it should bubble

## 5. Interface IEventListener

```
public interface IEventListener {
// Public Methods

    public void notifyEvent(IEvent event);

}
```

### 5.1. notifyEvent(IEvent)

```
public void notifyEvent(IEvent event);
```

Notify of event.

#### Parameters

event	the event object
-------	------------------

## 6. Interface IEventObservable

IEventObservable hold functionality of the well-known Observer pattern, that is it has a list of objects that listen to events.

### 6.1. Synopsis

```
public interface IEventObservable {
    // Public Methods

    public void addEventListener(IEventListener listener);

    public java.util.Iterator<org.red5.server.api.event.IEventListener> getEventListeners();

    public void removeEventListener(IEventListener listener);

}
```

### 6.2. addEventListener(IEventListener)

```
public void addEventListener(IEventListener listener);
```

Add event listener to this observable

#### Parameters

listener	Event listener
----------	----------------

### 6.3. getEventListeners()

```
public java.util.Iterator<org.red5.server.api.event.IEventListener> getEventListeners();
```

Iterator for event listeners

#### Parameters

return	Event listeners iterator
--------	--------------------------

### 6.4. removeEventListener(IEventListener)

```
public void removeEventListener(IEventListener listener);
```

Remove event listener from this observable

#### Parameters

listener	Event listener
----------	----------------

# 1. Interface IConnectionListener

Interface for listeners to connection events.

## 1.1. Synopsis

```
public interface IConnectionListener {  
    // Public Methods  
  
    public void notifyConnected(org.red5.server.api.IConnection conn);  
  
    public void notifyDisconnected(org.red5.server.api.IConnection conn);  
}
```

## 1.2. notifyConnected(IConnection)

```
public void notifyConnected(org.red5.server.api.IConnection conn);
```

A new connection was established.

### Parameters

conn	the new connection
------	--------------------

## 1.3. notifyDisconnected(IConnection)

```
public void notifyDisconnected(org.red5.server.api.IConnection conn);
```

A connection was disconnected.

### Parameters

conn	the disconnected connection
------	-----------------------------

# 2. Interface IScopeListener

Interface for listeners to scope events.

## 2.1. Synopsis

```
public interface IScopeListener {  
    // Public Methods  
  
    public void notifyScopeCreated(org.red5.server.api.IScope scope);  
  
    public void notifyScopeRemoved(org.red5.server.api.IScope scope);  
}
```

## 2.2. notifyScopeCreated(IScope)

```
public void notifyScopeCreated(org.red5.server.api.IScope scope);
```

A scope has been created.

### Parameters

scope	the new scope
-------	---------------

## 2.3. notifyScopeRemoved(IScope)

```
public void notifyScopeRemoved(org.red5.server.api.IScope scope);
```

A scope has been removed.

### Parameters

scope	the removed scope
-------	-------------------

# 1. Interface IPersistable

Base interface for objects that can be made persistent. Every object that complies to this interface must provide either a constructor that takes an input stream as only parameter or an empty constructor so it can be loaded from the persistence store. However this is not required for objects that are created by the application and initialized afterwards.

## 1.1. Synopsis

```
public interface IPersistable {  
    // Public Static Fields  
  
    public static final String TRANSIENT_PREFIX = "_transient";  
  
    // Public Methods  
  
    public void deserialize(org.red5.io.object.Input input)  
        throws IOException;  
  
    public long getLastModified();  
  
    public String getName();  
  
    public String getPath();  
  
    public IPersistenceStore getStore();  
  
    public String getType();  
  
    public boolean isPersistent();  
  
    public void serialize(org.red5.io.object.Output output)  
        throws IOException;  
  
    public void setName(String name);  
  
    public void setPath(String path);  
  
    public void setPersistent(boolean persistent);  
  
    public void setStore(IPersistenceStore store);  
}
```

### See Also

`org.red5.io.object.Input` , `load(java.lang.String)`

## 1.2. TRANSIENT\_PREFIX

```
public static final String TRANSIENT_PREFIX = "_transient";
```

Prefix for attribute names that should not be made persistent.

### 1.3. deserialize(Input)

```
public void deserialize(org.red5.io.object.Input input)
throws IOException;
```

Load the object from the passed input stream.

#### Parameters

<i>input</i>	Input stream to load from
--------------	---------------------------

`java.io.IOException`  
Any I/O exception

### 1.4. getLastModified()

```
public long getLastModified();
```

Returns the timestamp when the object was last modified.

#### Parameters

<i>return</i>	Last modification date in milliseconds
---------------	--

### 1.5. getName()

```
public String getName();
```

Returns the name of the persistent object.

#### Parameters

<i>return</i>	Object name
---------------	-------------

### 1.6. getPath()

```
public String getPath();
```

Returns the path of the persistent object.

#### Parameters

<i>return</i>	Persisted object path
---------------	-----------------------

### 1.7. getStore()

```
public IPersistenceStore getStore();
```

Returns the persistence store this object is stored in

#### Parameters

<i>return</i>	This object's persistence store
---------------	---------------------------------

## 1.8. getType()

```
public String getType();
```

Returns the type of the persistent object.

### Parameters

<i>return</i>	Object type
---------------	-------------

## 1.9. isPersistent()

```
public boolean isPersistent();
```

Returns `true` if the object is persistent, `false` otherwise.

### Parameters

<i>return</i>	<code>true</code> if object is persistent, <code>false</code> otherwise
---------------	---

## 1.10. serialize(Output)

```
public void serialize(org.red5.io.object.Output output)
throws IOException;
```

Write the object to the passed output stream.

### Parameters

<i>output</i>	Output stream to write to
---------------	---------------------------

`java.io.IOException`

Any I/O exception

## 1.11. setName(String)

```
public void setName(String name);
```

Set the name of the persistent object.

### Parameters

<i>name</i>	New object name
-------------	-----------------

## 1.12. setPath(String)

```
public void setPath(String path);
```

Set the path of the persistent object.

### Parameters

path	New persisted object path
------	---------------------------

## 1.13. setPersistent(boolean)

```
public void setPersistent(boolean persistent);
```

Set the persistent flag of the object.

### Parameters

persistent	true if object is persistent, false otherwise
------------	---

## 1.14. setStore(IPersistenceStore)

```
public void setStore(IPersistenceStore store);
```

Store a reference to the persistence store in the object.

### Parameters

store	Store the object is saved in
-------	------------------------------

## 2. Interface IPersistenceStore

Storage for persistent objects.

### 2.1. Synopsis

```
public interface IPersistenceStore {
    // Public Methods

    public java.util.Set<java.lang.String> getObjectNames();

    public java.util.Collection<org.red5.server.api.persistence.IPersistable> getObjects();

    public IPersistable load(String name);

    public boolean load(IPersistable obj);

    public void notifyClose();

    public boolean remove(String name);

    public boolean remove(IPersistable obj);

    public boolean save(IPersistable obj);

}
```

### 2.2. getObjectNames()

```
public java.util.Set<java.lang.String> getObjectNames();
```

Return iterator over the names of all already loaded objects in the storage.

#### Parameters

<i>return</i>	Set of all object names
---------------	-------------------------

## 2.3. **getObjects()**

```
public java.util.Collection<org.red5.server.api.persistence.IPersistable> getObjects();
```

Return iterator over the already loaded objects in the storage.

#### Parameters

<i>return</i>	Set of all objects
---------------	--------------------

## 2.4. **load(IPersistable)**

```
public boolean load(IPersistable obj);
```

Load state of an already instantiated persistent object.

#### Parameters

obj	the object to initialize
<i>return</i>	true if the object was initialized, false otherwise

## 2.5. **load(String)**

```
public IPersistable load(String name);
```

Load a persistent object with the given name. The object must provide either a constructor that takes an input stream as only parameter or an empty constructor so it can be loaded from the persistence store.

#### Parameters

name	the name of the object to load
<i>return</i>	The loaded object or <code>null</code> if no such object was found

## 2.6. **notifyClose()**

```
public void notifyClose();
```

Notify store that it's being closed. This allows the store to write any pending objects to disk.

## 2.7. **remove(IPersistable)**

```
public boolean remove(IPersistable obj);
```

Delete the passed persistent object.

**Parameters**

<i>obj</i>	the object to delete
<i>return</i>	<code>true</code> if object was persisted and thus can be removed, <code>false</code> otherwise

**2.8. remove(String)**

```
public boolean remove(String name);
```

Delete the persistent object with the given name.

**Parameters**

<i>name</i>	the name of the object to delete
<i>return</i>	<code>true</code> if object was persisted and thus can be removed, <code>false</code> otherwise

**2.9. save(IPersistable)**

```
public boolean save(IPersistable obj);
```

Persist given object.

**Parameters**

<i>obj</i>	Object to store
<i>return</i>	<code>true</code> on success, <code>false</code> otherwise

**3. Class PersistenceUtils**

Helper class for persistence.

**3.1. Synopsis**

```
public class PersistenceUtils {
    // Public Constructors

    public PersistenceUtils();

    // Public Static Methods

    public static IPersistenceStore getPersistenceStore(org.springframework.core.io.support.ResourcePa
                                                        String className)
        throws Exception;

}
```

**Methods inherited from java.lang.Object:** `clone` , `equals` , `finalize` , `getClass` , `hashCode` , `notify` , `notifyAll` , `toString` , `wait`

### 3.2. getPersistenceStore(ResourcePatternResolver, String)

```
public static IPersistenceStore getPersistenceStore(org.springframework.core.io.support.ResourcePatternResolver resolver, String className)
throws Exception;
```

Returns persistence store object. Persistence store is a special object that stores persistence objects and provides methods to manipulate them (save, load, remove, list).

Parameters	
resolver	Resolves connection pattern into Resource object
className	Name of persistence class
<i>return</i>	IPersistence store object that provides methods for persistence object handling

Exception

java.lang.Exception

# 1. Interface IRemotingConnection

Connection coming from Remoting clients.

## 1.1. Synopsis

```
public interface IRemotingConnection extends, org.red5.server.api.IConnection {  
    // Public Methods  
  
    public void addHeader(String name,  
                          Object value);  
  
    public void addHeader(String name,  
                          Object value,  
                          boolean mustUnderstand);  
  
    public java.util.Collection<org.red5.server.api.remoting.IRemotingHeader> getHeaders();  
  
    public void removeHeader(String name);  
  
}
```

## 1.2. addHeader(String, Object)

```
public void addHeader(String name,  
                      Object value);
```

Tell the client to add a header with all further requests. This is returned to the client as response for the next request received.

### Parameters

name	name of the header to add
value	value of the header to add

## 1.3. addHeader(String, Object, boolean)

```
public void addHeader(String name,  
                      Object value,  
                      boolean mustUnderstand);
```

Tell the client to add a header with all further requests. This is returned to the client as response for the next request received.

### Parameters

name	name of the header to add
value	value of the header to add
mustUnderstand	a boolean flag specifying if the server must process this header before handling following headers or messages

## 1.4. getHeaders()

```
public java.util.Collection<org.red5.server.api.remoting.IRemotingHeader> getHeaders();
```

Return headers to send.

### Parameters

<i>return</i>
---------------

## 1.5. removeHeader(String)

```
public void removeHeader(String name);
```

Tell the client to no longer send a header with all further requests. This is returned to the client as response for the next request received.

### Parameters

name	name of the header to remove
------	------------------------------

## 2. Interface IRemotingHeader

A Remoting header.

### 2.1. Synopsis

```
public interface IRemotingHeader {
// Public Static Fields

    public static final String APPEND_TO_GATEWAY_URL = "AppendToGatewayUrl";

    public static final String CREDENTIALS = "Credentials";

    public static final String DEBUG_SERVER = "amf_server_debug";

    public static final String PERSISTENT_HEADER = "RequestPersistentHeader";

    public static final String REPLACE_GATEWAY_URL = "ReplaceGatewayUrl";

// Public Methods

    public boolean getMustUnderstand();

    public String getName();

    public Object getValue();

}
```

### 2.2. APPEND\_TO\_GATEWAY\_URL

```
public static final String APPEND_TO_GATEWAY_URL = "AppendToGatewayUrl";
```

Name of header specifying string to add to gateway url.

## 2.3. CREDENTIALS

```
public static final String CREDENTIALS = "Credentials";
```

Name of header containing authentication data.

## 2.4. DEBUG\_SERVER

```
public static final String DEBUG_SERVER = "amf_server_debug";
```

Name of header to request debug informations from the server.

## 2.5. PERSISTENT\_HEADER

```
public static final String PERSISTENT_HEADER = "RequestPersistentHeader";
```

Name of header specifying new header to send.

## 2.6. REPLACE\_GATEWAY\_URL

```
public static final String REPLACE_GATEWAY_URL = "ReplaceGatewayUrl";
```

Name of header specifying new gateway url to use.

## 2.7. getMustUnderstand()

```
public boolean getMustUnderstand();
```

Return boolean flag if receiver must process header before handling other headers or messages.

### Parameters

<i>return</i>
---------------

## 2.8. getName()

```
public String getName();
```

Return name of header.

### Parameters

<i>return</i>
---------------

## 2.9. getValue()

```
public Object getValue();
```

Return value of header.

**Parameters**

*return*

# 1. Interface IScheduledJob

Interface that must be implemented by classes that can be scheduled for periodic execution.

## 1.1. Synopsis

```
public interface IScheduledJob {  
    // Public Methods  
  
    public void execute(ISchedulingService service)  
        throws CloneNotSupportedException;  
}
```

## 1.2. execute(ISchedulingService)

```
public void execute(ISchedulingService service)  
    throws CloneNotSupportedException;
```

Called each time the job is triggered by the scheduling service.

### Parameters

service	the service that called the job
---------	---------------------------------

# 2. Interface ISchedulingService

Service that supports periodic execution of jobs, adding, removing and getting their name as list.

## 2.1. Synopsis

```
public interface ISchedulingService extends org.red5.server.api.IScopeService {  
    // Public Static Fields  
  
    public static final String BEAN_NAME = "schedulingService";  
  
    // Public Methods  
  
    public String addScheduledJob(int interval,  
        IScheduledJob job);  
  
    public String addScheduledJobAfterDelay(int interval,  
        IScheduledJob job,  
        int delay);  
  
    public String addScheduledOnceJob(java.util.Date date,  
        IScheduledJob job);  
  
    public String addScheduledOnceJob(long timeDelta,  
        IScheduledJob job);
```

```

public java.util.List<java.lang.String> getScheduledJobNames();

public void removeScheduledJob(String name);

}

```

## 2.2. addScheduledJob(int, IScheduledJob)

```

public String addScheduledJob(int interval,
                             IScheduledJob job);

```

Schedule a job for periodic execution.

Parameters	
interval	time in milliseconds between two notifications of the job
job	the job to trigger periodically
<i>return</i>	the name of the scheduled job

## 2.3. addScheduledJobAfterDelay(int, IScheduledJob, int)

```

public String addScheduledJobAfterDelay(int interval,
                                       IScheduledJob job,
                                       int delay);

```

Schedule a job for periodic execution which will start after the specified delay.

Parameters	
interval	time in milliseconds between two notifications of the job
job	the job to trigger periodically
delay	time in milliseconds to pass before first execution.
<i>return</i>	the name of the scheduled job

## 2.4. addScheduledOnceJob(Date, IScheduledJob)

```

public String addScheduledOnceJob(java.util.Date date,
                                 IScheduledJob job);

```

Schedule a job for single execution at a given date. Please note that the jobs are not saved if Red5 is restarted in the meantime.

Parameters	
date	date when the job should be executed
job	the job to trigger
<i>return</i>	the name of the scheduled job

## 2.5. addScheduledOnceJob(long, IScheduledJob)

```
public String addScheduledOnceJob(long timeDelta,
                                  IScheduledJob job);
```

Schedule a job for single execution in the future. Please note that the jobs are not saved if Red5 is restarted in the meantime.

### Parameters

timeDelta	time delta in milliseconds from the current date
job	the job to trigger
<i>return</i>	the name of the scheduled job

## 2.6. getScheduledJobNames()

```
public java.util.List<java.lang.String> getScheduledJobNames();
```

Return names of scheduled jobs.

### Parameters

<i>return</i>	list of job names
---------------	-------------------

## 2.7. removeScheduledJob(String)

```
public void removeScheduledJob(String name);
```

Stop executing a previously scheduled job.

### Parameters

name	name of the job to stop
------	-------------------------

# 1. Interface IPendingServiceCall

IPendingServiceCall is a call that have a list of callbacks.

## 1.1. Synopsis

```
public interface IPendingServiceCall extends, org.red5.server.api.service.IServiceCall {  
    // Public Methods  
  
    public java.util.Set<org.red5.server.api.service.IPendingServiceCallback> getCallbacks();  
  
    public Object getResult();  
  
    public void registerCallback(IPendingServiceCallback callback);  
  
    public void setResult(Object result);  
  
    public void unregisterCallback(IPendingServiceCallback callback);  
}
```

## 1.2. getCallbacks()

```
public java.util.Set<org.red5.server.api.service.IPendingServiceCallback> getCallbacks();
```

Returns list of callback objects, usually callback object represented as an anonymous class instance that implements IPendingServiceCallback interface.

### Parameters

<i>return</i>	Set of pending operations callbacks
---------------	-------------------------------------

## 1.3. getResult()

```
public Object getResult();
```

Returns service call result

### Parameters

<i>return</i>	Remote call result
---------------	--------------------

## 1.4. registerCallback(IPendingServiceCallback)

```
public void registerCallback(IPendingServiceCallback callback);
```

Registers callback object usually represented as an anonymous class instance that implements IPendingServiceCallback interface.

### Parameters

callback

Callback object

## 1.5. setResult(Object)

```
public void setResult(Object result);
```

Setter for property 'result'.

Parameters

result

Value to set for property 'result'.

## 1.6. unregisterCallback(IPendingServiceCallback)

```
public void unregisterCallback(IPendingServiceCallback callback);
```

Unregisters callback object usually represented as an anonymous class instance that implements IPendingServiceCallback interface.

Parameters

callback

Callback object

# 2. Interface IPendingServiceCallback

Callback that will be executed when the result of a pending service call has been received.

## 2.1. Synopsis

```
public interface IPendingServiceCallback {
    // Public Methods

    public void resultReceived(IPendingServiceCall call);

}
```

## 2.2. resultReceived(IPendingServiceCall)

```
public void resultReceived(IPendingServiceCall call);
```

Triggered when results are received

Parameters

call

Call object this callback is applied to

# 3. Interface IServiceCall

Container for a Service Call

### 3.1. Synopsis

```
public interface IServiceCall {
// Public Methods

    public Object[] getArguments();

    public Exception getException();

    public String getServiceMethodName();

    public String getServiceName();

    public byte getStatus();

    public boolean isSuccess();

    public void setException(Exception exception);

    public void setStatus(byte status);

}
```

### 3.2. getArguments()

public Object[] getArguments();
---------------------------------

Returns array of service method arguments

#### Parameters

<i>return</i>	array of service method arguments
---------------	-----------------------------------

### 3.3. getException()

public Exception getException();
----------------------------------

Get service call exception

#### Parameters

<i>return</i>	service call exception
---------------	------------------------

### 3.4. getServiceMethodName()

public String getServiceMethodName();
---------------------------------------

Returns service method name

#### Parameters

<i>return</i>	Service method name as string
---------------	-------------------------------

### 3.5. getServiceName()

```
public String getServiceName();
```

Returns service name

#### Parameters

<i>return</i>	Service name
---------------	--------------

### 3.6. getStatus()

```
public byte getStatus();
```

Get service call status

#### Parameters

<i>return</i>	service call status
---------------	---------------------

### 3.7. isSuccess()

```
public boolean isSuccess();
```

Whether call was successful or not

#### Parameters

<i>return</i>	true on success, false otherwise
---------------	----------------------------------

### 3.8. setException(Exception)

```
public void setException(Exception exception);
```

Sets exception

#### Parameters

<i>exception</i>	Call exception
------------------	----------------

### 3.9. setStatus(byte)

```
public void setStatus(byte status);
```

Sets status

#### Parameters

<i>status</i>	Status as byte
---------------	----------------

## 4. Interface IServiceCapableConnection

Connection that has options to invoke and handle remote calls

## 4.1. Synopsis

```

public interface IServiceCapableConnection extends, org.?red5.?server.?api.?IConnection {
    // Public Methods

    public void invoke(String method);

    public void invoke(String method,
                      Object[] params);

    public void invoke(String method,
                      Object[] params,
                      IPendingServiceCallback callback);

    public void invoke(String method,
                      IPendingServiceCallback callback);

    public void invoke(IServiceCall call);

    public void invoke(IServiceCall call,
                      int channel);

    public void notify(String method);

    public void notify(String method,
                      Object[] params);

    public void notify(IServiceCall call);

    public void notify(IServiceCall call,
                      int channel);
}

```

## 4.2. invoke(IServiceCall)

```
public void invoke(IServiceCall call);
```

Invokes service using remoting call object

### Parameters

call	Service call object
------	---------------------

## 4.3. invoke(IServiceCall, int)

```
public void invoke(IServiceCall call,
                  int channel);
```

Invoke service using call and channel

### Parameters

call	Service call
------	--------------

channel	Channel used
---------	--------------

#### 4.4. invoke(String)

```
public void invoke(String method);
```

Invoke method by name

##### Parameters

method	Called method name
--------	--------------------

#### 4.5. invoke(String, IPendingServiceCallback)

```
public void invoke(String method,
                   IPendingServiceCallback callback);
```

Invoke method by name with callback

##### Parameters

method	Called method name
callback	Callback

#### 4.6. invoke(String, Object[])

```
public void invoke(String method,
                   Object[] params);
```

Invoke method with parameters

##### Parameters

method	Method name
params	Invocation parameters passed to method

#### 4.7. invoke(String, Object[], IPendingServiceCallback)

```
public void invoke(String method,
                   Object[] params,
                   IPendingServiceCallback callback);
```

##### Parameters

method	
params	
callback	

#### 4.8. notify(IServiceCall)

```
public void notify(IServiceCall call);
```

**Parameters**

call

**4.9. notify(IServiceCall, int)**

```
public void notify(IServiceCall call,
                   int channel);
```

**Parameters**

call

channel

**4.10. notify(String)**

```
public void notify(String method);
```

**Parameters**

method

**4.11. notify(String, Object[])**

```
public void notify(String method,
                   Object[] params);
```

**Parameters**

method

params

**5. Interface IServiceHandlerProvider**

Supports registration and lookup of service handlers.

**5.1. Synopsis**

```
public interface IServiceHandlerProvider {
    // Public Methods

    public Object getServiceHandler(String name);

    public java.util.Set<java.lang.String> getServiceHandlerNames();

    public void registerServiceHandler(String name,
                                      Object handler);

    public void unregisterServiceHandler(String name);

}
```

## 5.2. getServiceHandler(String)

```
public Object getServiceHandler(String name);
```

Return a previously registered service handler.

### Parameters

<i>name</i>	the name of the handler to return
<i>return</i>	the previously registered handler

## 5.3. getServiceHandlerNames()

```
public java.util.Set<java.lang.String> getServiceHandlerNames();
```

Get list of registered service handler names.

### Parameters

<i>return</i>	the names of the registered handlers
---------------	--------------------------------------

## 5.4. registerServiceHandler(String, Object)

```
public void registerServiceHandler(String name,
                                   Object handler);
```

Register an object that provides methods which can be called from a client.

Example:

If you registered a handler with the name "one.two" that provides a method "callMe", you can call a method "one.two.callMe" from the client.

### Parameters

<i>name</i>	the name of the handler
<i>handler</i>	the handler object

## 5.5. unregisterServiceHandler(String)

```
public void unregisterServiceHandler(String name);
```

Unregister service handler.

### Parameters

<i>name</i>	the name of the handler
-------------	-------------------------

## 6. Interface IServerProviderAware

Class that knows about objects which can provide service handlers.

## 6.1. Synopsis

```
public interface IServiceHandlerProviderAware {
    // Public Methods

    public IServiceHandlerProvider getServiceHandlerProvider();

}
```

## 6.2. getServiceHandlerProvider()

```
public IServiceHandlerProvider getServiceHandlerProvider();
```

Return object that knows about service handlers.

### Parameters

<i>return</i>
---------------

## 7. Interface IServicInvoker

Interface for objects that execute service calls (remote calls from client).

### 7.1. Synopsis

```
public interface IServicInvoker {
    // Public Methods

    public boolean invoke(IServiceCall call,
                          Object service);

    public boolean invoke(IServiceCall call,
                          org.red5.server.api.IScope scope);

}
```

## 7.2. invoke(IServiceCall, IScope)

```
public boolean invoke(IServiceCall call,
                      org.red5.server.api.IScope scope);
```

Execute the passed service call in the given scope. This looks up the handler for the call in the scope and the context of the scope.

### Parameters

call	the call to invoke
scope	the scope to search for a handler
<i>return</i>	true if the call was performed, otherwise false

## 7.3. invoke(IServiceCall, Object)

```
public boolean invoke(IServiceCall call,
                      Object service);
```

Execute the passed service call in the given object.

Parameters	
call	the call to invoke
service	the service to use
return	true if the call was performed, otherwise false

## 8. Class ServiceUtils

Utility functions to invoke methods on connections.

### 8.1. Synopsis

```
public class ServiceUtils {
    // Public Constructors

    public ServiceUtils();

    // Public Static Methods

    public static void invokeOnAllConnections(String method,
                                              Object[] params);

    public static void invokeOnAllConnections(String method,
                                              Object[] params,
                                              IPendingServiceCallback callback);

    public static void invokeOnAllConnections(org.red5.server.api.IScope scope,
                                              String method,
                                              Object[] params);

    public static void invokeOnAllConnections(org.red5.server.api.IScope scope,
                                              String method,
                                              Object[] params,
                                              IPendingServiceCallback callback);

    public static void invokeOnClient(org.red5.server.api.IClient client,
                                     org.red5.server.api.IScope scope,
                                     String method,
                                     Object[] params);

    public static void invokeOnClient(org.red5.server.api.IClient client,
                                     org.red5.server.api.IScope scope,
                                     String method,
                                     Object[] params,
                                     IPendingServiceCallback callback);

    public static boolean invokeOnConnection(String method,
                                            Object[] params);

    public static boolean invokeOnConnection(String method,
```

```

        Object[] params,
        IPendingServiceCallback callback);

public static boolean invokeOnConnection(org.red5.server.api.IConnection conn,
                                         String method,
                                         Object[] params);

public static boolean invokeOnConnection(org.red5.server.api.IConnection conn,
                                         String method,
                                         Object[] params,
                                         IPendingServiceCallback callback);

public static void notifyOnAllConnections(String method,
                                         Object[] params);

public static void notifyOnAllConnections(org.red5.server.api.IScope scope,
                                         String method,
                                         Object[] params);

public static void notifyOnClient(org.red5.server.api.IClient client,
                                 org.red5.server.api.IScope scope,
                                 String method,
                                 Object[] params);

public static boolean notifyOnConnection(String method,
                                         Object[] params);

public static boolean notifyOnConnection(org.red5.server.api.IConnection conn,
                                         String method,
                                         Object[] params);
}

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 8.2. invokeOnAllConnections(IScope, String, Object[])

```

public static void invokeOnAllConnections(org.red5.server.api.IScope scope,
                                         String method,
                                         Object[] params);

```

Invoke a method on all connections to a given scope.

### Parameters

scope	scope to get connections for
method	name of the method to invoke
params	parameters to pass to the method

## 8.3. invokeOnAllConnections(IScope, String, Object[], IPendingServiceCallback)

```

public static void invokeOnAllConnections(org.red5.server.api.IScope scope,
                                         String method,
                                         Object[] params,
                                         IPendingServiceCallback callback);

```

```
Object[] params,
IPendingServiceCallback callback);
```

Invoke a method on all connections to a given scope and handle result.

#### Parameters

scope	scope to get connections for
method	name of the method to invoke
params	parameters to pass to the method
callback	object to notify when result is received

### 8.4. invokeOnAllConnections(String, Object[])

```
public static void invokeOnAllConnections(String method,
                                         Object[] params);
```

Invoke a method on all connections to the current scope.

#### Parameters

method	name of the method to invoke
params	parameters to pass to the method

### 8.5. invokeOnAllConnections(String, Object[], IPendingServiceCallback)

```
public static void invokeOnAllConnections(String method,
                                         Object[] params,
                                         IPendingServiceCallback callback);
```

Invoke a method on all connections to the current scope and handle result.

#### Parameters

method	name of the method to invoke
params	parameters to pass to the method
callback	object to notify when result is received

### 8.6. invokeOnClient(IClient, IScope, String, Object[])

```
public static void invokeOnClient(org.red5.server.api.IClient client,
                                 org.red5.server.api.IScope scope,
                                 String method,
                                 Object[] params);
```

Invoke a method on all connections of a client to a given scope.

#### Parameters

client	client to get connections for
scope	scope to get connections of the client from

method	name of the method to invoke
params	parameters to pass to the method

## 8.7. invokeOnClient(IClient, IScope, String, Object[], IPendingServiceCallback)

```
public static void invokeOnClient(org.red5.server.api.IClient client,
                                org.red5.server.api.IScope scope,
                                String method,
                                Object[] params,
                                IPendingServiceCallback callback);
```

Invoke a method on all connections of a client to a given scope and handle result.

Parameters	
client	client to get connections for
scope	scope to get connections of the client from
method	name of the method to invoke
params	parameters to pass to the method
callback	object to notify when result is received

## 8.8. invokeOnConnection(IConnection, String, Object[])

```
public static boolean invokeOnConnection(org.red5.server.api.IConnection conn,
                                         String method,
                                         Object[] params);
```

Invoke a method on a given connection.

Parameters	
conn	connection to invoke method on
method	name of the method to invoke
params	parameters to pass to the method
return	true if the connection supports method calls, otherwise false

## 8.9. invokeOnConnection(IConnection, String, Object[], IPendingServiceCallback)

```
public static boolean invokeOnConnection(org.red5.server.api.IConnection conn,
                                         String method,
                                         Object[] params,
                                         IPendingServiceCallback callback);
```

Invoke a method on a given connection and handle result.

Parameters	

conn	connection to invoke method on
method	name of the method to invoke
params	parameters to pass to the method
callback	object to notify when result is received
return	true if the connection supports method calls, otherwise false

## 8.10. invokeOnConnection(String, Object[])

```
public static boolean invokeOnConnection(String method,
                                         Object[] params);
```

Invoke a method on the current connection.

Parameters	
method	name of the method to invoke
params	parameters to pass to the method
return	true if the connection supports method calls, otherwise false

## 8.11. invokeOnConnection(String, Object[], IPendingServiceCallback)

```
public static boolean invokeOnConnection(String method,
                                         Object[] params,
                                         IPendingServiceCallback callback);
```

Invoke a method on the current connection and handle result.

Parameters	
method	name of the method to invoke
params	parameters to pass to the method
callback	object to notify when result is received
return	true if the connection supports method calls, otherwise false

## 8.12. notifyOnAllConnections(IScope, String, Object[])

```
public static void notifyOnAllConnections(org.red5.server.api.IScope scope,
                                         String method,
                                         Object[] params);
```

Notify a method on all connections to a given scope.

Parameters	
scope	scope to get connections for
method	name of the method to notify
params	parameters to pass to the method

## 8.13. notifyOnAllConnections(String, Object[])

```
public static void notifyOnAllConnections(String method,
                                         Object[] params);
```

Notify a method on all connections to the current scope.

### Parameters

method	name of the method to notify
params	parameters to pass to the method

## 8.14. notifyOnClient(IClient, IScope, String, Object[])

```
public static void notifyOnClient(org.red5.server.api.IClient client,
                                 org.red5.server.api.IScope scope,
                                 String method,
                                 Object[] params);
```

Notify a method on all connections of a client to a given scope.

### Parameters

client	client to get connections for
scope	scope to get connections of the client from
method	name of the method to notify
params	parameters to pass to the method

## 8.15. notifyOnConnection(IConnection, String, Object[])

```
public static boolean notifyOnConnection(org.red5.server.api.IConnection conn,
                                       String method,
                                       Object[] params);
```

Notify a method on a given connection.

### Parameters

conn	connection to notify method on
method	name of the method to notify
params	parameters to pass to the method
<i>return</i>	<code>true</code> if the connection supports method calls, otherwise <code>false</code>

## 8.16. notifyOnConnection(String, Object[])

```
public static boolean notifyOnConnection(String method,
                                       Object[] params);
```

Notify a method on the current connection.

Parameters	
method	name of the method to notify
params	parameters to pass to the method
<i>return</i>	<code>true</code> if the connection supports method calls, otherwise <code>false</code>

# 1. Interface IClientSharedObject

Clientside access to shared objects.

## 1.1. Synopsis

```
public interface IClientSharedObject extends, org.red5.server.api.ISharedObjectBase {  
    // Public Methods  
  
    public void connect(org.red5.server.api.IConnection conn);  
  
    public void disconnect();  
  
    public boolean isConnected();  
}
```

## 1.2. connect(IConnection)

```
public void connect(org.red5.server.api.IConnection conn);
```

Connect the shared object using the passed connection.

### Parameters

conn

## 1.3. disconnect()

```
public void disconnect();
```

Disconnect the shared object.

## 1.4. isConnected()

```
public boolean isConnected();
```

Check if the shared object is connected to the server.

### Parameters

return

# 2. Interface ISharedObject

Serverside access to shared objects.

## 2.1. Synopsis

```
public interface ISharedObject extends, org.red5.server.api.IBasicScope, org.red5.server.api.I...
```

```
// Public Static Fields

    public static final String TYPE = "SharedObject";

// Public Methods

    public void acquire();

    public org.red5.server.api.statistics.ISharedObjectStatistics getStatistics();

    public boolean isAcquired();

    public void release();

}
```

## 2.2. acquire()

```
public void acquire();
```

Prevent shared object from being released. Each call to `acquire` must be paired with a call to `release` so the SO isn't held forever. This method basically is a noop for persistent SOs as their data is stored and they can be released without losing their contents.

## 2.3. getStatistics()

```
public org.red5.server.api.statistics.ISharedObjectStatistics getStatistics();
```

Return statistics about the shared object.

### Parameters

<i>return</i>	statistics
---------------	------------

## 2.4. isAcquired()

```
public boolean isAcquired();
```

Check if shared object currently is acquired.

### Parameters

<i>return</i>	true if the SO is acquired, otherwise false
---------------	---

## 2.5. release()

```
public void release();
```

Release previously acquired shared object. If the SO is non-persistent, no more clients are connected the SO isn't acquired any more, the data is released.

## 3. Interface ISharedObjectBase

Base interface for shared objects. Changes to the shared objects are propagated to all subscribed clients. If you want to modify multiple attributes and notify the clients about all changes at once, you can use code like this:

```
SharedObject.beginUpdate(); SharedObject.setAttribute( "One" , '1' );
SharedObject.setAttribute( "Two" , '2' ); SharedObject.removeAttribute( "Three" );
SharedObject.endUpdate();
```

All changes between "beginUpdate" and "endUpdate" will be sent to the clients using one notification event.

### 3.1. Synopsis

```
public interface ISharedObjectBase extends, org.?red5.?server.?api.?so.?ISharedObjectHandlerProvider,
// Public Methods

    public void addSharedObjectListener(ISharedObjectListener listener);

    public void beginUpdate();

    public void beginUpdate(org.red5.server.api.event.IEventListener source);

    public boolean clear();

    public void close();

    public void endUpdate();

    public java.util.Map<java.lang.String, java.lang.Object> getData();

    public int getVersion();

    public boolean isLocked();

    public boolean isPersistentObject();

    public void lock();

    public void removeSharedObjectListener(ISharedObjectListener listener);

    public void sendMessage(String handler,
                          java.util.List arguments);

    public void unlock();

}
```

### 3.2. addSharedObjectListener(ISharedObjectListener)

```
public void addSharedObjectListener(ISharedObjectListener listener);
```

Register object that will be notified about update events.

#### Parameters

listener	the object to notify
----------	----------------------

### 3.3. beginUpdate()

```
public void beginUpdate();
```

Start performing multiple updates to the shared object from serverside code.

### 3.4. beginUpdate(IEventListener)

```
public void beginUpdate(org.red5.server.api.event.IEventListener source);
```

Start performing multiple updates to the shared object from a connected client.

#### Parameters

source	Update events listener
--------	------------------------

### 3.5. clear()

```
public boolean clear();
```

Deletes all the attributes and sends a clear event to all listeners. The persistent data object is also removed from a persistent shared object.

#### Parameters

return	true if successful; false otherwise
--------	-------------------------------------

### 3.6. close()

```
public void close();
```

Detaches a reference from this shared object, this will destroy the reference immediately. This is useful when you don't want to proxy a shared object any longer.

### 3.7. endUpdate()

```
public void endUpdate();
```

The multiple updates are complete, notify clients about all changes at once.

### 3.8. getData()

```
public java.util.Map<java.lang.String, java.lang.Object> getData();
```

Return a map containing all attributes of the shared object.

NOTE: The returned map will be read-only.

#### Parameters

**return**

a map containing all attributes of the shared object

### 3.9. getVersion()

```
public int getVersion();
```

Returns the version of the shared object. The version is incremented automatically on each modification.

**Parameters**

**return**

the version of the shared object

### 3.10. isLocked()

```
public boolean isLocked();
```

Returns the locked state of this SharedObject.

**Parameters**

**return**

true if in a locked state; false otherwise

### 3.11. isPersistentObject()

```
public boolean isPersistentObject();
```

Check if the object has been created as persistent shared object by the client.

**Parameters**

**return**

true if the shared object is persistent, false otherwise

### 3.12. lock()

```
public void lock();
```

Locks the shared object instance. Prevents any changes to this object by clients until the SharedObject.unlock() method is called.

### 3.13. removeSharedObjectListener(ISharedObjectListener)

```
public void removeSharedObjectListener(ISharedObjectListener listener);
```

Unregister object to not longer receive update events.

**Parameters**

**listener**

the object to unregister

### 3.14. sendMessage(String, List)

```
public void sendMessage(String handler,
```

```
java.util.List arguments);
```

Send a message to a handler of the shared object.

#### Parameters

handler	the name of the handler to call
arguments	a list of objects that should be passed as arguments to the handler

### 3.15. unlock()

```
public void unlock();
```

Unlocks a shared object instance that was locked with SharedObject.lock().

## 4. Interface ISharedObjectHandlerProvider

Supports registration and lookup of shared object handlers.

### 4.1. Synopsis

```
public interface ISharedObjectHandlerProvider extends, org.?red5.?server.?api.?service.?IServiceHandlerProvider
// Public Methods

    public void registerServiceHandler(Object handler);

    public void unregisterServiceHandler(String name);

}
```

### 4.2. registerServiceHandler(Object)

```
public void registerServiceHandler(Object handler);
```

Register an object that provides methods which handle calls without a service name to a shared object.

#### Parameters

handler	the handler object
---------	--------------------

### 4.3. unregisterServiceHandler(String)

```
public void unregisterServiceHandler(String name);
```

**Specified by:** Method unregisterServiceHandler in interface IServiceHandlerProvider

Unregister the shared object handler for calls without a service name.

## 5. Interface ISharedObjectListener

Notifications about shared object updates.

## 5.1. Synopsis

```
public interface ISharedObjectListener {
// Public Methods

    public void onSharedObjectClear(ISharedObjectBase so);

    public void onSharedObjectConnect(ISharedObjectBase so);

    public void onSharedObjectDelete(ISharedObjectBase so,
                                    String key);

    public void onSharedObjectDisconnect(ISharedObjectBase so);

    public void onSharedObjectSend(ISharedObjectBase so,
                                String method,
                                java.util.List params);

    public void onSharedObjectUpdate(ISharedObjectBase so,
                                    String key,
                                    Object value);

    public void onSharedObjectUpdate(ISharedObjectBase so,
                                    java.util.Map<java.lang.String, java.lang.Object> values);

    public void onSharedObjectUpdate(ISharedObjectBase so,
                                    org.red5.server.api.IAttributeStore values);
}

}
```

## 5.2. onSharedObjectClear(ISharedObjectBase)

```
public void onSharedObjectClear(ISharedObjectBase so);
```

Called when all attributes of a shared object are removed.

### Parameters

so	the shared object
----	-------------------

## 5.3. onSharedObjectConnect(ISharedObjectBase)

```
public void onSharedObjectConnect(ISharedObjectBase so);
```

Called when a client connects to a shared object.

### Parameters

so	the shared object
----	-------------------

## 5.4. onSharedObjectDelete(ISharedObjectBase, String)

```
public void onSharedObjectDelete(ISharedObjectBase so,
```

```
String key);
```

Called when an attribute is deleted from the shared object.

#### Parameters

so	the shared object
key	the name of the attribute to delete

### 5.5. onSharedObjectDisconnect(ISharedObjectBase)

```
public void onSharedObjectDisconnect(ISharedObjectBase so);
```

Called when a client disconnects from a shared object.

#### Parameters

so	the shared object
----	-------------------

### 5.6. onSharedObjectSend(ISharedObjectBase, String, List)

```
public void onSharedObjectSend(ISharedObjectBase so,
                               String method,
                               java.util.List params);
```

Called when a shared object method call is sent.

#### Parameters

so	the shared object
method	the method name to call
params	the arguments

### 5.7. onSharedObjectUpdate(ISharedObjectBase, IAttributeStore)

```
public void onSharedObjectUpdate(ISharedObjectBase so,
                                 org.red5.server.api.IAttributeStore values);
```

Called when multiple attributes of a shared object are updated.

#### Parameters

so	the shared object
values	the new attributes of the shared object

### 5.8. onSharedObjectUpdate(ISharedObjectBase, Map<String, Object>)

```
public void onSharedObjectUpdate(ISharedObjectBase so,
                                 java.util.Map<java.lang.String, java.lang.Object> values);
```

Called when multiple attributes of a shared object are updated.

**Parameters**

so	the shared object
values	the new attributes of the shared object

**5.9. onSharedObjectUpdate(ISharedObjectBase, String, Object)**

```
public void onSharedObjectUpdate(ISharedObjectBase so,
                                String key,
                                Object value);
```

Called when a shared object attribute is updated.

**Parameters**

so	the shared object
key	the name of the attribute
value	the value of the attribute

**6. Interface ISharedObjectSecurity**

Interface for handlers that control access to shared objects.

**6.1. Synopsis**

```
public interface ISharedObjectSecurity {
    // Public Methods

    public boolean isConnectionAllowed(ISharedObject so);

    public boolean isCreationAllowed(org.red5.server.api.IScope scope,
                                    String name,
                                    boolean persistent);

    public boolean isDeleteAllowed(ISharedObject so,
                                 String key);

    public boolean isSendAllowed(ISharedObject so,
                               String message,
                               java.util.List arguments);

    public boolean isWriteAllowed(ISharedObject so,
                             String key,
                             Object value);
}
```

**6.2. isConnectionAllowed(ISharedObject)**

```
public boolean isConnectionAllowed(ISharedObject so);
```

Check if a connection to the given existing shared object is allowed.

**Parameters**

so

return

**6.3. isCreationAllowed(IScope, String, boolean)**

```
public boolean isCreationAllowed(org.red5.server.api.IScope scope,
                                String name,
                                boolean persistent);
```

Check if the a shared object may be created in the given scope.

**Parameters**

scope

name

persistent

return

**6.4. isDeleteAllowed(ISharedObject, String)**

```
public boolean isDeleteAllowed(ISharedObject so,
                               String key);
```

Check if the deletion of a property is allowed on the given shared object.

**Parameters**

so

key

return

**6.5. isSendAllowed(ISharedObject, String, List)**

```
public boolean isSendAllowed(ISharedObject so,
                            String message,
                            java.util.List arguments);
```

Check if sending a message to the shared object is allowed.

**Parameters**

so

message

arguments

return

## 6.6. isWriteAllowed(ISharedObject, String, Object)

```
public boolean isWriteAllowed(ISharedObject so,
                             String key,
                             Object value);
```

Check if a modification is allowed on the given shared object.

Parameters	
so	
key	
value	
<i>return</i>	

## 7. Interface ISharedObjectSecurityService

Service that supports protecting access to shared objects.

### 7.1. Synopsis

```
public interface ISharedObjectSecurityService extends, org.?red5.?server.?api.?IScopeService {
    // Public Static Fields

    public static final String BEAN_NAME = "sharedObjectSecurityService";

    // Public Methods

    public java.util.Set<org.red5.server.api.so.ISharedObjectSecurity> getSharedObjectSecurity();

    public void registerSharedObjectSecurity(ISharedObjectSecurity handler);

    public void unregisterSharedObjectSecurity(ISharedObjectSecurity handler);

}
```

### 7.2. BEAN\_NAME

```
public static final String BEAN_NAME = "sharedObjectSecurityService";
```

Name of a bean defining that scope service.

### 7.3. getSharedObjectSecurity()

```
public java.util.Set<org.red5.server.api.so.ISharedObjectSecurity> getSharedObjectSecurity();
```

Get handlers that protect shared objects.

Parameters	
<i>return</i>	list of handlers

## 7.4. registerSharedObjectSecurity(ISharedObjectSecurity)

```
public void registerSharedObjectSecurity(ISharedObjectSecurity handler);
```

Add handler that protects shared objects.

### Parameters

handler	Handler to add.
---------	-----------------

## 7.5. unregisterSharedObjectSecurity(ISharedObjectSecurity)

```
public void unregisterSharedObjectSecurity(ISharedObjectSecurity handler);
```

Remove handler that protects shared objects.

### Parameters

handler	Handler to remove.
---------	--------------------

# 8. Interface ISharedObjectService

Service that manages shared objects for given scope.

## 8.1. Synopsis

```
public interface ISharedObjectService extends org.?red5.?server.?api.?IScopeService {
    // Public Static Fields

    public static final String BEAN_NAME = "sharedObjectService";

    // Public Methods

    public boolean clearSharedObjects(org.red5.server.api.IScope scope,
                                    String name);

    public boolean createSharedObject(org.red5.server.api.IScope scope,
                                    String name,
                                    boolean persistent);

    public ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                       String name);

    public ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                       String name,
                                       boolean persistent);

    public java.util.Set<java.lang.String> getSharedObjectNames(org.red5.server.api.IScope scope);

    public boolean hasSharedObject(org.red5.server.api.IScope scope,
                                 String name);

}
```

## 8.2. clearSharedObjects(IScope, String)

```
public boolean clearSharedObjects(org.red5.server.api.IScope scope,
                                String name);
```

Deletes persistent shared objects specified by name and clears all properties from active shared objects (persistent and nonpersistent). The name parameter specifies the name of a shared object, which can include a slash (/) as a delimiter between directories in the path. The last element in the path can contain wildcard patterns (for example, a question mark [?] and an asterisk [\*]) or a shared object name. The clearSharedObjects() method traverses the shared object hierarchy along the specified path and clears all the shared objects. Specifying a slash (/) clears all the shared objects associated with an application instance.

The following values are possible for the soPath parameter:

/ clears all local and persistent shared objects associated with the instance.

/foo/bar clears the shared object /foo/bar; if bar is a directory name, no shared objects are deleted.

/foo/bar/\* clears all shared objects stored under the instance directory /foo/bar. The bar directory is also deleted if no persistent shared objects are in use within this namespace.

/foo/bar/XX?? clears all shared objects that begin with XX, followed by any two characters. If a directory name matches this specification, all the shared objects within this directory are cleared.

If you call the clearSharedObjects() method and the specified path matches a shared object that is currently active, all its properties are deleted, and a "clear" event is sent to all subscribers of the shared object. If it is a persistent shared object, the persistent store is also cleared.

### Parameters

scope	the scope to check for the shared object
name	the name of the shared object
<i>return</i>	true if the shared object at the specified path was deleted; otherwise, false. If using wildcard characters to delete multiple files, the method returns true only if all the shared objects matching the wildcard pattern were successfully deleted; otherwise, it will return false.

## 8.3. createSharedObject(IScope, String, boolean)

```
public boolean createSharedObject(org.red5.server.api.IScope scope,
                                String name,
                                boolean persistent);
```

Create a new shared object.

### Parameters

scope	the scope to create the shared object in
name	the name of the shared object
persistent	will the shared object be persistent
<i>return</i>	<code>true</code> if the shared object was created, otherwise <code>false</code>

## 8.4. **getSharedObject(IScope, String)**

```
public ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                    String name);
```

Get a shared object by name.

Parameters	
scope	the scope to get the shared object from
name	the name of the shared object
<i>return</i>	shared object, or <code>null</code> if not found

## 8.5. **getSharedObject(IScope, String, boolean)**

```
public ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                    String name,
                                    boolean persistent);
```

Get a shared object by name and create it if it doesn't exist.

Parameters	
scope	the scope to get the shared object from
name	the name of the shared object
persistent	should the shared object be created persistent
<i>return</i>	the shared object

## 8.6. **getSharedObjectNames(IScope)**

```
public java.util.Set<java.lang.String> getSharedObjectNames(org.red5.server.api.IScope scope);
```

Get a set of the shared object names.

Parameters	
scope	the scope to return the shared object names from
<i>return</i>	set containing the shared object names

## 8.7. **hasSharedObject(IScope, String)**

```
public boolean hasSharedObject(org.red5.server.api.IScope scope,
```

```
String name);
```

Check if a shared object exists.

#### Parameters

scope	the scope to check for the shared object
name	the name of the shared object
<i>return</i>	<code>true</code> if the shared object exists, otherwise <code>false</code>

# 1. Interface IClientBroadcastStreamStatistics

Statistical informations about a stream that is broadcasted by a client.

## 1.1. Synopsis

```
public interface IClientBroadcastStreamStatistics extends, org.red5.server.api.statistics.IStreamStatistics
// Public Methods

    public int getActiveSubscribers();

    public long getBytesReceived();

    public int getMaxSubscribers();

    public String getPublishedName();

    public String getSaveFilename();

    public int getTotalSubscribers();

}
```

## 1.2. getActiveSubscribers()

```
public int getActiveSubscribers();
```

Return current number of subscribers.

### Parameters

<i>return</i>	number of subscribers
---------------	-----------------------

## 1.3. getBytesReceived()

```
public long getBytesReceived();
```

Return total number of bytes received from client for this stream.

### Parameters

<i>return</i>	number of bytes
---------------	-----------------

## 1.4. getMaxSubscribers()

```
public int getMaxSubscribers();
```

Return maximum number of concurrent subscribers.

### Parameters

<i>return</i>	number of subscribers
---------------	-----------------------

## 1.5. getPublishedName()

```
public String getPublishedName();
```

Get stream publish name. Publish name is the value of the first parameter had been passed to `NetStream.publish` on client side in SWF.

### Parameters

<i>return</i>	Stream publish name
---------------	---------------------

## 1.6. getSaveFilename()

```
public String getSaveFilename();
```

Get the filename the stream is being saved as.

### Parameters

<i>return</i>	The filename relative to the scope or <code>null</code> if the stream is not being saved.
---------------	---

## 1.7. getTotalSubscribers()

```
public int getTotalSubscribers();
```

Return total number of subscribers.

### Parameters

<i>return</i>	number of subscribers
---------------	-----------------------

## 2. Interface IPlaylistSubscriberStreamStatistics

Statistical informations about a stream that is subscribed by a client.

### 2.1. Synopsis

```
public interface IPlaylistSubscriberStreamStatistics extends, org.?red5.?server.?api.?statistics.?IST
// Public Methods

    public long getBytesSent();

    public int getClientBufferDuration();

    public double getEstimatedBufferFill();

}
```

## 2.2. getBytesSent()

```
public long getBytesSent();
```

Return total number of bytes sent to the client from this stream.

### Parameters

<i>return</i>	number of bytes
---------------	-----------------

## 2.3. getClientBufferDuration()

```
public int getClientBufferDuration();
```

Return the buffer duration as requested by the client.

### Parameters

<i>return</i>	the buffer duration in milliseconds
---------------	-------------------------------------

## 2.4. getEstimatedBufferFill()

```
public double getEstimatedBufferFill();
```

Return estimated fill ratio of the client buffer.

### Parameters

<i>return</i>	fill ratio in percent
---------------	-----------------------

# 3. Interface IScopeStatistics

Statistical informations about a scope.

## 3.1. Synopsis

```
public interface IScopeStatistics extends, org.?red5.?server.?api.?statistics.?IStatisticsBase {
    // Public Methods

    public int getActiveClients();

    public int getActiveConnections();

    public int getActiveSubscopes();

    public int getDepth();

    public int getMaxClients();

    public int getMaxConnections();

    public int getMaxSubscopes();

    public String getName();
```

```

    public String getPath();

    public int getTotalClients();

    public int getTotalConnections();

    public int getTotalSubscopes();

}

```

### 3.2. **getActiveClients()**

```
public int getActiveClients();
```

Return current number of clients connected to the scope.

#### Parameters

<i>return</i>	number of clients
---------------	-------------------

### 3.3. **getActiveConnections()**

```
public int getActiveConnections();
```

Return current number of connections to the scope.

#### Parameters

<i>return</i>	number of connections
---------------	-----------------------

### 3.4. **getActiveSubscopes()**

```
public int getActiveSubscopes();
```

Return number of currently existing subscopes.

#### Parameters

<i>return</i>	number of subscopes
---------------	---------------------

### 3.5. **getDepth()**

```
public int getDepth();
```

Get the scopes depth, how far down the scope tree is it. The lowest depth is 0x00, the depth of Global scope. Application scope depth is 0x01. Room depth is 0x02, 0x03 and so forth.

#### Parameters

<i>return</i>	the depth
---------------	-----------

### 3.6. getMaxClients()

```
public int getMaxClients();
```

Return maximum number of clients concurrently connected to the scope.

#### Parameters

<i>return</i>	number of clients
---------------	-------------------

### 3.7. getMaxConnections()

```
public int getMaxConnections();
```

Return maximum number of concurrent connections to the scope.

#### Parameters

<i>return</i>	number of connections
---------------	-----------------------

### 3.8. getMaxSubscopes()

```
public int getMaxSubscopes();
```

Return maximum number of concurrently existing subscopes.

#### Parameters

<i>return</i>	number of subscopes
---------------	---------------------

### 3.9. getName()

```
public String getName();
```

Get the name of this scope. Eg. someroom.

#### Parameters

<i>return</i>	the name
---------------	----------

### 3.10. getPath()

```
public String getPath();
```

Get the full absolute path. Eg. host/myapp/someroom.

#### Parameters

<i>return</i>	Absolute scope path
---------------	---------------------

### 3.11. getTotalClients()

```
public int getTotalClients();
```

Return total number of clients connected to the scope.

#### Parameters

<i>return</i>	number of clients
---------------	-------------------

### 3.12. getTotalConnections()

```
public int getTotalConnections();
```

Return total number of connections to the scope.

#### Parameters

<i>return</i>	number of connections
---------------	-----------------------

### 3.13. getTotalSubscopes()

```
public int getTotalSubscopes();
```

Return total number of subscopes created.

#### Parameters

<i>return</i>	number of subscopes created
---------------	-----------------------------

## 4. Interface ISharedObjectStatistics

Statistics informations about a shared object.

### 4.1. Synopsis

```
public interface ISharedObjectStatistics extends, org.?red5.?server.?api.?statistics.?IStatisticsBase
// Public Methods

    public int getActiveListeners();

    public int getMaxListeners();

    public String getName();

    public int getTotalChanges();

    public int getTotalDeletes();

    public int getTotalListeners();

    public int getTotalSends();

    public int getVersion();
```

```
public boolean isPersistentObject();  
}
```

## 4.2. getActiveListeners()

```
public int getActiveListeners();
```

Return current number of subscribed listeners.

### Parameters

<i>return</i>	number of listeners
---------------	---------------------

## 4.3. getMaxListeners()

```
public int getMaxListeners();
```

Return maximum number of concurrent subscribed listeners.

### Parameters

<i>return</i>	number of listeners
---------------	---------------------

## 4.4. getName()

```
public String getName();
```

Return the name of the shared object.

### Parameters

<i>return</i>	the name of the shared object
---------------	-------------------------------

## 4.5. getTotalChanges()

```
public int getTotalChanges();
```

Return number of attribute changes.

### Parameters

<i>return</i>	number of changes
---------------	-------------------

## 4.6. getTotalDeletes()

```
public int getTotalDeletes();
```

Return number of attribute deletes.

### Parameters

<i>return</i>	number of deletes
---------------	-------------------

#### 4.7. getTotalListeners()

```
public int getTotalListeners();
```

Return total number of subscribed listeners.

##### Parameters

<i>return</i>	number of listeners
---------------	---------------------

#### 4.8. getTotalSends()

```
public int getTotalSends();
```

Return number of times a message was sent.

##### Parameters

<i>return</i>	number of sends
---------------	-----------------

#### 4.9. getVersion()

```
public int getVersion();
```

Return the version number of the shared object.

##### Parameters

<i>return</i>	the version
---------------	-------------

#### 4.10. isPersistentObject()

```
public boolean isPersistentObject();
```

Check if the shared object is persistent.

##### Parameters

<i>return</i>	True if the shared object is persistent, otherwise False
---------------	--

### 5. Interface IStatisticsBase

Base class for all statistics informations.

#### 5.1. Synopsis

```
public interface IStatisticsBase {
// Public Methods

    public long getCreationTime();
```

}

## 5.2. getCreationTime()

```
public long getCreationTime();
```

Return the timestamp the object was created.

### Parameters

<i>return</i>	the timestamp in milliseconds since midnight, January 1, 1970 UTC.
---------------	--

## 6. Interface IStatisticsService

Statistics methods for Red5. They can be used to poll for updates of given elements inside the server. Statistics data will be stored as properties of different shared objects. Use `getScopeStatisticsSO` and `getSharedObjectStatisticsSO` to get these shared objects. The property names are `scopeName` for scope attributes and `scopeName|sharedObjectName` for shared object attributes. Each property holds a Map containing key/value mappings of the corresponding attributes. Sometime in the future, the updates on the shared objects will be done automatically so a client doesn't need to poll for them.

### 6.1. Synopsis

```
public interface IStatisticsService {
    // Public Methods

    public org.red5.server.api.so.ISharedObject getScopeStatisticsSO(org.red5.server.api.IScope scope)

    public java.util.Set<java.lang.String> getScopes();

    public java.util.Set<java.lang.String> getScopes(String path)
        throws ScopeNotFoundException;

    public org.red5.server.api.so.ISharedObject getSharedObjectStatisticsSO(org.red5.server.api.IScope scope)

    public java.util.Set<org.red5.server.api.statistics.ISharedObjectStatistics> getSharedObjects(String path)

    public void updateScopeStatistics(String path)
        throws ScopeNotFoundException;

    public void updateSharedObjectStatistics(String path,
        String name)
        throws ScopeNotFoundException, SharedObjectException;
}
```

## 6.2. getScopes()

```
public java.util.Set<java.lang.String> getScopes();
```

Return a list of all scopes that currently exist on the server.

#### Parameters

<i>return</i>	list of scope names
---------------	---------------------

### 6.3. getScopes(String)

```
public java.util.Set<java.lang.String> getScopes(String path)
throws ScopeNotFoundException;
```

Return a list of all scopes that currently exist on the server below a current path.

#### Parameters

path	Path to start looking for scopes.
<i>return</i>	list of scope names

ScopeNotFoundException

if the path on the server doesn't exist

### 6.4. getScopeStatisticsSO(IScope)

```
public org.red5.server.api.so.ISharedObject getScopeStatisticsSO(org.red5.server.api.IScope scope);
```

Return the shared object that will be used to keep scope statistics.

#### Parameters

scope	A scope to return the shared object for.
<i>return</i>	the shared object containing scope statistics

### 6.5. getSharedObjects(String)

```
public java.util.Set<org.red5.server.api.statistics.ISharedObjectStatistics> getSharedObjects(String path);
```

Return informations about shared objects for a given scope.

#### Parameters

path	Path to scope to return shared object names for.
<i>return</i>	list of informations about shared objects

### 6.6. getSharedObjectStatisticsSO(IScope)

```
public org.red5.server.api.so.ISharedObject getSharedObjectStatisticsSO(org.red5.server.api.IScope scope);
```

Return the shared object that will be used to keep SO statistics.

#### Parameters

scope	A scope to return the shared object for.
-------	--

`return`

the shared object containing SO statistics

## 6.7. updateScopeStatistics(String)

```
public void updateScopeStatistics(String path)
throws ScopeNotFoundException;
```

Update statistics for a given scope.

### Parameters

path	Path to scope to update.
------	--------------------------

ScopeNotFoundException  
if the given scope doesn't exist

## 6.8. updateSharedObjectStatistics(String, String)

```
public void updateSharedObjectStatistics(String path,
                                         String name)
throws ScopeNotFoundException, SharedObjectException;
```

Update informations about a shared object in a given scope.

### Parameters

path	Path to scope that contains the shared object.
name	Name of shared object to update.

ScopeNotFoundException  
if the given scope doesn't exist

SharedObjectException  
if no shared object with the given name exists

## 7. Interface IStreamStatistics

Base class for all stream statistics.

### 7.1. Synopsis

```
public interface IStreamStatistics extends, org.red5.server.api.statistics.IStatisticsBase {
// Public Methods

    public int getCurrentTimestamp();

}
```

### 7.2. getCurrentTimestamp()

```
public int getCurrentTimestamp();
```

Return the currently active timestamp inside the stream.

#### Parameters

<i>return</i>	the timestamp in milliseconds
---------------	-------------------------------

# 1. Class StatisticsCounter

Counts numbers used by the statistics. Keeps track of current, maximum and total numbers.

## 1.1. Synopsis

```
public class StatisticsCounter {  
    // Public Constructors  
  
    public StatisticsCounter();  
  
    // Public Methods  
  
    public void decrement();  
  
    public int getCurrent();  
  
    public int getMax();  
  
    public int getTotal();  
  
    public void increment();  
  
}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 1.2. decrement()

```
public void decrement();
```

Decrement statistics by one.

## 1.3. getCurrent()

```
public int getCurrent();
```

Get current number.

### Parameters

*return*

## 1.4. getMax()

```
public int getMax();
```

Get maximum number.

### Parameters

<i>return</i>	
---------------	--

## 1.5. getTotal()

public int getTotal();
------------------------

Get total number.

<b>Parameters</b>	
-------------------	--

<i>return</i>	
---------------	--

## 1.6. increment()

public void increment();
--------------------------

Increment statistics by one.

# 1. Interface IBroadcastStream

A broadcast stream is a stream source to be subscribed by clients. To subscribe a stream from your client Flash application use NetStream.play method. Broadcast stream can be saved at server-side.

## 1.1. Synopsis

```
public interface IBroadcastStream extends, org.red5.server.api.stream.IStream {  
    // Public Methods  
  
    public void addStreamListener(IStreamListener listener);  
  
    public org.red5.server.messaging.IProvider getProvider();  
  
    public String getPublishedName();  
  
    public String getSaveFilename();  
  
    public java.util.Collection<org.red5.server.api.stream.IStreamListener> getStreamListeners();  
  
    public void removeStreamListener(IStreamListener listener);  
  
    public void saveAs(String filePath,  
                      boolean isAppend)  
        throws IOException, ResourceNotFoundException, ResourceExistException;  
  
    public void setPublishedName(String name);  
}
```

## 1.2. addStreamListener(IStreamListener)

```
public void addStreamListener(IStreamListener listener);
```

Add a listener to be notified about received packets.

### Parameters

listener	the listener to add
----------	---------------------

## 1.3. getProvider()

```
public org.red5.server.messaging.IProvider getProvider();
```

Get the provider corresponding to this stream. Provider objects are object that

### Parameters

return	
--------	--

## 1.4. getPublishedName()

```
public String getPublishedName();
```

Get stream publish name. Publish name is the value of the first parameter had been passed to `NetStream.publish` on client side in SWF.

### Parameters

<i>return</i>	Stream publish name
---------------	---------------------

## 1.5. getSaveFilename()

```
public String getSaveFilename();
```

Get the filename the stream is being saved as.

### Parameters

<i>return</i>	The filename relative to the scope or <code>null</code> if the stream is not being saved.
---------------	---

## 1.6. getStreamListeners()

```
public java.util.Collection<org.red5.server.api.stream.IStreamListener> getStreamListeners();
```

Return registered stream listeners.

### Parameters

<i>return</i>	the registered listeners
---------------	--------------------------

## 1.7. removeStreamListener(IStreamListener)

```
public void removeStreamListener(IStreamListener listener);
```

Remove a listener from being notified about received packets.

### Parameters

<i>listener</i>	the listener to remove
-----------------	------------------------

## 1.8. saveAs(String, boolean)

```
public void saveAs(String filePath,
                   boolean isAppend)
throws IOException, ResourceNotFoundException, ResourceExistException;
```

Save the broadcast stream as a file.

### Parameters

filePath	The path of the file relative to the scope.
isAppend	Whether to append to the end of file.

`IOException`

File could not be created/written to.

`ResourceExistException`

Resource exist when trying to create.

`ResourceNotFoundException`

Resource not exist when trying to append.

## 1.9. setPublishedName(String)

```
public void setPublishedName(String name);
```

### Parameters

name	Set stream publish name
------	-------------------------

## 2. Interface IBroadcastStreamService

```
public interface IBroadcastStreamService {
    // Public Static Fields

    public static final String BROADCAST_STREAM_SERVICE = "broadcastStreamService";

    // Public Methods

    public IBroadcastStream getBroadcastStream(org.red5.server.api.IScope scope,
                                                String name);

    public java.util.List<java.lang.String> getBroadcastStreamNames(org.red5.server.api.IScope scope);

    public boolean hasBroadcastStream(org.red5.server.api.IScope scope,
                                      String name);
}
```

## 2.1. getBroadcastStream(IScope, String)

```
public IBroadcastStream getBroadcastStream(org.red5.server.api.IScope scope,
                                         String name);
```

Get a broadcast stream by name

### Parameters

scope	the scope to return the stream from
name	the name of the broadcast

<i>return</i>	broadcast stream object
---------------	-------------------------

## 2.2. getBroadcastStreamNames(IScope)

```
public java.util.List<java.lang.String> getBroadcastStreamNames(org.red5.server.api.IScope scope);
```

Get a set containing the names of all the broadcasts

### Parameters

scope	the scope to search for streams
<i>return</i>	set containing all broadcast names

## 2.3. hasBroadcastStream(IScope, String)

```
public boolean hasBroadcastStream(org.red5.server.api.IScope scope,
                                 String name);
```

Does the scope have a broadcast stream registered with a given name

### Parameters

scope	the scope to check for the stream
name	name of the broadcast
<i>return</i>	true if a stream exists, otherwise false

## 3. Interface IClientBroadcastStream

A broadcast stream that comes from client.

### 3.1. Synopsis

```
public interface IClientBroadcastStream extends, org.?red5.?server.?api.?stream.?IClientStream, org.?
// Public Methods

    public org.red5.server.api.statistics.IClientBroadcastStreamStatistics getStatistics();

    public void startPublishing();

}
```

## 3.2. getStatistics()

```
public org.red5.server.api.statistics.IClientBroadcastStreamStatistics getStatistics();
```

Return statistics about the stream.

### Parameters

<i>return</i>	statistics
---------------	------------

### 3.3. startPublishing()

```
public void startPublishing();
```

Notify client that stream is ready for publishing.

## 4. Interface IClientStream

A stream that is bound to a client.

### 4.1. Synopsis

```
public interface IClientStream extends, org.?red5.?server.?api.?stream.?IStream, org.?red5.?server.?a
// Public Static Fields

    public static final String MODE_APPEND = "append";

    public static final String MODE_LIVE = "live";

    public static final String MODE_READ = "read";

    public static final String MODE_RECORD = "record";

// Public Methods

    public IStreamCapableConnection getConnection();

    public int getStreamId();

    public void setClientBufferDuration(int bufferTime);

}
```

### 4.2. getConnection()

```
public IStreamCapableConnection getConnection();
```

Get connection containing the stream.

#### Parameters

<i>return</i>	the connection object or <code>null</code> if the connection is no longer active
---------------	--

### 4.3. getStreamId()

```
public int getStreamId();
```

Get stream id allocated in a connection.

#### Parameters

<i>return</i>
---------------

## 4.4. setClientBufferDuration(int)

```
public void setClientBufferDuration(int bufferTime);
```

Set the buffer duration for this stream as requested by the client.

### Parameters

bufferTime	duration in ms the client wants to buffer
------------	---

## 5. Interface IOnDemandStream

Extends stream to add methods for on demand access.

### 5.1. Synopsis

```
public interface IOnDemandStream extends, org.red5.server.api.stream.IStream {
    // Public Methods

    public boolean isPaused();

    public boolean isPlaying();

    public boolean isStopped();

    public void pause();

    public void play();

    public void play(int length);

    public void resume();

    public void seek(int position);

    public void stop();

}
```

### 5.2. isPaused()

```
public boolean isPaused();
```

Is the stream paused

### Parameters

return	true if the stream is paused
--------	------------------------------

### 5.3. isPlaying()

```
public boolean isPlaying();
```

Is the stream playing

#### Parameters

<i>return</i>	true if the stream is playing
---------------	-------------------------------

### 5.4. isStopped()

```
public boolean isStopped();
```

Is the stream stopped

#### Parameters

<i>return</i>	true if the stream is stopped
---------------	-------------------------------

### 5.5. pause()

```
public void pause();
```

Pause the stream

### 5.6. play()

```
public void play();
```

Start playback

### 5.7. play(int)

```
public void play(int length);
```

Start playback with a given maximum duration.

#### Parameters

<i>length</i>	maximum duration in milliseconds
---------------	----------------------------------

### 5.8. resume()

```
public void resume();
```

Resume a paused stream

### 5.9. seek(int)

```
public void seek(int position);
```

Seek to the keyframe nearest to position

#### Parameters

position	position in milliseconds
----------	--------------------------

## 5.10. stop()

```
public void stop();
```

**Specified by:** Method stop in interface IStream

Stop the stream, this resets the position to the start

# 6. Interface IOnDemandStreamService

```
public interface IOnDemandStreamService extends, org.?red5.?server.?api.?IScopeService {
// Public Static Fields

    public static final String BEAN_NAME = "onDemandStreamService";

// Public Methods

    public IOnDemandStream getOnDemandStream(org.red5.server.api.IScope scope,
                                              String name);

    public boolean hasOnDemandStream(org.red5.server.api.IScope scope,
                                     String name);

}
```

## 6.1. getOnDemandStream(IScope, String)

```
public IOnDemandStream getOnDemandStream(org.red5.server.api.IScope scope,
                                         String name);
```

Get a stream that can be used for playback of the on-demand stream

### Parameters

scope	the scope to return the stream from
name	the name of the stream
return	the on-demand stream

## 6.2. hasOnDemandStream(IScope, String)

```
public boolean hasOnDemandStream(org.red5.server.api.IScope scope,
                                 String name);
```

Has the service an on-demand stream with the passed name?

### Parameters

scope	the scope to check for the stream
name	the name of the stream

<i>return</i>	true if the stream exists, false otherwise
---------------	--

## 7. Interface IPlayItem

Playlist item. Each playlist item has name, start time, length in milliseconds and message input source.

### 7.1. Synopsis

```
public interface IPlayItem {
// Public Methods

    public long getLength();

    public org.red5.server.messaging.IMessageInput getMessageInput();

    public String getName();

    public long getSize();

    public long getStart();

}
```

### 7.2. getLength()

```
public long getLength();
```

Play length in milliseconds.

#### Parameters

<i>return</i>
---------------

### 7.3. getMessageInput()

```
public org.red5.server.messaging.IMessageInput getMessageInput();
```

Get a message input for play. This object overrides the default algorithm for finding the appropriate VOD or Live stream provider according to the item name.

#### Parameters

<i>return</i>
---------------

### 7.4. getName()

```
public String getName();
```

Get name of item. The VOD or Live stream provider is found according to this name.

**Parameters**

<i>return</i>	
---------------	--

**7.5. getSize()**

public long getSize();
------------------------

Size in bytes.

**Parameters**

<i>return</i>	
---------------	--

**7.6. getStart()**

public long getStart();
-------------------------

Start time in milliseconds.

**Parameters**

<i>return</i>	
---------------	--

**8. Interface IPlaylist**

Playlist

**8.1. Synopsis**

```
public interface IPlaylist {
    // Public Methods

    public void addItem(IPlayItem item);

    public void addItem(IPlayItem item,
                      int index);

    public IPlayItem getCurrentItem();

    public int getCurrentItemIndex();

    public IPlayItem getItem(int index);

    public int getItemCount();

    public boolean hasMoreItems();

    public boolean isRandom();

    public boolean isRepeat();

    public boolean isRewind();
}
```

```

    public void nextItem();

    public void previousItem();

    public void removeAllItems();

    public void removeItem(int index);

    public void setItem(int index);

    public void setPlaylistController(IPlaylistController controller);

    public void setRandom(boolean random);

    public void setRepeat(boolean repeat);

    public void setRewind(boolean rewind);

}

```

## 8.2. addItem(IPlayItem)

```
public void addItem(IPlayItem item);
```

Add an item to the list.

### Parameters

item	Playlist item
------	---------------

## 8.3. addItem(IPlayItem, int)

```
public void addItem(IPlayItem item,
                    int index);
```

Add an item to specific index.

### Parameters

item	Playlist item
index	Index in list

## 8.4. getCurrentItem()

```
public IPlayItem getCurrentItem();
```

Get currently playing item

### Parameters

return	Item
--------	------

## 8.5. getCurrentItemIndex()

```
public int getCurrentItemIndex();
```

Get currently playing item index.

### Parameters

<i>return</i>	Currently playing item index.
---------------	-------------------------------

## 8.6. getItem(int)

```
public IPlayItem getItem(int index);
```

Get the item according to the index.

### Parameters

index	Item index
<i>return</i>	Item at that index in list

## 8.7. getItemCount()

```
public int getItemCount();
```

Return number of items in list

### Parameters

<i>return</i>	Number of items in list
---------------	-------------------------

## 8.8. hasMoreItems()

```
public boolean hasMoreItems();
```

Check if the playlist has more items after the currently playing one.

### Parameters

<i>return</i>	true if more items are available, false otherwise
---------------	---

## 8.9. isRandom()

```
public boolean isRandom();
```

Whether items are randomly played.

### Parameters

<i>return</i>	true if shuffle is on for this list, false otherwise
---------------	--

## 8.10. isRepeat()

```
public boolean isRepeat();
```

Whether repeat playing an item.

### Parameters

<i>return</i>	<code>true</code> if repeat mode is on for this playlist, <code>false</code> otherwise
---------------	--

## 8.11. isRewind()

```
public boolean isRewind();
```

Whether rewind the list.

### Parameters

<i>return</i>	<code>true</code> if playlist is rewind on end, <code>false</code> otherwise
---------------	--

## 8.12. nextItem()

```
public void nextItem();
```

Go for next item decided by controller logic.

## 8.13. previousItem()

```
public void previousItem();
```

Go for the previous played item.

## 8.14. removeAllItems()

```
public void removeAllItems();
```

Remove all items.

## 8.15. removeItem(int)

```
public void removeItem(int index);
```

Remove an item from list.

### Parameters

<i>index</i>	Index in list
--------------	---------------

## 8.16. setItem(int)

```
public void setItem(int index);
```

Set the current item for playing.

**Parameters**

index	Position in list
-------	------------------

## 8.17. setPlaylistController(IPlaylistController)

```
public void setPlaylistController(IPlaylistController controller);
```

Set list controller.

**Parameters**

controller	Playlist controller
------------	---------------------

## 8.18. setRandom(boolean)

```
public void setRandom(boolean random);
```

Set whether items should be randomly played.

**Parameters**

random	Shuffle flag
--------	--------------

## 8.19. setRepeat(boolean)

```
public void setRepeat(boolean repeat);
```

Set whether repeat playing an item.

**Parameters**

repeat	New value for item playback repeat flag
--------	---

## 8.20. setRewind(boolean)

```
public void setRewind(boolean rewind);
```

Set whether rewind the list.

**Parameters**

rewind	New value for rewind flag
--------	---------------------------

# 9. Interface IPlaylistController

A play list controller that controls the order of play items.

## 9.1. Synopsis

```
public interface IPlaylistController {
    // Public Methods
}
```

```

public int nextItem(IPlaylist playlist,
                    int itemIndex);

public int previousItem(IPlaylist playlist,
                       int itemIndex);

}

```

## 9.2. nextItem(IPlaylist, int)

```
public int nextItem(IPlaylist playlist,
                    int itemIndex);
```

Get next item to play.

### Parameters

playlist	The related play list.
itemIndex	The current item index. -1 indicates to retrieve the first item for play.
<i>return</i>	The next item index to play. -1 reaches the end.

## 9.3. previousItem(IPlaylist, int)

```
public int previousItem(IPlaylist playlist,
                       int itemIndex);
```

Get previous item to play.

### Parameters

playlist	The related play list.
itemIndex	The current item index. <code>IPlaylist.itemSize</code> indicated to retrieve the last item for play.
<i>return</i>	The previous item index to play. -1 reaches the beginning.

# 10. Interface IPlaylistSubscriberStream

IPlaylistSubscriberStream has methods of both ISubscriberStream and IPlaylist but adds nothing new

## 10.1. Synopsis

```

public interface IPlaylistSubscriberStream extends, org.?red5.?server.?api.?stream.?ISubscriberStream
// Public Methods

    public org.red5.server.api.statistics.IPlaylistSubscriberStreamStatistics getStatistics();

}

```

## 10.2. getStatistics()

```
public org.red5.server.api.statistics.IPlaylistSubscriberStreamStatistics getStatistics();
```

Return statistics about this stream.

### Parameters

<i>return</i>	statistics
---------------	------------

## 11. Interface IServerStream

IServerStream has both IPlaylist and IBroadcastStream methods but add nothing new. It represents a stream broadcasted from the server.

### 11.1. Synopsis

```
public interface IServerStream extends, org.?red5.?server.?api.?stream.?IPlaylist, org.?red5.?server.?broadcast.?IBroadcastStream {
    // Public Methods

    public void pause();

    public void seek(int position);

}
```

### 11.2. pause()

```
public void pause();
```

Toggles the paused state.

### 11.3. seek(int)

```
public void seek(int position);
```

Seek to a given position in the stream.

### Parameters

<i>position</i>	new playback position in milliseconds
-----------------	---------------------------------------

## 12. Interface ISingleItemSubscriberStream

A subscriber stream that has only one item for play.

### 12.1. Synopsis

```
public interface ISingleItemSubscriberStream extends, org.?red5.?server.?api.?stream.?ISubscriberStream
```

```
// Public Methods

    public void setPlayItem(IPlayItem item);

}
```

## 12.2. setPlayItem(IPlayItem)

```
public void setPlayItem(IPlayItem item);
```

Setter for property 'playItem'.

### Parameters

item	Value to set for property 'playItem'.
------	---------------------------------------

# 13. Interface IStream

Base interface for stream objects. A stream object is always associated with a scope.

## 13.1. Synopsis

```
public interface IStream {
// Public Methods

    public void close();

    public IStreamCodecInfo getCodecInfo();

    public String getName();

    public org.red5.server.api.IScope getScope();

    public void start();

    public void stop();

}
```

## 13.2. close()

```
public void close();
```

Close this stream.

## 13.3. getCodecInfo()

```
public IStreamCodecInfo getCodecInfo();
```

Get Codec info for a stream.

**Parameters**

<i>return</i>
---------------

**13.4. getName()**

public String getName();
--------------------------

Get the name of the stream. The name is unique across the server. This is just an id of the stream and NOT the name that is used at client side to subscribe to the stream. For that name, use `getPublishedName()`

**Parameters**

<i>return</i>	the name of the stream
---------------	------------------------

**13.5. getScope()**

public org.red5.server.api.IScope getScope();
---

Get the scope this stream is associated with.

**Parameters**

<i>return</i>	scope object
---------------	--------------

**13.6. start()**

public void start();
----------------------

Start this stream.

**13.7. stop()**

public void stop();
---------------------

Stop this stream.

**14. Interface IStreamAwareScopeHandler**

A scope handler that is stream aware.

**14.1. Synopsis**

```
public interface IStreamAwareScopeHandler extends, org.?red5.?server.?api.?IScopeHandler {
// Public Methods

    public void streamBroadcastClose(IBroadcastStream stream);

    public void streamBroadcastStart(IBroadcastStream stream);

    public void streamPlaylistItemPlay(IPlaylistSubscriberStream stream,
                                      IPlayItem item,
```

```

        boolean isLive);

public void streamPlaylistItemStop(IPlaylistSubscriberStream stream,
                                  IPlayItem item);

public void streamPlaylistVODItemPause(IPlaylistSubscriberStream stream,
                                       IPlayItem item,
                                       int position);

public void streamPlaylistVODItemResume(IPlaylistSubscriberStream stream,
                                       IPlayItem item,
                                       int position);

public void streamPlaylistVODItemSeek(IPlaylistSubscriberStream stream,
                                      IPlayItem item,
                                      int position);

public void streamPublishStart(IBroadcastStream stream);

public void streamRecordStart(IBroadcastStream stream);

public void streamSubscriberClose(ISubscriberStream stream);

public void streamSubscriberStart(ISubscriberStream stream);

}

```

## 14.2. streamBroadcastClose(IBroadcastStream)

```
public void streamBroadcastClose(IBroadcastStream stream);
```

Notified when a broadcaster closes.

### Parameters

stream
--------

## 14.3. streamBroadcastStart(IBroadcastStream)

```
public void streamBroadcastStart(IBroadcastStream stream);
```

Notified when a broadcaster starts.

### Parameters

stream
--------

## 14.4. streamPlaylistItemPlay(IPlaylistSubscriberStream, IPlayItem, boolean)

```
public void streamPlaylistItemPlay(IPlaylistSubscriberStream stream,
                                  IPlayItem item,
                                  boolean isLive);
```

Notified when a playlist item plays.

#### Parameters

stream	
item	
isLive	TODO

### 14.5. streamPlaylistItemStop(IPlaylistSubscriberStream, IPlayItem)

```
public void streamPlaylistItemStop(IPlaylistSubscriberStream stream,
                                  IPlayItem item);
```

Notified when a playlist item stops.

#### Parameters

stream	
item	

### 14.6. streamPlaylistVODItemPause(IPlaylistSubscriberStream, IPlayItem, int)

```
public void streamPlaylistVODItemPause(IPlaylistSubscriberStream stream,
                                       IPlayItem item,
                                       int position);
```

Notified when a playlist vod item pauses.

#### Parameters

stream	
item	
position	

### 14.7. streamPlaylistVODItemResume(IPlaylistSubscriberStream, IPlayItem, int)

```
public void streamPlaylistVODItemResume(IPlaylistSubscriberStream stream,
                                         IPlayItem item,
                                         int position);
```

Notified when a playlist vod item resumes.

#### Parameters

stream	
item	
position	

## 14.8. streamPlaylistVODItemSeek(IPlaylistSubscriberStream, IPlayItem, int)

```
public void streamPlaylistVODItemSeek(IPlaylistSubscriberStream stream,
                                      IPlayItem item,
                                      int position);
```

Notified when a playlist vod item seeks.

### Parameters

stream	
--------	--

item	
------	--

position	
----------	--

## 14.9. streamPublishStart(IBroadcastStream)

```
public void streamPublishStart(IBroadcastStream stream);
```

A broadcast stream starts being published. This will be called when the first video packet has been received.

### Parameters

stream	
--------	--

## 14.10. streamRecordStart(IBroadcastStream)

```
public void streamRecordStart(IBroadcastStream stream);
```

A broadcast stream starts being recorded. This will be called when the first video packet has been received.

### Parameters

stream	
--------	--

## 14.11. streamSubscriberClose(ISubscriberStream)

```
public void streamSubscriberClose(ISubscriberStream stream);
```

Notified when a subscriber closes.

### Parameters

stream	
--------	--

## 14.12. streamSubscriberStart(ISubscriberStream)

```
public void streamSubscriberStart(ISubscriberStream stream);
```

Notified when a subscriber starts.

Parameters
------------

stream
--------

## 15. Interface IStreamCapableConnection

A connection that supports streaming.

### 15.1. Synopsis

```
public interface IStreamCapableConnection extends, org.?red5.?server.?api.?IConnection, org.?red5.?se
// Public Methods

    public void deleteStreamById(int streamId);

    public long getPendingVideoMessages(int streamId);

    public IClientStream getStreamById(int streamId);

    public IClientBroadcastStream newBroadcastStream(int streamId);

    public IPlaylistSubscriberStream newPlaylistSubscriberStream(int streamId);

    public ISingleItemSubscriberStream newSingleItemSubscriberStream(int streamId);

    public int reserveStreamId();

    public void unreserveStreamId(int streamId);

}
```

### 15.2. deleteStreamById(int)

```
public void deleteStreamById(int streamId);
```

Deletes the stream with the given id.

Parameters
------------

streamId	ID of stream to delete
----------	------------------------

### 15.3. getPendingVideoMessages(int)

```
public long getPendingVideoMessages(int streamId);
```

Total number of video messages that are pending to be sent to a stream.

Parameters
------------

streamId	Stream id
----------	-----------

return	Number of pending video messages
--------	----------------------------------

## 15.4. getStreamById(int)

```
public IClientStream getStreamById(int streamId);
```

Get a stream by its id.

### Parameters

streamId	Stream id
<i>return</i>	Stream with given id

## 15.5. newBroadcastStream(int)

```
public IClientBroadcastStream newBroadcastStream(int streamId);
```

Create a broadcast stream.

### Parameters

streamId	Stream id
<i>return</i>	New broadcast stream

## 15.6. newPlaylistSubscriberStream(int)

```
public IPlaylistSubscriberStream newPlaylistSubscriberStream(int streamId);
```

Create a stream that can play a list.

### Parameters

streamId	Stream id
<i>return</i>	New stream that can play sequence of items

## 15.7. newSingleItemSubscriberStream(int)

```
public ISingleItemSubscriberStream newSingleItemSubscriberStream(int streamId);
```

Create a stream that can play only one item.

### Parameters

streamId	Stream id
<i>return</i>	New subscriber stream that can play only one item

## 15.8. reserveStreamId()

```
public int reserveStreamId();
```

Return a reserved stream id for use. According to FCS/FMS regulation, the base is 1.

### Parameters

<i>return</i>	Reserved stream id
---------------	--------------------

## 15.9. unreserveStreamId(int)

```
public void unreserveStreamId(int streamId);
```

Unreserve this id for future use.

### Parameters

streamId	ID of stream to unreserve
----------	---------------------------

# 16. Interface IStreamCodecInfo

Stream codec information

## 16.1. Synopsis

```
public interface IStreamCodecInfo {
// Public Methods

    public String getAudioCodecName();

    public IVideoStreamCodec getVideoCodec();

    public String getVideoCodecName();

    public boolean hasAudio();

    public boolean hasVideo();

}
```

## 16.2. getAudioCodecName()

```
public String getAudioCodecName();
```

Getter for audio codec name

### Parameters

<i>return</i>	Audio codec name
---------------	------------------

## 16.3. getVideoCodec()

```
public IVideoStreamCodec getVideoCodec();
```

Return video codec

### Parameters

<i>return</i>	Video codec used by stream codec
---------------	----------------------------------

## 16.4. getVideoCodecName()

```
public String getVideoCodecName();
```

Getter for video codec name

### Parameters

<i>return</i>	Video codec name
---------------	------------------

## 16.5. hasAudio()

```
public boolean hasAudio();
```

Has audio support?

### Parameters

<i>return</i>	true if stream codec has audio support, false otherwise
---------------	---

## 16.6. hasVideo()

```
public boolean hasVideo();
```

Has video support?

### Parameters

<i>return</i>	true if stream codec has video support, false otherwise
---------------	---

# 17. Interface IStreamFilenameGenerator

A class that can generate filenames for streams.

## 17.1. Synopsis

```
public interface IStreamFilenameGenerator extends, org.?red5.?server.?api.?IScopeService {
    // Public Static Fields

    public static final String BEAN_NAME = "streamFilenameGenerator";

    // Public Methods

    public String generateFilename(org.red5.server.api.IScope scope,
                                  String name,
                                  String extension,
                                  org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType type);

    public String generateFilename(org.red5.server.api.IScope scope,
                                  String name,
                                  org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType type);

    public boolean resolvesToAbsolutePath();

}
```

## 17.2. BEAN\_NAME

```
public static final String BEAN_NAME = "streamFilenameGenerator";
```

Name of the bean to setup a custom filename generator in an application.

## 17.3. generateFilename(Icope, String, IStreamFilenameGenerator.GenerationType)

```
public String generateFilename(org.red5.server.api.Icope scope,
                           String name,
                           org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType type);
```

Generate a filename without an extension.

### Parameters

scope	Scope to use
name	Stream name
type	Generation strategy (either playback or record)
<i>return</i>	Full filename

## 17.4. generateFilename(Icope, String, String, IStreamFilenameGenerator.GenerationType)

```
public String generateFilename(org.red5.server.api.Icope scope,
                           String name,
                           String extension,
                           org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType type);
```

Generate a filename with an extension.

### Parameters

scope	Scope to use
name	Stream filename
extension	Extension
type	Generation strategy (either playback or record)
<i>return</i>	Full filename with extension

## 17.5. resolvesToAbsolutePath()

```
public boolean resolvesToAbsolutePath();
```

True if returned filename is an absolute path, else relative to application. If relative to application, you need to use `scope.getContext().getResources(fileName)`

[0].getFile() to resolve this to a file. If absolute (ie returns true) simply use new File(generateFilename(scope, name))

#### Parameters

<i>return</i>
---------------

## 18. Class IStreamFilenameGenerator.GenerationType

Possible filename generation types.

### 18.1. Synopsis

```
public static final class IStreamFilenameGenerator.GenerationType extends, java.?lang.?Enum {
// Public Static Fields

    public static final org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType PLAYBACK ;

    public static final org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType RECORD ;

// Public Static Methods

    public static org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType valueOf(String na

    public static org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType[] values();

}
```

**Methods inherited from java.lang.Enum:** clone, compareTo, equals, finalize, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

**Methods inherited from java.lang.Object:** getClass, notify, notifyAll, wait

## 19. Interface IStreamHandler

```
public interface IStreamHandler {
// Public Methods

    public void onBroadcastStreamStart(IStream stream);

    public void onBroadcastStreamSubscribe(IBroadcastStream stream);

    public void onBroadcastStreamUnsubscribe(IBroadcastStream stream);

    public void onOnDemandStreamConnect(IOnDemandStream stream);

    public void onOnDemandStreamDisconnect(IOnDemandStream stream);

    public void onRecordStreamStart(IStream stream);

    public void onRecordStreamStop(IStream stream);

    public void onStreamPublishStart(IStream stream);
```

```
public void onStreamPublishStop(IStream stream);  
}
```

## 19.1. onBroadcastStreamStart(IStream)

```
public void onBroadcastStreamStart(IStream stream);
```

Called when the broadcast starts

### Parameters

stream	the stream object
--------	-------------------

## 19.2. onBroadcastStreamSubscribe(IBroadcastStream)

```
public void onBroadcastStreamSubscribe(IBroadcastStream stream);
```

Called when a client subscribes to a broadcast

### Parameters

stream	the stream object
--------	-------------------

## 19.3. onBroadcastStreamUnsubscribe(IBroadcastStream)

```
public void onBroadcastStreamUnsubscribe(IBroadcastStream stream);
```

Called when a client unsubscribes from a broadcast

### Parameters

stream	the stream object
--------	-------------------

## 19.4. onOnDemandStreamConnect(IOnDemandStream)

```
public void onOnDemandStreamConnect(IOnDemandStream stream);
```

Called when a client connects to an on demand stream

### Parameters

stream	the stream object
--------	-------------------

## 19.5. onOnDemandStreamDisconnect(IOnDemandStream)

```
public void onOnDemandStreamDisconnect(IOnDemandStream stream);
```

Called when a client disconnects from an on demand stream

### Parameters

stream

the stream object

## 19.6. onRecordStreamStart(IStream)

```
public void onRecordStreamStart(IStream stream);
```

Called when a recording starts

Parameters

stream

the stream object

## 19.7. onRecordStreamStop(IStream)

```
public void onRecordStreamStop(IStream stream);
```

Called when a recording stops

Parameters

stream

the stream object

## 19.8. onStreamPublishStart(IStream)

```
public void onStreamPublishStart(IStream stream);
```

Called when the client begins publishing

Parameters

stream

the stream object

## 19.9. onStreamPublishStop(IStream)

```
public void onStreamPublishStop(IStream stream);
```

Called when the client stops publishing

Parameters

stream

the stream object

# 20. Interface IStreamListener

Listener that is notified about packets received from a stream.

## 20.1. Synopsis

```
public interface IStreamListener {
    // Public Methods

    public void packetReceived(IBroadcastStream stream,
```

```
IStreamPacket packet);  
}
```

## 20.2. packetReceived(IBroadcastStream, IStreamPacket)

```
public void packetReceived(IBroadcastStream stream,  
                           IStreamPacket packet);
```

A packet has been received from a stream.

### Parameters

stream	the stream the packet has been received for
packet	the packet received

# 21. Interface IStreamPacket

Packet containing stream data.

## 21.1. Synopsis

```
public interface IStreamPacket {  
    // Public Methods  
  
    public org.apache.mina.common.ByteBuffer getData();  
  
    public byte getDataType();  
  
    public int getTimestamp();  
}
```

## 21.2. getData()

```
public org.apache.mina.common.ByteBuffer getData();
```

Packet contents.

### Parameters

<i>return</i>	the contents
---------------	--------------

## 21.3. getDataType()

```
public byte getDataType();
```

Type of this packet. This is one of the `TYPE_` constants.

### Parameters

<i>return</i>	the type
---------------	----------

## 21.4. getTimestamp()

```
public int getTimestamp();
```

Timestamp of this packet.

### Parameters

<i>return</i>	the timestamp in milliseconds
---------------	-------------------------------

## 22. Interface IStreamPlaybackSecurity

Interface for handlers that control access to stream playback.

### 22.1. Synopsis

```
public interface IStreamPlaybackSecurity {
    // Public Methods

    public boolean isPlaybackAllowed(org.red5.server.api.IScope scope,
                                     String name,
                                     int start,
                                     int length,
                                     boolean flushPlaylist);

}
```

### 22.2. isPlaybackAllowed(IScope, String, int, int, boolean)

```
public boolean isPlaybackAllowed(org.red5.server.api.IScope scope,
                                 String name,
                                 int start,
                                 int length,
                                 boolean flushPlaylist);
```

Check if playback of a stream with the given name is allowed.

### Parameters

scope	Scope the stream is about to be played back from.
-------	---

name	Name of the stream to play.
------	-----------------------------

start	Position to start playback from (in milliseconds).
-------	--

length	Duration to play (in milliseconds).
--------	-------------------------------------

flushPlaylist	Flush playlist?
---------------	-----------------

<i>return</i>	True if playback is allowed, otherwise False
---------------	--

## 23. Interface IStreamPublishSecurity

Interface for handlers that control access to stream publishing.

## 23.1. Synopsis

```
public interface IStreamPublishSecurity {
// Public Methods

    public boolean isPublishAllowed(org.red5.server.api.IScope scope,
                                    String name,
                                    String mode);

}
```

## 23.2. isPublishAllowed(IScope, String, String)

```
public boolean isPublishAllowed(org.red5.server.api.IScope scope,
                               String name,
                               String mode);
```

Check if publishing a stream with the given name is allowed.

Parameters	
scope	Scope the stream is about to be published in.
name	Name of the stream to publish.
mode	Publishing mode.
<i>return</i>	True if publishing is allowed, otherwise False

# 24. Interface IStreamSecurityService

Service that supports protecting access to streams.

## 24.1. Synopsis

```
public interface IStreamSecurityService extends, org.?red5.?server.?api.?IScopeService {
// Public Static Fields

    public static final String BEAN_NAME = "streamSecurityService";

// Public Methods

    public java.util.Set<org.red5.server.api.stream.IStreamPlaybackSecurity> getStreamPlaybackSecurity();

    public java.util.Set<org.red5.server.api.stream.IStreamPublishSecurity> getStreamPublishSecurity();

    public void registerStreamPlaybackSecurity(IStreamPlaybackSecurity handler);

    public void registerStreamPublishSecurity(IStreamPublishSecurity handler);

    public void unregisterStreamPlaybackSecurity(IStreamPlaybackSecurity handler);

    public void unregisterStreamPublishSecurity(IStreamPublishSecurity handler);
```

}

## 24.2. BEAN\_NAME

```
public static final String BEAN_NAME = "streamSecurityService";
```

Name of a bean defining that scope service.

## 24.3. getStreamPlaybackSecurity()

```
public java.util.Set<org.red5.server.api.stream.IStreamPlaybackSecurity> getStreamPlaybackSecurity();
```

Get handlers that protect stream playback.

### Parameters

<i>return</i>	list of handlers
---------------	------------------

## 24.4. getStreamPublishSecurity()

```
public java.util.Set<org.red5.server.api.stream.IStreamPublishSecurity> getStreamPublishSecurity();
```

Get handlers that protect stream publishing.

### Parameters

<i>return</i>	list of handlers
---------------	------------------

## 24.5. registerStreamPlaybackSecurity(IStreamPlaybackSecurity)

```
public void registerStreamPlaybackSecurity(IStreamPlaybackSecurity handler);
```

Add handler that protects stream playback.

### Parameters

<i>handler</i>	Handler to add.
----------------	-----------------

## 24.6. registerStreamPublishSecurity(IStreamPublishSecurity)

```
public void registerStreamPublishSecurity(IStreamPublishSecurity handler);
```

Add handler that protects stream publishing.

### Parameters

<i>handler</i>	Handler to add.
----------------	-----------------

## 24.7. unregisterStreamPlaybackSecurity(IStreamPlaybackSecurity)

```
public void unregisterStreamPlaybackSecurity(IStreamPlaybackSecurity handler);
```

Remove handler that protects stream playback.

#### Parameters

handler	Handler to remove.
---------	--------------------

## 24.8. unregisterStreamPublishSecurity(IStreamPublishSecurity)

```
public void unregisterStreamPublishSecurity(IStreamPublishSecurity handler);
```

Remove handler that protects stream publishing.

#### Parameters

handler	Handler to remove.
---------	--------------------

## 25. Interface IStreamService

This interface represents the stream methods that can be called through RTMP.

### 25.1. Synopsis

```
public interface IStreamService extends, org.?red5.?server.?api.?IScopeService {
// Public Static Fields
```

```
    public static final String BEAN_NAME = "streamService";
```

```
// Public Methods
```

```
    public void closeStream();
```

```
    public int createStream();
```

```
    public void deleteStream(int streamId);
```

```
    public void deleteStream(IStreamCapableConnection conn,
                           int streamId);
```

```
    public void pause(boolean pausePlayback,
                      int position);
```

```
    public void play(Boolean dontStop);
```

```
    public void play(String name);
```

```
    public void play(String name,
                     int start);
```

```
    public void play(String name,
                     int start,
                     int length);
```

```
    public void play(String name,
                     int start,
                     int length,
                     boolean flushPlaylist);
```

```

public void publish(Boolean dontStop);

public void publish(String name);

public void publish(String name,
                   String mode);

public void receiveAudio(boolean receive);

public void receiveVideo(boolean receive);

public void releaseStream(String streamName);

public void seek(int position);

}

```

## 25.2. closeStream()

```
public void closeStream();
```

Close the stream but not deallocate the resources.

## 25.3. createStream()

```
public int createStream();
```

Create a stream and return a corresponding id.

### Parameters

<i>return</i>	ID of created stream
---------------	----------------------

## 25.4. deleteStream(int)

```
public void deleteStream(int streamId);
```

Close the stream if not been closed. Deallocation the related resources.

### Parameters

streamId	Stram id
----------	----------

## 25.5. deleteStream(IStreamCapableConnection, int)

```
public void deleteStream(IStreamCapableConnection conn,
                        int streamId);
```

Delete stream

### Parameters

conn	Stream capable connection
streamId	Stream id

## 25.6. pause(boolean, int)

```
public void pause(boolean pausePlayback,
                  int position);
```

Pauses playback

### Parameters

pausePlayback	Pause flag
position	Pause position

## 25.7. play(Boolean)

```
public void play(Boolean dontStop);
```

Play stream without initial stop

### Parameters

dontStop	Stoppage flag
----------	---------------

## 25.8. play(String)

```
public void play(String name);
```

Play stream with name

### Parameters

name	Stream name
------	-------------

## 25.9. play(String, int)

```
public void play(String name,
                 int start);
```

Play stream with name from start position

### Parameters

name	Stream name
start	Start position

## 25.10. play(String, int, int)

```
public void play(String name,
                 int start,
```

```
int length);
```

Play stream with name from start position and for given amount if time

#### Parameters

name	Stream name
start	Start position
length	Playback length

### 25.11. play(String, int, int, boolean)

```
public void play(String name,
                 int start,
                 int length,
                 boolean flushPlaylist);
```

Publishes stream from given position for given amount of time

#### Parameters

name	Stream published name
start	Start position
length	Playback length
flushPlaylist	Flush playlist?

### 25.12. publish(Boolean)

```
public void publish(Boolean dontStop);
```

Publish

#### Parameters

dontStop	Whether need to stop first
----------	----------------------------

### 25.13. publish(String)

```
public void publish(String name);
```

Publishes stream with given name

#### Parameters

name	Stream published name
------	-----------------------

### 25.14. publish(String, String)

```
public void publish(String name,
                   String mode);
```

Publishes stream with given name and mode

#### Parameters

name	Stream published name
mode	Stream publishing mode

### 25.15. receiveAudio(boolean)

```
public void receiveAudio(boolean receive);
```

Can recieve audio?

#### Parameters

receive	Boolean flag
---------	--------------

### 25.16. receiveVideo(boolean)

```
public void receiveVideo(boolean receive);
```

Can recieve video?

#### Parameters

receive	Boolean flag
---------	--------------

### 25.17. releaseStream(String)

```
public void releaseStream(String streamName);
```

Called by FME.

#### Parameters

streamName
------------

### 25.18. seek(int)

```
public void seek(int position);
```

Seek to position

#### Parameters

position	Seek position
----------	---------------

## 26. Interface ISubscriberStream

ISubscriberStream is a stream from subscriber's point of view. That is, it provides methods for common stream operations like play, pause or seek.

## 26.1. Synopsis

```
public interface ISubscriberStream extends, org.?red5.?server.?api.?stream.?IClientStream {
    // Public Methods

    public boolean isPaused();

    public void pause(int position);

    public void play()
        throws IOException;

    public void receiveAudio(boolean receive);

    public void receiveVideo(boolean receive);

    public void resume(int position);

    public void seek(int position)
        throws OperationNotSupportedException;

    public void stop();
}
```

## 26.2. isPaused()

public boolean isPaused();
----------------------------

Check if the stream is currently paused.

### Parameters

<i>return</i>
---------------

## 26.3. pause(int)

public void pause(int position);
----------------------------------

Pause at a position for current playing item.

### Parameters

position	Position for pause in millisecond.
----------	------------------------------------

## 26.4. play()

public void play()         throws IOException;
--

Start playing.

**IOException**  
if an IO error occurred while starting to play the stream

## 26.5. receiveAudio(boolean)

```
public void receiveAudio(boolean receive);
```

Should the stream send audio to the client?

### Parameters

receive
---------

## 26.6. receiveVideo(boolean)

```
public void receiveVideo(boolean receive);
```

Should the stream send video to the client?

### Parameters

receive
---------

## 26.7. resume(int)

```
public void resume(int position);
```

Resume from a position for current playing item.

### Parameters

position	Position for resume in millisecond.
----------	-------------------------------------

## 26.8. seek(int)

```
public void seek(int position)
    throws OperationNotSupportedException;
```

Seek into a position for current playing item.

### Parameters

position	Position for seek in millisecond.
----------	-----------------------------------

`OperationNotSupportedException`  
if the stream doesn't support seeking.

## 26.9. stop()

```
public void stop();
```

Stop playing.

## 27. Interface ISubscriberStreamService

```
public interface ISubscriberStreamService extends, org.?red5.?server.?api.?IScopeService {
    // Public Static Fields
```

```

public static final String BEAN_NAME = "subscriberStreamService";

// Public Methods

public ISubscriberStream getSubscriberStream(org.red5.server.api.IScope scope,
                                              String name);

}

```

## 27.1. getSubscriberStream(IScope, String)

```

public ISubscriberStream getSubscriberStream(org.red5.server.api.IScope scope,
                                             String name);

```

Returns a stream that can subscribe a broadcast stream with the given name using "IBroadcastStream.subscribe".

### Parameters

scope	the scope to return the stream from
name	the name of the stream
<i>return</i>	the stream object

## 28. Interface IVideoStreamCodec

```

public interface IVideoStreamCodec {
    // Public Methods

    public boolean addData(org.apache.mina.common.ByteBuffer data);

    public boolean canDropFrames();

    public boolean canHandleData(org.apache.mina.common.ByteBuffer data);

    public org.apache.mina.common.ByteBuffer getKeyframe();

    public String getName();

    public void reset();

}

```

## 28.1. addData(ByteBuffer)

```

public boolean addData(org.apache.mina.common.ByteBuffer data);

```

Update the state of the codec with the passed data.

### Parameters

<i>data</i>	
<i>return</i>	

## 28.2. canDropFrames()

```
public boolean canDropFrames();
```

Check if the codec supports frame dropping.

Parameters	
------------	--

<i>return</i>	
---------------	--

## 28.3. canHandleData(ByteBuffer)

```
public boolean canHandleData(org.apache.mina.common.ByteBuffer data);
```

Returns true if the codec knows how to handle the passed stream data.

Parameters	
------------	--

<i>data</i>	
-------------	--

<i>return</i>	
---------------	--

## 28.4. getKeyframe()

```
public org.apache.mina.common.ByteBuffer getKeyframe();
```

Return the data for a keyframe.

Parameters	
------------	--

<i>return</i>	
---------------	--

## 28.5. getName()

```
public String getName();
```

Return the name of the video codec.

Parameters	
------------	--

<i>return</i>	
---------------	--

## 28.6. reset()

```
public void reset();
```

Reset the codec to its initial state.

# 29. Exception OperationNotSupportedException

The requested operation is not supported by the stream.

## 29.1. Synopsis

```
public class OperationNotSupportedException extends java.lang.Exception {
// Public Constructors

    public OperationNotSupportedException();

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 30. Exception ResourceExistException

```
public class ResourceExistException extends java.lang.Exception {
// Public Constructors

    public ResourceExistException();

    public ResourceExistException(String message);

    public ResourceExistException(String message,
                                  Throwable cause);

    public ResourceExistException(Throwable cause);

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

### 30.1. ResourceExistException()

```
public ResourceExistException();
```

Constructs a new ResourceExistException.

## 31. Exception ResourceNotFoundException

```
public class ResourceNotFoundException extends java.lang.Exception {
// Public Constructors
```

```
public ResourceNotFoundException();  
  
public ResourceNotFoundException(String message);  
  
public ResourceNotFoundException(String message,  
                               Throwable cause);  
  
public ResourceNotFoundException(Throwable cause);  
}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

### 31.1. ResourceNotFoundException()

```
public ResourceNotFoundException();
```

Constructs a new ResourceNotFoundException.

# 1. Class DenyAllStreamAccess

Stream security handler that denies access to all streams.

## 1.1. Synopsis

```
public class DenyAllStreamAccess implements org.red5.server.api.stream.IStreamPublishSecurity, c
// Public Constructors

    public DenyAllStreamAccess();

// Public Methods

    public boolean isPlaybackAllowed(org.red5.server.api.IScope scope,
                                    String name,
                                    int start,
                                    int length,
                                    boolean flushPlaylist);

    public boolean isPublishAllowed(org.red5.server.api.IScope scope,
                                String name,
                                String mode);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. isPlaybackAllowed(IScope, String, int, int, boolean)

```
public boolean isPlaybackAllowed(org.red5.server.api.IScope scope,
                                String name,
                                int start,
                                int length,
                                boolean flushPlaylist);
```

**Specified by:** Method isPlaybackAllowed in interface IStreamPlaybackSecurity

Check if playback of a stream with the given name is allowed.

## 1.3. isPublishAllowed(IScope, String, String)

```
public boolean isPublishAllowed(org.red5.server.api.IScope scope,
                                String name,
                                String mode);
```

**Specified by:** Method isPublishAllowed in interface IStreamPublishSecurity

Check if publishing a stream with the given name is allowed.

# 2. Class SimpleBandwidthConfigure

This class is the only IBandwidthConfigure implementation provided in 0.6. It's a kind of ValueObject (item with a set of values that just stores data) that is used to configure Red5

application bandwidth settings. Note: To configure the connection's bandwidth, you should use the implementation of `org.red5.server.api.IConnectionBWConfig` instead.

## 2.1. Synopsis

```
public class SimpleBandwidthConfigure implements org.red5.server.api.IBandwidthConfigure {
    // Public Constructors

    public SimpleBandwidthConfigure();

    public SimpleBandwidthConfigure(org.red5.server.api.IBandwidthConfigure config);

    // Public Methods

    public long[] getChannelBandwidth();

    public long[] getChannelInitialBurst();

    // Protected Methods

    protected Object clone()
        throws CloneNotSupportedException;

}
```

**Direct known subclasses:** `org.red5.server.api.stream.support.SimpleConnectionBWConfig`

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

### See Also

`org.red5.server.api.IConnectionBWConfig`,  
`org.red5.server.api.stream.support.SimpleConnectionBWConfig`

## 2.2. getChannelBandwidth()

```
public long[] getChannelBandwidth();
```

**Specified by:** Method `getChannelBandwidth` in interface `IBandwidthConfigure`

Return the bandwidth configure for 3 channels: audio, video, data and the overall bandwidth. The unit is bit per second. A value of `-1` means "don't care" so that there's no limit on bandwidth for that channel. The last element is the overall bandwidth. If it's not `-1`, the value of the first three elements will be ignored.

### Parameters

<code>return</code>	The 4-element array of bandwidth configure.
---------------------	---

**Description copied from interface: `getChannelBandwidth`**

## 2.3. getChannelInitialBurst()

```
public long[] getChannelInitialBurst();
```

**Specified by:** Method getChannelInitialBurst in interface IBandwidthConfigure

Return the byte count of initial burst value for 3 channels: audio, video, data and the overall bandwidth. If the value is -1, the default will be used per the implementation of bandwidth controller.

#### Parameters

<i>return</i>	The 4-element array of byte count of initial burst value.
---------------	---

**Description copied from interface: getChannelInitialBurst**

## 3. Class SimpleConnectionBWConfig

Simple implementation of connection bandwidth configuration.

### 3.1. Synopsis

```
public class SimpleConnectionBWConfig extends, org.?red5.?server.?api.?stream.?support.?SimpleBandwidth
    implements, org.?red5.?server.?api.?IConnectionBWConfig {
    // Public Constructors

    public SimpleConnectionBWConfig();

    // Public Methods

    public long getDownstreamBandwidth();

    public long getUpstreamBandwidth();

    public void setUpstreamBandwidth(long bw);

}
```

#### Methods inherited from

**org.red5.server.api.stream.support.SimpleBandwidthConfigure:** clone ,  
getChannelBandwidth , getChannelInitialBurst

**Methods inherited from java.lang.Object:** equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

### 3.2. getDownstreamBandwidth()

```
public long getDownstreamBandwidth();
```

**Specified by:** Method getDownstreamBandwidth in interface IConnectionBWConfig

Getter for downstream bandwidth

#### Parameters

<i>return</i>	Downstream bandwidth, from server to client
---------------	---

**Description copied from interface: getDownstreamBandwidth**

### 3.3. getUpstreamBandwidth()

```
public long getUpstreamBandwidth();
```

**Specified by:** Method getUpstreamBandwidth in interface IConnectionBWConfig

Get the upstream bandwidth to be notified to the client. Upstream is the data that is sent from the client to the server.

**Parameters**

<i>return</i>	Upstream (from client to server) bandwidth configuration
---------------	--

**Description copied from interface: getUpstreamBandwidth**

### 3.4. setUpstreamBandwidth(long)

```
public void setUpstreamBandwidth(long bw);
```

**Specified by:** Method setUpstreamBandwidth in interface IConnectionBWConfig

Set the upstream bandwidth to be notified to the client. Upstream is the data that is sent from the client to the server.

**Parameters**

<i>bw</i>	Bandwidth
-----------	-----------

**Description copied from interface: setUpstreamBandwidth**

## 4. Class SimplePlayItem

Simple playlist item implementation

### 4.1. Synopsis

```
public class SimplePlayItem implements, org.?red5.?server.?api.?stream.?IPlayItem {
// Public Constructors

    public SimplePlayItem();

// Public Methods

    public long getLength();

    public org.red5.server.messaging.IMessageInput getMessageInput();

    public org.red5.server.messaging.IMessageInput getMsgInput();

    public String getName();

    public long getSize();
```

```

public long getStart();

public void setLength(long length);

public void setMsgInput(org.red5.server.messaging.IMessageInput msgInput);

public void setName(String name);

public void setSize(long size);

public void setStart(long start);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 4.2. getLength()

```
public long getLength();
```

**Specified by:** Method getLength in interface IPlayItem

Returns play item length in milliseconds

### Parameters

<i>return</i>	Play item length in milliseconds
---------------	----------------------------------

## 4.3. getMessageInput()

```
public org.red5.server.messaging.IMessageInput getMessageInput();
```

**Specified by:** Method getMessageInput in interface IPlayItem

Returns IMessagelInput object. IMessagelInput is an endpoint for a consumer to connect.

### Parameters

<i>return</i>	IMessagelInput object
---------------	-----------------------

## 4.4. getMsgInput()

```
public org.red5.server.messaging.IMessageInput getMsgInput();
```

Alias for getMessageInput

### Parameters

<i>return</i>	Message input source
---------------	----------------------

## 4.5. getName()

```
public String getName();
```

**Specified by:** Method getName in interface IPlayItem

Returns item name

### Parameters

<i>return</i>	item name
---------------	-----------

## 4.6. getSize()

```
public long getSize();
```

**Specified by:** Method getSize in interface IPlayItem

Returns size in bytes

## 4.7. getStart()

```
public long getStart();
```

**Specified by:** Method getStart in interface IPlayItem

Returns boolean value that specifies whether item can be played

## 4.8. setLength(long)

```
public void setLength(long length);
```

Setter for length

### Parameters

<i>length</i>	Item length.
---------------	--------------

## 4.9. setMsgInput(IMessageInput)

```
public void setMsgInput(org.red5.server.messaging.IMessageInput msgInput);
```

Setter for message input

### Parameters

<i>msgInput</i>	Message input
-----------------	---------------

## 4.10. setName(String)

```
public void setName(String name);
```

Setter name

**Parameters**

name	Item name
------	-----------

**4.11. setSize(long)**

```
public void setSize(long size);
```

Set the size in bytes

**Parameters**

size
------

**4.12. setStart(long)**

```
public void setStart(long start);
```

Setter for start.

**Parameters**

start	Start position.
-------	-----------------

**5. Class StreamUtils**

Stream helper methods.

**5.1. Synopsis**

```
public abstract class StreamUtils {
// Public Constructors

    public StreamUtils();

// Public Static Methods

    public static org.red5.server.api.stream.IServerStream createServerStream(org.red5.server.api.IScope
        String name);

    public static org.red5.server.api.stream.IServerStream getServerStream(org.red5.server.api.IScope
        String name);

    public static void putServerStream(org.red5.server.api.IScope scope,
        String name,
        org.red5.server.api.stream.IServerStream stream);

    public static void removeServerStream(org.red5.server.api.IScope scope,
        String name);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 5.2. createServerStream(IScope, String)

```
public static org.red5.server.api.stream.IServerStream createServerStream(org.red5.server.api.IScope
String name);
```

Creates server stream

### Parameters

scope	Scope of stream
name	Name of stream
<i>return</i>	IStream object

## 5.3. getServerStream(IScope, String)

```
public static org.red5.server.api.stream.IServerStream getServerStream(org.red5.server.api.IScope
String name);
```

Looks up a server stream in the stream map. Null will be returned if the stream is not found.

### Parameters

scope	Scope of stream
name	Name of stream
<i>return</i>	IStream object

## 5.4. putServerStream(IScope, String, IStream)

```
public static void putServerStream(org.red5.server.api.IScope scope,
String name,
org.red5.server.api.stream.IServerStream stream);
```

Puts a server stream in the stream map

### Parameters

scope	Scope of stream
name	Name of stream
stream	ServerStream object

## 5.5. removeServerStream(IScope, String)

```
public static void removeServerStream(org.red5.server.api.IScope scope,
String name);
```

Removes a server stream from the stream map

### Parameters

scope	Scope of stream
-------	-----------------

name	Name of stream

# 1. Class CacheImpl

Provides an implementation of an object cache.

## 1.1. Synopsis

```
public class CacheImpl implements, org.red5.server.api.cache.ICacheStore, org.springframework.context.ApplicationContext
// Protected Fields
protected static org.slf4j.Logger log;

// Public Static Methods
public static org.springframework.context.ApplicationContext getApplicationContext();
public static long getCacheHit();
public static long getCacheMiss();
public static CacheImpl getInstance();

// Public Methods
public void destroy();
public org.red5.server.api.cache.ICacheable get(String name);
public java.util.Iterator<java.lang.String> getObjectNames();
public java.util.Iterator<java.lang.ref.SoftReference<? extends org.red5.server.api.cache.ICacheable>> getSoftObjectNames();
public void init();
public boolean offer(String name,
                     Object obj);
public boolean offer(String key,
                     org.apache.mina.common.ByteBuffer obj);
public void put(String name,
                Object obj);
public boolean remove(String name);
public boolean remove(org.red5.server.api.cache.ICacheable obj);
public void setApplicationContext(org.springframework.context.ApplicationContext context)
throws BeansException;
public void setMaxEntries(int max);

// Protected Methods
protected void put(String name,
                  org.red5.server.api.cache.ICacheable obj);
}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 1.2. destroy()

```
public void destroy();
```

**Specified by:** Method destroy in interface ICacheStore

Allows for cleanup of a cache implementation.

## 1.3. get(String)

```
public org.red5.server.api.cache.ICacheable get(String name);
```

**Specified by:** Method get in interface ICacheStore

Return a cached object with the given name.

## 1.4. getApplicationContext()

```
public static org.springframework.context.ApplicationContext getApplicationContext();
```

Getter for property 'applicationContext'.

### Parameters

<i>return</i>	Value for property 'applicationContext'.
---------------	--

## 1.5. getCacheHit()

```
public static long getCacheHit();
```

Getter for property 'cacheHit'.

### Parameters

<i>return</i>	Value for property 'cacheHit'.
---------------	--------------------------------

## 1.6. getCacheMiss()

```
public static long getCacheMiss();
```

Getter for property 'cacheMiss'.

### Parameters

<i>return</i>	Value for property 'cacheMiss'.
---------------	---------------------------------

## 1.7. getInstance()

```
public static CacheImpl getInstance();
```

Returns the instance of this class.

#### Parameters

<i>return</i>
---------------

## 1.8. getObjectNames()

```
public java.util.Iterator<java.lang.String> getObjectNames();
```

**Specified by:** Method getObjectNames in interface ICacheStore

Return iterator over the names of all already loaded objects in the storage.

## 1.9. getObjects()

```
public java.util.Iterator<java.lang.ref.SoftReference<? extends org.red5.server.api.cache.ICacheable>> getObjects();
```

**Specified by:** Method getObjects in interface ICacheStore

Return iterator over the already loaded objects in the storage.

## 1.10. offer(String, Object)

```
public boolean offer(String name,  
                     Object obj);
```

**Specified by:** Method offer in interface ICacheStore

Offer an object to the cache with an associated key. If the named object exists in cache, it will not be accepted.

## 1.11. put(String, Object)

```
public void put(String name,  
                Object obj);
```

**Specified by:** Method put in interface ICacheStore

Puts an object in the cache with the associated key.

## 1.12. remove(ICacheable)

```
public boolean remove(org.red5.server.api.cache.ICacheable obj);
```

**Specified by:** Method remove in interface ICacheStore

Delete the passed cached object.

## 1.13. remove(String)

```
public boolean remove(String name);
```

**Specified by:** Method remove in interface ICacheStore

Delete the cached object with the given name.

## 1.14. setApplicationContext(ApplicationContext)

```
public void setApplicationContext(org.springframework.context.ApplicationContext context)
    throws BeansException;
```

**Specified by:** Method `setApplicationContext` in interface `ApplicationContextAware`

## 1.15. setMaxEntries(int)

```
public void setMaxEntries(int max);
```

**Specified by:** Method `setMaxEntries` in interface `ICacheStore`

Sets the maximum number of entries for the cache.

# 2. Class CacheableImpl

Provides an implementation of a cacheable object.

## 2.1. Synopsis

```
public class CacheableImpl implements org.?red5.?server.?api.?cache.?ICacheable {
    // Protected Fields

    protected org.springframework.context.ApplicationContext applicationContext ;

    protected static org.slf4j.Logger log ;

    // Public Constructors

    public CacheableImpl(Object obj);

    public CacheableImpl(org.apache.mina.common.ByteBuffer buffer);

    // Public Methods

    public void addRequest();

    public org.apache.mina.common.ByteBuffer getByteBuffer();

    public byte[] getBytes();

    public String getName();

    public boolean isCached();

    public void setCached(boolean cached);

    public void setName(String name);

}
```

**Methods inherited from java.lang.Object:** `clone` , `equals` , `finalize` , `getClass` , `hashCode` , `notify` , `notifyAll` , `toString` , `wait`

## 2.2. getByteBuffer()

```
public org.apache.mina.common.ByteBuffer getByteBuffer();
```

**Specified by:** Method getByteBuffer in interface ICacheable

Returns a readonly byte buffer.

## 2.3. getBytes()

```
public byte[] getBytes();
```

**Specified by:** Method getBytes in interface ICacheable

Returns the object contained within the cacheable reference.

## 2.4. getName()

```
public String getName();
```

**Specified by:** Method getName in interface ICacheable

Returns the name of the cached object.

## 2.5. isCached()

```
public boolean isCached();
```

**Specified by:** Method isCached in interface ICacheable

Returns `true` if the object is cached, `false` otherwise.

## 2.6. setCached(boolean)

```
public void setCached(boolean cached);
```

**Specified by:** Method setCached in interface ICacheable

Sets a flag to represent the cached status of a cacheable object.

## 2.7. setName(String)

```
public void setName(String name);
```

**Specified by:** Method setName in interface ICacheable

Set the name of the cached object.

## 3. Class EhCacheImpl

Provides an implementation of an object cache using EhCache.

### 3.1. Synopsis

```
public class EhCacheImpl implements org.?red5.?server.?api.?cache.?ICacheStore, org.?springframework...
```

## Package org.?red5.?server.?cache

```
// Protected Fields

    protected static org.slf4j.Logger log ;

// Public Constructors

    public EhCacheImpl();

// Public Static Methods

    public static org.springframework.context.ApplicationContext getApplicationContext();

    public static long getCacheHit();

    public static long getCacheMiss();

// Public Methods

    public void destroy();

    public org.red5.server.api.cache.ICacheable get(String name);

    public net.sf.ehcache.event.CacheManagerEventListener getCacheManagerEventListener();

    public int getDiskExpiryThreadIntervalSeconds();

    public String getDiskStore();

    public String getMemoryStoreEvictionPolicy();

    public java.util.Iterator<java.lang.String> getObjectNames();

    public java.util.Iterator<java.lang.ref.SoftReference<? extends org.red5.server.api.cache.ICacheab>
```

public void init();

```
public boolean offer(String name,
                      Object obj);
```

```
public void put(String name,
                 Object obj);
```

```
public boolean remove(String name);
```

```
public boolean remove(org.red5.server.api.cache.ICacheable obj);
```

```
public void setApplicationContext(org.springframework.context.ApplicationContext context)
throws BeansException;
```

```
public void setCacheConfigs(java.util.List<net.sf.ehcache.config.CacheConfiguration> configs);
```

```
public void setCacheManagerEventListener(net.sf.ehcache.event.CacheManagerEventListener cacheManag
```

```
public void setDiskExpiryThreadIntervalSeconds(int diskExpiryThreadIntervalSeconds);
```

```
public void setDiskStore(String diskStore);
```

```
public void setMaxEntries(int capacity);
```

```
public void setMemoryStoreEvictionPolicy(String memoryStoreEvictionPolicy);
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

See Also

ehcache homepage [<http://ehcache.sourceforge.net/>]

### 3.2. destroy()

```
public void destroy();
```

**Specified by:** Method destroy in interface ICacheStore

Allows for cleanup of a cache implementation.

### 3.3. get(String)

```
public org.red5.server.api.cache.ICacheable get(String name);
```

**Specified by:** Method get in interface ICacheStore

Return a cached object with the given name.

### 3.4. getApplicationContext()

```
public static org.springframework.context.ApplicationContext getApplicationContext();
```

Getter for property 'applicationContext'.

Parameters

return	Value for property 'applicationContext'.
--------	--

### 3.5. getCacheHit()

```
public static long getCacheHit();
```

Getter for property 'cacheHit'.

Parameters

return	Value for property 'cacheHit'.
--------	--------------------------------

### 3.6. getCacheManagerEventListener()

```
public net.sf.ehcache.event.CacheManagerEventListener getCacheManagerEventListener();
```

Getter for property 'cacheManagerEventListener'.

Parameters

return	Value for property 'cacheManagerEventListener'.
--------	---

### 3.7. getCacheMiss()

public static long getCacheMiss();
------------------------------------

Getter for property 'cacheMiss'.

Parameters
------------

return	Value for property 'cacheMiss'.
--------	---------------------------------

### 3.8. getDiskExpiryThreadIntervalSeconds()

public int getDiskExpiryThreadIntervalSeconds();
--

Getter for property 'diskExpiryThreadIntervalSeconds'.

Parameters
------------

return	Value for property 'diskExpiryThreadIntervalSeconds'.
--------	---

### 3.9. getDiskStore()

public String getDiskStore();
-------------------------------

Getter for property 'diskStore'.

Parameters
------------

return	Value for property 'diskStore'.
--------	---------------------------------

### 3.10. getMemoryStoreEvictionPolicy()

public String getMemoryStoreEvictionPolicy();
---

Getter for property 'memoryStoreEvictionPolicy'.

Parameters
------------

return	Value for property 'memoryStoreEvictionPolicy'.
--------	---

### 3.11. getObjectNames()

public java.util.Iterator<java.lang.String> getObjectNames();
---

**Specified by:** Method getObjectNames in interface ICacheStore

Return iterator over the names of all already loaded objects in the storage.

### 3.12. getObjects()

public java.util.Iterator<java.lang.ref.SoftReference<? extends org.red5.server.api.cache.ICacheable>> getObjects();
--

**Specified by:** Method `getObjects` in interface `ICacheStore`

Return iterator over the already loaded objects in the storage.

### 3.13. `offer(String, Object)`

```
public boolean offer(String name,
                     Object obj);
```

**Specified by:** Method `offer` in interface `ICacheStore`

Offer an object to the cache with an associated key. If the named object exists in cache, it will not be accepted.

### 3.14. `put(String, Object)`

```
public void put(String name,
                Object obj);
```

**Specified by:** Method `put` in interface `ICacheStore`

Puts an object in the cache with the associated key.

### 3.15. `remove(ICacheable)`

```
public boolean remove(org.red5.server.api.cache.ICacheable obj);
```

**Specified by:** Method `remove` in interface `ICacheStore`

Delete the passed cached object.

### 3.16. `remove(String)`

```
public boolean remove(String name);
```

**Specified by:** Method `remove` in interface `ICacheStore`

Delete the cached object with the given name.

### 3.17. `setApplicationContext(ApplicationContext)`

```
public void setApplicationContext(org.springframework.context.ApplicationContext context)
                                throws BeansException;
```

**Specified by:** Method `setApplicationContext` in interface `ApplicationContextAware`

### 3.18. `setCacheConfigs(List<CacheConfiguration>)`

```
public void setCacheConfigs(java.util.List<net.sf.ehcache.config.CacheConfiguration> configs);
```

Setter for property 'cacheConfigs'.

#### Parameters

configs	Value to set for property 'cacheConfigs'.
---------	---

### 3.19. setCacheManagerEventListener(CacheManagerEventListener)

```
public void setCacheManagerEventListener(net.sf.ehcache.event.CacheManagerEventListener cacheManagerEventListener);
```

Setter for property 'cacheManagerEventListener'.

#### Parameters

cacheManagerEventListener	Value to set for property 'cacheManagerEventListener'.
---------------------------	--

### 3.20. setDiskExpiryThreadIntervalSeconds(int)

```
public void setDiskExpiryThreadIntervalSeconds(int diskExpiryThreadIntervalSeconds);
```

Setter for property 'diskExpiryThreadIntervalSeconds'.

#### Parameters

diskExpiryThreadIntervalSeconds	Value to set for property 'diskExpiryThreadIntervalSeconds'.
---------------------------------	--

### 3.21. setDiskStore(String)

```
public void setDiskStore(String diskStore);
```

Setter for property 'diskStore'.

#### Parameters

diskStore	Value to set for property 'diskStore'.
-----------	--

### 3.22. setMaxEntries(int)

```
public void setMaxEntries(int capacity);
```

**Specified by:** Method setMaxEntries in interface ICacheStore

Sets the maximum number of entries for the cache.

### 3.23. setMemoryStoreEvictionPolicy(String)

```
public void setMemoryStoreEvictionPolicy(String memoryStoreEvictionPolicy);
```

Setter for property 'memoryStoreEvictionPolicy'.

#### Parameters

memoryStoreEvictionPolicy	Value to set for property 'memoryStoreEvictionPolicy'.
---------------------------	--

## 4. Class NoCacheImpl

Provides an implementation of an object cache which actually does not provide a cache.

## 4.1. Synopsis

```

public class NoCacheImpl implements, org.?red5.?server.?api.?cache.?ICacheStore, org.?springframework...
// Protected Fields

    protected static org.slf4j.Logger log ;

// Public Static Methods

    public static org.springframework.context.ApplicationContext getApplicationContext();

    public static long getCacheHit();

    public static long getCacheMiss();

    public static NoCacheImpl getInstance();

// Public Methods

    public void destroy();

    public org.red5.server.api.cache.ICacheable get(String name);

    public java.util.Iterator<java.lang.String> getObjectNames();

    public java.util.Iterator<java.lang.ref.SoftReference<? extends org.red5.server.api.cache.ICacheab...>> getEntries();

    public boolean offer(String name,
                        Object obj);

    public boolean offer(String key,
                        org.apache.mina.common.ByteBuffer obj);

    public void put(String name,
                  Object obj);

    public boolean remove(String name);

    public boolean remove(org.red5.server.api.cache.ICacheable obj);

    public void setApplicationContext(org.springframework.context.ApplicationContext context)
        throws BeansException;

    public void setMaxEntries(int max);
}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 4.2. destroy()

```
public void destroy();
```

**Specified by:** Method destroy in interface ICacheStore

Allows for cleanup of a cache implementation.

### 4.3. get(String)

```
public org.red5.server.api.cache.ICacheable get(String name);
```

**Specified by:** Method get in interface ICacheStore

Return a cached object with the given name.

### 4.4. getApplicationContext()

```
public static org.springframework.context.ApplicationContext getApplicationContext();
```

Getter for property 'applicationContext'.

#### Parameters

<i>return</i>	Value for property 'applicationContext'.
---------------	--

### 4.5. getCacheHit()

```
public static long getCacheHit();
```

Getter for property 'cacheHit'.

#### Parameters

<i>return</i>	Value for property 'cacheHit'.
---------------	--------------------------------

### 4.6. getCacheMiss()

```
public static long getCacheMiss();
```

Getter for property 'cacheMiss'.

#### Parameters

<i>return</i>	Value for property 'cacheMiss'.
---------------	---------------------------------

### 4.7. getInstance()

```
public static NoCacheImpl getInstance();
```

Returns the instance of this class.

#### Parameters

<i>return</i>
---------------

### 4.8. getObjectNames()

```
public java.util.Iterator<java.lang.String> getObjectNames();
```

**Specified by:** Method getObjectNames in interface ICacheStore

Return iterator over the names of all already loaded objects in the storage.

#### 4.9. **getObjects()**

```
public java.util.Iterator<java.lang.ref.SoftReference<? extends org.red5.server.api.cache.ICacheable>> getObjects();
```

**Specified by:** Method `getObjects` in interface `ICacheStore`

Return iterator over the already loaded objects in the storage.

#### 4.10. **offer(String, Object)**

```
public boolean offer(String name,  
                     Object obj);
```

**Specified by:** Method `offer` in interface `ICacheStore`

Offer an object to the cache with an associated key. If the named object exists in cache, it will not be accepted.

#### 4.11. **put(String, Object)**

```
public void put(String name,  
                Object obj);
```

**Specified by:** Method `put` in interface `ICacheStore`

Puts an object in the cache with the associated key.

#### 4.12. **remove(ICacheable)**

```
public boolean remove(org.red5.server.api.cache.ICacheable obj);
```

**Specified by:** Method `remove` in interface `ICacheStore`

Delete the passed cached object.

#### 4.13. **remove(String)**

```
public boolean remove(String name);
```

**Specified by:** Method `remove` in interface `ICacheStore`

Delete the cached object with the given name.

#### 4.14. **setApplicationContext(ApplicationContext)**

```
public void setApplicationContext(org.springframework.context.ApplicationContext context)  
throws BeansException;
```

**Specified by:** Method `setApplicationContext` in interface `ApplicationContextAware`

#### 4.15. **setMaxEntries(int)**

```
public void setMaxEntries(int max);
```

**Specified by:** Method `setMaxEntries` in interface `ICacheStore`

Sets the maximum number of entries for the cache.

# 1. Exception AccessDeniedException

Access denied

## 1.1. Synopsis

```
public class AccessDeniedException extends java.lang.RuntimeException {  
    // Public Constructors  
  
    public AccessDeniedException();  
  
}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

# 2. Exception ClientDetailsException

Exception class than contains additional parameters to return to the client.

## 2.1. Synopsis

```
public class ClientDetailsException extends java.lang.RuntimeException {  
    // Public Constructors  
  
    public ClientDetailsException(String message,  
                                Object params);  
  
    public ClientDetailsException(String message,  
                                Object params,  
                                boolean includeStacktrace);  
  
    // Public Methods  
  
    public Object getParameters();  
  
    public boolean includeStacktrace();  
  
}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 2.2. ClientDetailsException(String, Object)

```
public ClientDetailsException(String message,
                             Object params);
```

Create new exception object from message and parameters. By default, no stacktrace is returned to the client.

### Parameters

message	
params	

## 2.3. ClientDetailsException(String, Object, boolean)

```
public ClientDetailsException(String message,
                             Object params,
                             boolean includeStacktrace);
```

Create new exception object from message and parameters with optional stacktrace.

### Parameters

message	
params	
includeStacktrace	

## 2.4. getParameters()

```
public Object getParameters();
```

Get parameters to return to the client.

### Parameters

<i>return</i>	
---------------	--

## 2.5. includeStacktrace()

```
public boolean includeStacktrace();
```

Should the stacktrace returned to the client?

### Parameters

<i>return</i>	
---------------	--

## 3. Exception ClientNotFoundException

Client not found

### 3.1. Synopsis

```
public class ClientNotFoundException extends java.lang.RuntimeException {
// Public Constructors

    public ClientNotFoundException(String id);

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

### 3.2. ClientNotFoundException(String)

```
public ClientNotFoundException(String id);
```

Create exception from given string message

#### Parameters

id
----

## 4. Exception ClientRejectedException

The client is not allowed to connect. Reason that provided with this exception is sent to client-side status event description.

### 4.1. Synopsis

```
public class ClientRejectedException extends java.lang.RuntimeException {
// Public Constructors

    public ClientRejectedException();

    public ClientRejectedException(Object reason);

// Public Methods

    public Object getReason();

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 4.2. ClientRejectedException()

```
public ClientRejectedException();
```

Constructs a new ClientRejectedException.

## 4.3. ClientRejectedException(Object)

```
public ClientRejectedException(Object reason);
```

Create new exception with given rejection reason

### Parameters

reason	Rejection reason
--------	------------------

## 4.4. getReason()

```
public Object getReason();
```

Getter for reason

### Parameters

return	Rejection reason
--------	------------------

## 5. Exception ScopeHandlerNotFoundException

Scope handler not found. Thrown when scope handler with given name can't be found

### 5.1. Synopsis

```
public class ScopeHandlerNotFoundException extends java.lang.RuntimeException {
// Public Constructors

    public ScopeHandlerNotFoundException(String handlerName);

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 5.2. ScopeHandlerNotFoundException(String)

```
public ScopeHandlerNotFoundException(String handlerName);
```

Create exception from given scope handler name

#### Parameters

handlerName	Scope handler name
-------------	--------------------

## 6. Exception ScopeNotFoundException

Scope not found, thrown when child scope wasn't found.

### 6.1. Synopsis

```
public class ScopeNotFoundException extends java.lang.RuntimeException {
// Public Constructors

    public ScopeNotFoundException(org.red5.server.api.IScope scope,
                                String childName);

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

### 6.2. ScopeNotFoundException(IScope, String)

```
public ScopeNotFoundException(org.red5.server.api.IScope scope,
                            String childName);
```

Create exception from given scope object and given child subscope

#### Parameters

scope	Scope
childName	Subscope name

## 7. Exception ScopeShuttingDownException

Scope is currently shutting down.

### 7.1. Synopsis

```
public class ScopeShuttingDownException extends java.lang.RuntimeException {
// Public Constructors

    public ScopeShuttingDownException(org.red5.server.api.IScope scope);

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 7.2. ScopeShuttingDownException(IScope)

```
public ScopeShuttingDownException(org.red5.server.api.IScope scope);
```

Create exception from given scope object

### Parameters

scope	Scope
-------	-------

## 8. Exception ServiceNotFoundException

Thrown when server can't be found

### 8.1. Synopsis

```
public class ServiceNotFoundException extends java.lang.RuntimeException {
// Public Constructors

    public ServiceNotFoundException();
}

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 9. Exception SharedObjectException

Base for all shared object-related exceptions

### 9.1. Synopsis

```
public class SharedObjectException extends java.lang.RuntimeException {
// Public Constructors

    public SharedObjectException();
}

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause ,  
getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace ,  
setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , wait

## 10. Exception StreamControlException

Stream control exception

### 10.1. Synopsis

```
public class StreamControlException extends, java.?lang.?RuntimeException {
// Public Constructors

    public StreamControlException();

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause ,  
getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace ,  
setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , wait

## 11. Exception StreamDataException

Thrown on stream data exception

### 11.1. Synopsis

```
public class StreamDataException extends, java.?lang.?RuntimeException {
// Public Constructors

    public StreamDataException();

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause ,  
getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace ,  
setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , wait

# 1. Class JettyApplicationContext

Class that wraps a Jetty webapp context.

## 1.1. Synopsis

```
public class JettyApplicationContext implements org.red5.server.api.IApplicationContext {  
    // Protected Fields  
  
    protected static org.slf4j.Logger log ;  
  
    // Protected Constructors  
  
    protected JettyApplicationContext(org.mortbay.jetty.webapp.WebAppContext context);  
  
    // Public Methods  
  
    public void stop();  
  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. JettyApplicationContext(WebAppContext)

```
protected JettyApplicationContext(org.mortbay.jetty.webapp.WebAppContext context);
```

Wrap the passed Jetty webapp context.

### Parameters

context
---------

## 1.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 1.4. stop()

```
public void stop();
```

**Specified by:** Method stop in interface IApplicationContext

Stop the web application.

# 2. Class JettyApplicationLoader

Class that can load new applications in Jetty.

## 2.1. Synopsis

```

public class JettyApplicationLoader implements org.red5.server.api.IApplicationLoader {
    // Protected Fields

    protected static org.slf4j.Logger log;

    // Protected Constructors

    protected JettyApplicationLoader(org.mortbay.jetty.Server server,
                                    org.springframework.context.ApplicationContext rootCtx);

    // Public Methods

    public org.springframework.context.ApplicationContext getRootContext();

    public void loadApplication(String contextPath,
                               String virtualHosts,
                               String directory)
        throws Exception;
}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. JettyApplicationLoader(Server, ApplicationContext)

```
protected JettyApplicationLoader(org.mortbay.jetty.Server server,
                                org.springframework.context.ApplicationContext rootCtx);
```

Create new application loader for Jetty servers.

### Parameters

server
--------

## 2.3. log

```
protected static org.slf4j.Logger log;
```

Logger

## 2.4. getRootContext()

```
public org.springframework.context.ApplicationContext getRootContext();
```

**Specified by:** Method getRootContext in interface IApplicationLoader

Return the root org.springframework.context.ApplicationContext.

## 2.5. loadApplication(String, String, String)

```
public void loadApplication(String contextPath,
                           String virtualHosts,
                           String directory)
```

```
throws Exception;
```

**Specified by:** Method loadApplication in interface IApplicationLoader

Load a new application for the given context path from a directory.

## 3. Class JettyLoader

Class that loads Red5 applications using Jetty.

### 3.1. Synopsis

```
public class JettyLoader extends, org.?red5.?server.?LoaderBase
    implements, org.?red5.?server.?LoaderMBean {
// Protected Fields

    protected String defaultWebConfig ;

    protected org.mortbay.jetty.Server jetty ;

    protected String jettyConfig ;

    protected static org.slf4j.Logger log ;

// Public Constructors

    public JettyLoader();

// Public Methods

    public void init();

    public void registerJMX();

    public void removeContext(String path);

    public void shutdown();

}
```

**Methods inherited from org.red5.server.LoaderBase:** getApplicationContext ,  
getApplicationLoader , getRed5ApplicationContext , removeContext ,  
removeRed5ApplicationContext , setApplicationContext , setApplicationLoader ,  
setRed5ApplicationContext , setWebappFolder

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.LoaderBase:** applicationContext , loader ,  
red5AppCtx , webappFolder

### 3.2. defaultWebConfig

```
protected String defaultWebConfig ;
```

Default web config filename

### 3.3. jetty

```
protected org.mortbay.jetty.Server jetty ;
```

I Server implementation

### 3.4. jettyConfig

```
protected String jettyConfig ;
```

Jetty config path

### 3.5. log

```
protected static org.slf4j.Logger log ;
```

Logger

### 3.6. removeContext(String)

```
public void removeContext(String path);
```

**Specified by:** Method removeContext in interface LoaderMBean

Remove context from the current host.

#### Parameters

path	Path
------	------

### 3.7. shutdown()

```
public void shutdown();
```

**Specified by:** Method shutdown in interface LoaderMBean

Shut server down

## 4. Class Red5WebPropertiesConfiguration

Red web properties configuration

### 4.1. Synopsis

```
public class Red5WebPropertiesConfiguration implements, org.?mortbay.?jetty.?webapp.?Configuration, ja
// Protected Fields

    protected org.mortbay.jetty.webapp.WebAppContext _context ;

    protected static org.slf4j.Logger log ;

// Public Constructors
```

```

public Red5WebPropertiesConfiguration();

// Public Methods

public void configureClassLoader()
throws Exception;

public void configureDefaults()
throws Exception;

public void configureWebApp()
throws Exception;

public void deconfigureWebApp()
throws Exception;

public org.mortbay.jetty.webapp.WebAppContext getWebAppContext();

public void setWebAppContext(org.mortbay.jetty.webapp.WebAppContext context);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

## 4.2. \_context

```
protected org.mortbay.jetty.webapp.WebAppContext _context ;
```

Web application context

## 4.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 4.4. configureClassLoader()

```

public void configureClassLoader()
throws Exception;

```

**Specified by:** Method configureClassLoader in interface Configuration

## 4.5. configureDefaults()

```

public void configureDefaults()
throws Exception;

```

**Specified by:** Method configureDefaults in interface Configuration

## 4.6. configureWebApp()

```
public void configureWebApp()
```

```
throws Exception;
```

**Specified by:** Method configureWebApp in interface Configuration

## 4.7. deconfigureWebApp()

```
public void deconfigureWebApp()
    throws Exception;
```

**Specified by:** Method deconfigureWebApp in interface Configuration

## 4.8. getWebAppContext()

```
public org.mortbay.jetty.webapp.WebAppContext getWebAppContext();
```

**Specified by:** Method getWebAppContext in interface Configuration

## 4.9. setWebAppContext(WebAppContext)

```
public void setWebAppContext(org.mortbay.jetty.webapp.WebAppContext context);
```

**Specified by:** Method setWebAppContext in interface Configuration

# 1. Class JMXAgent

Provides the connection adapters as well as registration and unregistration of MBeans.

## 1.1. Synopsis

```
public class JMXAgent implements javax.management.NotificationListener {  
    // Public Constructors  
  
    public JMXAgent();  
  
    // Public Static Methods  
  
    public static boolean isEnableMinaMonitor();  
  
    public static boolean registerMBean(Object instance,  
                                       String className,  
                                       Class interfaceClass);  
  
    public static boolean registerMBean(Object instance,  
                                       String className,  
                                       Class interfaceClass,  
                                       String name);  
  
    public static boolean registerMBean(Object instance,  
                                       String className,  
                                       Class interfaceClass,  
                                       javax.management.ObjectName name);  
  
    public static void shutdown();  
  
    public static String trimClassName(String className);  
  
    public static boolean unregisterMBean(javax.management.ObjectName oName);  
  
    public static boolean updateMBeanAttribute(javax.management.ObjectName oName,  
                                              String key,  
                                              int value);  
  
    public static boolean updateMBeanAttribute(javax.management.ObjectName oName,  
                                              String key,  
                                              String value);  
  
    // Public Methods  
  
    public String getHtmlAdapterPort();  
  
    public void handleNotification(javax.management.Notification notification,  
                                 Object handback);  
  
    public void init();  
  
    public void setEnableHtmlAdapter(boolean enableHtmlAdapter);  
  
    public void setEnableHtmlAdapter(String enableHtmlAdapterString);  
  
    public void setEnableMinaMonitor(boolean enableMinaMonitor);
```

```

    public void setEnableMinaMonitor(String enableMinaMonitor);

    public void setEnableRmiAdapter(boolean enableRmiAdapter);

    public void setEnableRmiAdapter(String enableRmiAdapterString);

    public void setEnableSsl(boolean enableSsl);

    public void setEnableSsl(String enableSslString);

    public void setHtmlAdapterPort(String htmlAdapterPort);

    public void setRemoteAccessProperties(String remoteAccessProperties);

    public void setRemotePasswordProperties(String remotePasswordProperties);

    public void setRemoteSSLKeystore(String remoteSSLKeystore);

    public void setRemoteSSLKeystorePass(String remoteSSLKeystorePass);

    public void setRmiAdapterHost(String rmiAdapterHost);

    public void setRmiAdapterPort(String rmiAdapterPort);

    public void setRmiAdapterRemotePort(String rmiAdapterRemotePort);

    public void setStartRegistry(boolean startRegistry);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. shutdown()

```
public static void shutdown();
```

Shuts down any instanced connectors.

## 1.3. trimClassName(String)

```
public static String trimClassName(String className);
```

Convenience to remove packages etc from a class name.

### Parameters

className	
-----------	--

return	
--------	--

## 1.4. unregisterMBean(ObjectName)

```
public static boolean unregisterMBean(javax.management.ObjectName oName);
```

Unregisters an mbean instance. If the instance is not found or if a failure occurs, false will be returned.

#### Parameters

oName	
<i>return</i>	

## 1.5. updateMBeanAttribute(ObjectName, String, int)

```
public static boolean updateMBeanAttribute(javax.management.ObjectName oName,
                                         String key,
                                         int value);
```

Updates a named attribute of a registered mbean.

#### Parameters

oName	
key	
value	
<i>return</i>	

## 1.6. updateMBeanAttribute(ObjectName, String, String)

```
public static boolean updateMBeanAttribute(javax.management.ObjectName oName,
                                         String key,
                                         String value);
```

Updates a named attribute of a registered mbean.

#### Parameters

oName	
key	
value	
<i>return</i>	

## 2. Class JMXFactory

Provides access to the MBeanServer as well as registration and creation of new MBean instances. For most classes the creation and registration is performed using StandardMBean wrappers.

References: <http://www.onjava.com/pub/a/onjava/2004/09/29/tigerjmx.html?page=1> <http://java.sun.com/developer/JDCTechTips/2005/tt0315.html#2>

Examples: <http://java.sun.com/javase/6/docs/technotes/guides/jmx/examples.html>

## 2.1. Synopsis

```

public class JMXFactory {
    // Public Constructors

    public JMXFactory();

    // Public Static Methods

    public static javax.management.ObjectName createMBean(String className,
                                                          String attributes);

    public static javax.management.ObjectName createObjectName(String[] strings);

    public static javax.management.ObjectName createSimpleMBean(String className,
                                                               String objectNameStr);

    public static String getDefaultDomain();

    public static javax.management.MBeanServer getMBeanServer();

    public static boolean registerNewMBean(String className,
                                          Class interfaceClass);

    public static boolean registerNewMBean(String className,
                                          Class interfaceClass,
                                          String name);

    public static boolean registerNewMBean(String className,
                                          Class interfaceClass,
                                          javax.management.ObjectName name);

    // Public Methods

    public String getDomain();

    public void setDomain(String domain);
}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 3. Class JMXUtil

Helper methods for working with ObjectName or MBean instances.

### 3.1. Synopsis

```

public class JMXUtil {
    // Public Constructors

    public JMXUtil();

    // Public Static Methods

    public static void printMBeanInfo(javax.management.ObjectName objectName,

```

```
    String className);  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

# 1. Class AbstractMessage

Abstract base for all messages

## 1.1. Synopsis

```
public class AbstractMessage implements org.red5.server.messaging.IMessage {  
    // Protected Fields  
  
    protected String correlationID ;  
  
    protected java.util.Map extraHeaders ;  
  
    protected String messageID ;  
  
    protected String messageType ;  
  
    // Public Constructors  
  
    public AbstractMessage();  
  
    // Public Methods  
  
    public boolean getBooleanProperty(String name);  
  
    public byte getByteProperty(String name);  
  
    public String getCorrelationID();  
  
    public double getDoubleProperty(String name);  
  
    public float getFloatProperty(String name);  
  
    public int getIntProperty(String name);  
  
    public long getLongProperty(String name);  
  
    public String getMessageID();  
  
    public String getMessageType();  
  
    public Object getObjectProperty(String name);  
  
    public short getShortProperty(String name);  
  
    public String getStringProperty(String name);  
  
    public void setBooleanProperty(String name,  
                                  boolean value);  
  
    public void setByteProperty(String name,  
                               byte value);  
  
    public void setCorrelationID(String id);  
  
    public void setDoubleProperty(String name,
```

```

        double value);

public void setFloatProperty(String name,
                             float value);

public void setIntProperty(String name,
                           int value);

public void setLongProperty(String name,
                            long value);

public void setMessageID(String id);

public void setMessageType(String type);

public void setObjectProperty(String name,
                             Object value);

public void setShortProperty(String name,
                            short value);

public void setStringProperty(String name,
                             String value);

}

}

```

**Direct known subclasses:** org.?red5.?server.?stream.?message.?RTMPMessage , org.?red5.?server.?stream.?message.?ResetMessage , org.?red5.?server.?stream.?message.?StatusMessage

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

#### See Also

`org.red5.server.messaging.IMessage`

## 1.2. correlationID

```
protected String correlationID ;
```

## 1.3. extraHeaders

```
protected java.util.Map extraHeaders ;
```

## 1.4. messageID

```
protected String messageID ;
```

## 1.5. messageType

```
protected String messageType ;
```

## 1.6. getBooleanProperty(String)

```
public boolean getBooleanProperty(String name);
```

**Specified by:** Method getBooleanProperty in interface IMessage

Getter for boolean property

## 1.7. getByteProperty(String)

```
public byte getByteProperty(String name);
```

**Specified by:** Method getByteProperty in interface IMessage

Add byte property to message

## 1.8. getCorrelationID()

```
public String getCorrelationID();
```

**Specified by:** Method getCorrelationID in interface IMessage

Return correlation id

## 1.9. getDoubleProperty(String)

```
public double getDoubleProperty(String name);
```

**Specified by:** Method getDoubleProperty in interface IMessage

Return double property by name

## 1.10. getFloatProperty(String)

```
public float getFloatProperty(String name);
```

**Specified by:** Method getFloatProperty in interface IMessage

Return float property by name

## 1.11. getIntProperty(String)

```
public int getIntProperty(String name);
```

**Specified by:** Method getIntProperty in interface IMessage

Return int property by name

## 1.12. getLongProperty(String)

```
public long getLongProperty(String name);
```

**Specified by:** Method getLongProperty in interface IMessage

Return long property to message

### 1.13. getMessageID()

```
public String getMessageID();
```

**Specified by:** Method getMessageID in interface IMensaje

Return message id

### 1.14. getMessageType()

```
public String getMessageType();
```

**Specified by:** Method getMessageType in interface IMensaje

Return message type

### 1.15. getObjectProperty(String)

```
public Object getObjectProperty(String name);
```

**Specified by:** Method getObjectProperty in interface IMensaje

Return object property to message

### 1.16. getShortProperty(String)

```
public short getShortProperty(String name);
```

**Specified by:** Method getShortProperty in interface IMensaje

Return short property to message

### 1.17. getStringProperty(String)

```
public String getStringProperty(String name);
```

**Specified by:** Method getStringProperty in interface IMensaje

Return string property to message

### 1.18. setBooleanProperty(String, boolean)

```
public void setBooleanProperty(String name,  
                                boolean value);
```

**Specified by:** Method setBooleanProperty in interface IMensaje

Add boolean property to message

### 1.19. setByteProperty(String, byte)

```
public void setByteProperty(String name,
```

```
byte value);
```

**Specified by:** Method setByteProperty in interface IMessage

Add byte property to message

## 1.20. setCorrelationID(String)

```
public void setCorrelationID(String id);
```

**Specified by:** Method setCorrelationID in interface IMessage

Setter for correlation id

## 1.21. setDoubleProperty(String, double)

```
public void setDoubleProperty(String name,  
                             double value);
```

**Specified by:** Method setDoubleProperty in interface IMessage

Add double property to message

## 1.22. setFloatProperty(String, float)

```
public void setFloatProperty(String name,  
                            float value);
```

**Specified by:** Method setFloatProperty in interface IMessage

Add float property to message

## 1.23. setIntProperty(String, int)

```
public void setIntProperty(String name,  
                           int value);
```

**Specified by:** Method setIntProperty in interface IMessage

Add int property to message

## 1.24. setLongProperty(String, long)

```
public void setLongProperty(String name,  
                           long value);
```

**Specified by:** Method setLongProperty in interface IMessage

Add long property to message

## 1.25. setMessageID(String)

```
public void setMessageID(String id);
```

**Specified by:** Method setMessageID in interface IMessage

Setter for new message id

## 1.26. setMessageType(String)

```
public void setMessageType(String type);
```

**Specified by:** Method setMessageType in interface IMESSAGE

Setter for message type

## 1.27. setObjectProperty(String, Object)

```
public void setObjectProperty(String name,
                             Object value);
```

**Specified by:** Method setObjectProperty in interface IMESSAGE

Add object property to message

## 1.28. setShortProperty(String, short)

```
public void setShortProperty(String name,
                            short value);
```

**Specified by:** Method setShortProperty in interface IMESSAGE

Add short property to message

## 1.29. setStringProperty(String, String)

```
public void setStringProperty(String name,
                           String value);
```

**Specified by:** Method setStringProperty in interface IMESSAGE

Add string property to message

# 2. Class AbstractPipe

Abstract pipe that books providers/consumers and listeners. Aim to ease the implementation of concrete pipes. For more information on what pipe is, see IPipe interface documentation.

## 2.1. Synopsis

```
public abstract class AbstractPipe implements, org.?red5.?server.?messaging.?IPipe {
    // Protected Fields

    protected volatile java.util.List<org.red5.server.messaging.IConsumer> consumers ;
    protected volatile java.util.List<org.red5.server.messaging.IPipeConnectionListener> listeners ;
    protected volatile java.util.List<org.red5.server.messaging.IProvider> providers ;

    // Public Constructors
```

## Package org.?red5.?server.?messaging

```
public AbstractPipe();

// Public Methods

public void addPipeConnectionListener(IPipeConnectionListener listener);

public java.util.List<org.red5.server.messaging.IConsumer> getConsumers();

public java.util.List<org.red5.server.messaging.IPipeConnectionListener> getListeners();

public java.util.List<org.red5.server.messaging.IProvider> getProviders();

public void removePipeConnectionListener(IPipeConnectionListener listener);

public void sendOOBControlMessage(IConsumer consumer,
                                  OOBControlMessage oobCtrlMsg);

public void sendOOBControlMessage(IProvider provider,
                                  OOBControlMessage oobCtrlMsg);

public void setListeners(java.util.List<org.red5.server.messaging.IPipeConnectionListener> listeners);

public boolean subscribe(IConsumer consumer,
                        java.util.Map paramMap);

public boolean subscribe(IProvider provider,
                        java.util.Map paramMap);

public boolean unsubscribe(IConsumer consumer);

public boolean unsubscribe(IProvider provider);

// Protected Methods

protected void fireConsumerConnectionEvent(IConsumer consumer,
                                           int type,
                                           java.util.Map paramMap);

protected void firePipeConnectionEvent(PipeConnectionEvent event);

protected void fireProviderConnectionEvent(IProvider provider,
                                           int type,
                                           java.util.Map paramMap);

}
```

**Direct known subclasses:** org.?red5.?server.?messaging.?InMemoryPullPullPipe , org.?red5.?server.?messaging.?InMemoryPushPushPipe

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

### See Also

[org.red5.server.messaging.IPipe](#)

## 2.2. consumers

```
protected volatile java.util.List<org.red5.server.messaging.IConsumer> consumers ;
```

Pipe consumers list

## 2.3. listeners

```
protected volatile java.util.List<org.red5.server.messaging.IPipeConnectionListener> listeners ;
```

Event listeners

## 2.4. providers

```
protected volatile java.util.List<org.red5.server.messaging.IProvider> providers ;
```

Pipe providers list

## 2.5. addPipeConnectionListener(IPipeConnectionListener)

```
public void addPipeConnectionListener(IPipeConnectionListener listener);
```

**Specified by:** Method addPipeConnectionListener in interface IPipe

Registers pipe connect events listener

### Parameters

listener	Listener
----------	----------

## 2.6. fireConsumerConnectionEvent(IConsumer, int, Map)

```
protected void fireConsumerConnectionEvent(IConsumer consumer,
                                         int type,
                                         java.util.Map paramMap);
```

Broadcast consumer connection event

### Parameters

consumer	Consumer that has connected
type	Event type
paramMap	Parameters passed with connection

## 2.7. firePipeConnectionEvent(PipeConnectionEvent)

```
protected void firePipeConnectionEvent(PipeConnectionEvent event);
```

Fire any pipe connection event and run all it's tasks

### Parameters

event	Pipe connection event
-------	-----------------------

## 2.8. fireProviderConnectionEvent(IProvider, int, Map)

```
protected void fireProviderConnectionEvent(IProvider provider,
                                         int type,
                                         java.util.Map paramMap);
```

Broadcast provider connection event

### Parameters

provider	Provider that has connected
type	Event type
paramMap	Parameters passed with connection

## 2.9. getConsumers()

```
public java.util.List<org.red5.server.messaging.IConsumer> getConsumers();
```

Getter for consumers

### Parameters

<i>return</i>	consumers list
---------------	----------------

## 2.10. getListeners()

```
public java.util.List<org.red5.server.messaging.IPipeConnectionListener> getListeners();
```

Getter for pipe connection events listeners

### Parameters

<i>return</i>	Listeners
---------------	-----------

## 2.11. getProviders()

```
public java.util.List<org.red5.server.messaging.IProvider> getProviders();
```

Getter for providers

### Parameters

<i>return</i>	Providers list
---------------	----------------

## 2.12. removePipeConnectionListener(IPipeConnectionListener)

```
public void removePipeConnectionListener(IPipeConnectionListener listener);
```

**Specified by:** Method removePipeConnectionListener in interface IPipe

Removes pipe connection listener

## Parameters

listener	Listener
----------	----------

**2.13. sendOOBControlMessage(IConsumer, OOBControlMessage)**

```
public void sendOOBControlMessage(IConsumer consumer,
                                  OOBControlMessage oobCtrlMsg);
```

Send out-of-band ("special") control message to all providers

## Parameters

consumer	Consumer, may be used in concrete implementations
oobCtrlMsg	Out-of-band control message

**2.14. sendOOBControlMessage(IProvider, OOBControlMessage)**

```
public void sendOOBControlMessage(IProvider provider,
                                  OOBControlMessage oobCtrlMsg);
```

Send out-of-band ("special") control message to all consumers

## Parameters

provider	Provider, may be used in concrete implementations
oobCtrlMsg	Out-of-band control message

**2.15. setListeners(List<IPipeConnectionListener>)**

```
public void setListeners(java.util.List<org.red5.server.messaging.IPipeConnectionListener> listeners)
```

Setter for pipe connection events listeners

## Parameters

listeners	Listeners
-----------	-----------

**2.16. subscribe(IConsumer, Map)**

```
public boolean subscribe(IConsumer consumer,
                        java.util.Map paramMap);
```

Connect consumer to this pipe. Doesn't allow to connect one consumer twice. Does register event listeners if instance of IPipeConnectionListener is given.

## Parameters

consumer	Consumer
paramMap	Parameters passed with connection, used in concrete pipe implementations

<i>return</i>	true if consumer was added, false otherwise
---------------	---

## 2.17. subscribe(IProvider, Map)

```
public boolean subscribe(IProvider provider,
                        java.util.Map paramMap);
```

Connect provider to this pipe. Doesn't allow to connect one provider twice. Does register event listeners if instance of IPipeConnectionListener is given.

### Parameters

provider	Provider
paramMap	Parameters passed with connection, used in concrete pipe implementations
<i>return</i>	true if provider was added, false otherwise

## 2.18. unsubscribe(IConsumer)

```
public boolean unsubscribe(IConsumer consumer);
```

Disconnects consumer from this pipe. Fires pipe connection event.

### Parameters

consumer	Consumer that should be removed
<i>return</i>	true on success, false otherwise

## 2.19. unsubscribe(IProvider)

```
public boolean unsubscribe(IProvider provider);
```

Disconnects provider from this pipe. Fires pipe connection event.

### Parameters

provider	Provider that should be removed
<i>return</i>	true on success, false otherwise

## 3. Interface IConsumer

Signature for the message consumer.

### 3.1. Synopsis

```
public interface IConsumer extends, org.?red5.?server.?messaging.?IMessageComponent { }
```

## 4. Interface IFilter

Filter marker interface groups consumer and provider interfaces

### 4.1. Synopsis

```
public interface IFilter extends, org.?red5.?server.?messaging.?IConsumer, org.?red5.?server.?messag
```

## 5. Interface IMessage

Common interface for all messages.

Structure of messages is designed according to JMS Message interface. Message is composed of header and body. Header contains commonly used pre-defined headers and extensible headers.

Each message has correlation ID that is never used so far and is subject to be removed in future.

Message has type and number of properties.

### 5.1. Synopsis

```
public interface IMessage {
// Public Methods

    public boolean getBooleanProperty(String name);

    public byte getByteProperty(String name);

    public String getCorrelationID();

    public double getDoubleProperty(String name);

    public float getFloatProperty(String name);

    public int getIntProperty(String name);

    public long getLongProperty(String name);

    public String getMessageID();

    public String getMessageType();

    public Object getObjectProperty(String name);

    public short getShortProperty(String name);

    public String getStringProperty(String name);

    public void setBooleanProperty(String name,
```

```

        boolean value);

public void setByteProperty(String name,
                           byte value);

public void setCorrelationID(String id);

public void setDoubleProperty(String name,
                           double value);

public void setFloatProperty(String name,
                           float value);

public void setIntProperty(String name,
                           int value);

public void setLongProperty(String name,
                           long value);

public void setMessageID(String id);

public void setMessageType(String type);

public void setObjectProperty(String name,
                           Object value);

public void setShortProperty(String name,
                           short value);

public void setStringProperty(String name,
                           String value);

}

```

## 5.2. getBooleanProperty(String)

```
public boolean getBooleanProperty(String name);
```

Getter for boolean property

### Parameters

<i>name</i>	Boolean property name
<i>return</i>	Boolean property

## 5.3. getByteProperty(String)

```
public byte getByteProperty(String name);
```

Add byte property to message

### Parameters

<i>name</i>	Byte property name
-------------	--------------------

<i>return</i>	Byte property value
---------------	---------------------

## 5.4. getCorrelationID()

```
public String getCorrelationID();
```

Return correlation id

### Parameters

<i>return</i>	Correlation id
---------------	----------------

## 5.5. getDoubleProperty(String)

```
public double getDoubleProperty(String name);
```

Return double property by name

### Parameters

name	Double property name
<i>return</i>	Double property value

## 5.6. getFloatProperty(String)

```
public float getFloatProperty(String name);
```

Return float property by name

### Parameters

name	Float property name
<i>return</i>	Float property value

## 5.7. getIntProperty(String)

```
public int getIntProperty(String name);
```

Return int property by name

### Parameters

name	Int property name
<i>return</i>	Int property value

## 5.8. getLongProperty(String)

```
public long getLongProperty(String name);
```

Return long property to message

**Parameters**

<i>name</i>	Long property name
<i>return</i>	Long property value

**5.9. getMessageID()**

```
public String getMessageID();
```

Return message id

**Parameters**

<i>return</i>	Message id
---------------	------------

**5.10. getMessageType()**

```
public String getMessageType();
```

Return message type

**Parameters**

<i>return</i>	Message type
---------------	--------------

**5.11. getObjectProperty(String)**

```
public Object getObjectProperty(String name);
```

Return object property to message

**Parameters**

<i>name</i>	Object property name
<i>return</i>	Object property value

**5.12. getShortProperty(String)**

```
public short getShortProperty(String name);
```

Return short property to message

**Parameters**

<i>name</i>	Short property name
<i>return</i>	Short property value

**5.13. getStringProperty(String)**

```
public String getStringProperty(String name);
```

Return string property to message

Parameters	
name	String property name
<i>return</i>	String property value

## 5.14. setBooleanProperty(String, boolean)

```
public void setBooleanProperty(String name,
                               boolean value);
```

Add boolean property to message

Parameters	
name	Boolean property name
value	Boolean property value

## 5.15. setByteProperty(String, byte)

```
public void setByteProperty(String name,
                           byte value);
```

Add byte property to message

Parameters	
name	Byte property name
value	Byte property value

## 5.16. setCorrelationID(String)

```
public void setCorrelationID(String id);
```

Setter for correlation id

Parameters	
id	Correlation id

## 5.17. setDoubleProperty(String, double)

```
public void setDoubleProperty(String name,
                            double value);
```

Add double property to message

Parameters	
name	Double property name

value	Double property value
-------	-----------------------

## 5.18. setFloatProperty(String, float)

```
public void setFloatProperty(String name,
                           float value);
```

Add float property to message

### Parameters

name	Float property name
value	Float property value

## 5.19. setIntProperty(String, int)

```
public void setIntProperty(String name,
                           int value);
```

Add int property to message

### Parameters

name	Int property name
value	Int property value

## 5.20. setLongProperty(String, long)

```
public void setLongProperty(String name,
                           long value);
```

Add long property to message

### Parameters

name	Long property name
value	Long property value

## 5.21. setMessageID(String)

```
public void setMessageID(String id);
```

Setter for new message id

### Parameters

id	Message id
----	------------

## 5.22. setMessageType(String)

```
public void setMessageType(String type);
```

Setter for message type

#### Parameters

type	Message type
------	--------------

### 5.23. setObjectProperty(String, Object)

```
public void setObjectProperty(String name,
                             Object value);
```

Add object property to message

#### Parameters

name	Object property name
value	Object property value

### 5.24. setShortProperty(String, short)

```
public void setShortProperty(String name,
                            short value);
```

Add short property to message

#### Parameters

name	Short property name
value	Short property value

### 5.25. setStringProperty(String, String)

```
public void setStringProperty(String name,
                           String value);
```

Add string property to message

#### Parameters

name	String property name
value	String property value

## 6. Interface IMessageComponent

Message component handles out-of-band control messages

### 6.1. Synopsis

```
public interface IMessageComponent {
    // Public Methods

    public void onOOBControlMessage(IMessageComponent source,
```

```

        IPipe pipe,
        OOBControlMessage oobCtrlMsg);

}

```

## 6.2. onOOBControlMessage(IMessageComponent, IPipe, OOBControlMessage)

```

public void onOOBControlMessage(IMessageComponent source,
                                IPipe pipe,
                                OOBControlMessage oobCtrlMsg);

```

### Parameters

source	Message component source
pipe	Connection pipe TODO
oobCtrlMsg	Out-of-band control message

## 7. Interface IMessageInput

Input Endpoint for a consumer to connect.

### 7.1. Synopsis

```

public interface IMessageInput {
    // Public Methods

    public java.util.List<org.red5.server.messaging.IConsumer> getConsumers();

    public IMessage pullMessage()
        throws IOException;

    public IMessage pullMessage(long wait);

    public void sendOOBControlMessage(IConsumer consumer,
                                      OOBControlMessage oobCtrlMsg);

    public boolean subscribe(IConsumer consumer,
                           java.util.Map paramMap);

    public boolean unsubscribe(IConsumer consumer);

}

```

### 7.2. getConsumers()

```

public java.util.List<org.red5.server.messaging.IConsumer> getConsumers();

```

Getter for consumers list.

**Parameters**

<i>return</i>	Consumers.
---------------	------------

**7.3. pullMessage()**

```
public IMessage pullMessage()
    throws IOException;
```

Pull message from this input endpoint. Return w/o waiting.

**Parameters**

<i>return</i>	The pulled message or <code>null</code> if message is not available.
---------------	--

**7.4. pullMessage(long)**

```
public IMessage pullMessage(long wait);
```

Pull message from this input endpoint. Wait `wait` milliseconds if message is not available.

**Parameters**

<i>wait</i>	milliseconds to wait when message is not available.
-------------	---

<i>return</i>	The pulled message or <code>null</code> if message is not available.
---------------	--

**7.5. sendOOBControlMessage(IConsumer, OOBControlMessage)**

```
public void sendOOBControlMessage(IConsumer consumer,
    OOBControlMessage oobCtrlMsg);
```

Send OOB Control Message to all providers on the other side of pipe.

**Parameters**

<i>consumer</i>	The consumer that sends the message
-----------------	-------------------------------------

<i>oobCtrlMsg</i>	Out-of-band control message
-------------------	-----------------------------

**7.6. subscribe(IConsumer, Map)**

```
public boolean subscribe(IConsumer consumer,
    java.util.Map paramMap);
```

Connect to a consumer.

**Parameters**

<i>consumer</i>	Consumer
-----------------	----------

<i>paramMap</i>	Parameters map
-----------------	----------------

<i>return</i>	<code>true</code> when successfully subscribed, <code>false</code> otherwise.
---------------	---

## 7.7. unsubscribe(IConsumer)

```
public boolean unsubscribe(IConsumer consumer);
```

Disconnect from a consumer.

### Parameters

consumer	Consumer to disconnect
<i>return</i>	<code>true</code> when successfully unsubscribed, <code>false</code> otherwise.

## 8. Interface IMessagOutput

Output Endpoint for a provider to connect.

### 8.1. Synopsis

```
public interface IMessagOutput {
    // Public Methods

    public java.util.List<org.red5.server.messaging.IProvider> getProviders();

    public void pushMessage(IMessage message)
        throws IOException;

    public void sendOOBControlMessage(IProvider provider,
        OOBControlMessage oobCtrlMsg);

    public boolean subscribe(IProvider provider,
        java.util.Map paramMap);

    public boolean unsubscribe(IProvider provider);
}
```

### 8.2. getProviders()

```
public java.util.List<org.red5.server.messaging.IProvider> getProviders();
```

Getter for providers

### Parameters

<i>return</i>	Providers
---------------	-----------

### 8.3. pushMessage(IMessage)

```
public void pushMessage(IMessage message)
    throws IOException;
```

Push a message to this output endpoint. May block the pusher when output can't handle the message at the time.

**Parameters**

message	Message to be pushed.
---------	-----------------------

IOException

If message could not be written.

**8.4. sendOOBControlMessage(IProvider, OOBControlMessage)**

```
public void sendOOBControlMessage(IProvider provider,
                                  OOBControlMessage oobCtrlMsg);
```

Send OOB Control Message to all consumers on the other side of pipe.

**Parameters**

provider	The provider that sends the message
oobCtrlMsg	Out-of-band control message

**8.5. subscribe(IProvider, Map)**

```
public boolean subscribe(IProvider provider,
                        java.util.Map paramMap);
```

Connect to a provider. Note that params passed has nothing to deal with NetConnection.connect in client-side Flex/Flash RIA.

**Parameters**

provider	Provider
paramMap	Params passed with connection
<i>return</i>	true when successfully subscribed, false otherwise.

**8.6. unsubscribe(IProvider)**

```
public boolean unsubscribe(IProvider provider);
```

Disconnect from a provider.

**Parameters**

provider	Provider
<i>return</i>	true when successfully unsubscribed, false otherwise.

**9. Interface IPassive**

Signature to mark a provider/consumer never actively providers/consumers messages.

**9.1. Synopsis**

```
public interface IPassive {
```

```
// Public Static Fields

    public static final String KEY ;

}
```

## 10. Interface IPipe

A pipe is an object that connects message providers and message consumers. Its main function is to transport messages in kind of ways it provides. Pipes fire events as they go, these events are common way to work with pipes for higher level parts of server.

### 10.1. Synopsis

```
public interface IPipe extends, org.red5.server.messaging.IMessageInput, org.red5.server.messaging.IPipeConnectionListener {
    // Public Methods

    public void addPipeConnectionListener(IPipeConnectionListener listener);

    public void removePipeConnectionListener(IPipeConnectionListener listener);

}
```

### 10.2. addPipeConnectionListener(IPipeConnectionListener)

```
public void addPipeConnectionListener(IPipeConnectionListener listener);
```

Add connection event listener to pipe

#### Parameters

listener	Connection event listener
----------	---------------------------

### 10.3. removePipeConnectionListener(IPipeConnectionListener)

```
public void removePipeConnectionListener(IPipeConnectionListener listener);
```

Add connection event listener to pipe

#### Parameters

listener	Connection event listener
----------	---------------------------

## 11. Interface IPipeConnectionListener

A listener that wants to listen to events when provider/consumer connects to or disconnects from a specific pipe.

### 11.1. Synopsis

```
public interface IPipeConnectionListener {
```

```
// Public Methods

    public void onPipeConnectionEvent(PipeConnectionEvent event);

}
```

## 11.2. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(PipeConnectionEvent event);
```

Pipe connection event handler

### Parameters

event	Pipe connection event
-------	-----------------------

## 12. Interface IProvider

Signature for message provider.

### 12.1. Synopsis

```
public interface IProvider extends, org.?red5.?server.?messaging.?IMessageComponent { }
```

## 13. Interface IPullableProvider

A provider that supports passive pulling of messages.

### 13.1. Synopsis

```
public interface IPullableProvider extends, org.?red5.?server.?messaging.?IProvider {
// Public Static Fields

    public static final String KEY ;

// Public Methods

    public IMessage pullMessage(IPipe pipe)
        throws IOException;

    public IMessage pullMessage(IPipe pipe,
        long wait)
        throws IOException;

}
```

## 14. Interface IPushableConsumer

A consumer that supports event-driven message handling and message pushing through pipes.

## 14.1. Synopsis

```
public interface IPushableConsumer extends, org.?red5.?server.?messaging.?IConsumer {
    // Public Static Fields

    public static final String KEY ;

    // Public Methods

    public void pushMessage(IPipe pipe,
                           IMessage message)
        throws IOException;

}
```

## 14.2. pushMessage(IPipe, IMessage)

```
public void pushMessage(IPipe pipe,
                       IMessage message)
    throws IOException;
```

Pushes message through pipe

### Parameters

pipe	Pipe
message	Message

IOException  
if message could not be written

## 15. Class InMemoryPullPullPipe

A simple in-memory version of pull-pull pipe. It is triggered by an active consumer that pulls messages through it from a pullable provider.

## 15.1. Synopsis

```
public class InMemoryPullPullPipe extends, org.?red5.?server.?messaging.?AbstractPipe {
    // Public Constructors

    public InMemoryPullPullPipe();

    // Public Methods

    public IMessage pullMessage()
        throws IOException;

    public IMessage pullMessage(long wait);

    public void pushMessage(IMessage message);
```

```

public boolean subscribe(IConsumer consumer,
                        java.util.Map paramMap);

public boolean subscribe(IProvider provider,
                        java.util.Map paramMap);

}

```

**Methods inherited from org.red5.server.messaging.AbstractPipe:**

addPipeConnectionListener , fireConsumerConnectionEvent , firePipeConnectionEvent , fireProviderConnectionEvent , getConsumers , getListeners , getProviders , removePipeConnectionListener , sendOOBControlMessage , setListeners , subscribe , unsubscribe

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.messaging.AbstractPipe:** consumers , listeners , providers

## 15.2. pullMessage()

```

public IMessage pullMessage()
    throws IOException;

```

## 15.3. pullMessage(long)

```

public IMessage pullMessage(long wait);

```

## 15.4. pushMessage(IMessage)

```

public void pushMessage(IMessage message);

```

## 15.5. subscribe(IConsumer, Map)

```

public boolean subscribe(IConsumer consumer,
                        java.util.Map paramMap);

```

## 15.6. subscribe(IProvider, Map)

```

public boolean subscribe(IProvider provider,
                        java.util.Map paramMap);

```

# 16. Class InMemoryPushPushPipe

A simple in-memory version of push-push pipe. It is triggered by an active provider to push messages through it to an event-driven consumer.

## 16.1. Synopsis

```

public class InMemoryPushPushPipe extends, org.?red5.?server.?messaging.?AbstractPipe {
// Public Constructors

```

```

public InMemoryPushPushPipe();

// Public Methods

public IMessage pullMessage();

public IMessage pullMessage(long wait);

public void pushMessage(IMessage message)
    throws IOException;

public boolean subscribe(IConsumer consumer,
    java.util.Map paramMap);

public boolean subscribe(IProvider provider,
    java.util.Map paramMap);

}

```

**Methods inherited from org.red5.server.messaging.AbstractPipe:**

addPipeConnectionListener , fireConsumerConnectionEvent , firePipeConnectionEvent , fireProviderConnectionEvent , getConsumers , getListeners , getProviders , removePipeConnectionListener , sendOOBControlMessage , setListeners , subscribe , unsubscribe

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.messaging.AbstractPipe:** consumers , listeners , providers

## 16.2. pullMessage()

```
public IMessage pullMessage();
```

## 16.3. pullMessage(long)

```
public IMessage pullMessage(long wait);
```

## 16.4. pushMessage(IMessage)

```
public void pushMessage(IMessage message)
    throws IOException;
```

Pushes a message out to all the PushableConsumers.

## 16.5. subscribe(IConsumer, Map)

```
public boolean subscribe(IConsumer consumer,
    java.util.Map paramMap);
```

## 16.6. subscribe(IProvider, Map)

```
public boolean subscribe(IProvider provider,
```

```
java.util.Map paramMap);
```

## 17. Class OOBControlMessage

Out-of-band control message used by inter-components communication which are connected with pipes. Out-of-band data is a separate data stream used for specific purposes (in TCP it's referenced as "urgent data"), like lifecycle control. 'Target' is used to represent the receiver who may be interested for receiving. It's a string of any form. XXX shall we design a standard form for Target, like "class.instance"?

### 17.1. Synopsis

```
public class OOBControlMessage implements, java.io.Serializable {
// Public Constructors

    public OOBControlMessage();

// Public Methods

    public Object getResult();

    public String getServiceName();

    public java.util.Map getServiceParamMap();

    public String getTarget();

    public void setResult(Object result);

    public void setServiceName(String serviceName);

    public void setServiceParamMap(java.util.Map serviceParamMap);

    public void setTarget(String target);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 17.2. getResult()

```
public Object getResult();
```

Getter for property 'result'.

#### Parameters

<i>return</i>	Value for property 'result'.
---------------	------------------------------

### 17.3. getServiceName()

```
public String getServiceName();
```

Getter for property 'serviceName'.

#### Parameters

<i>return</i>	Value for property 'serviceName'.
---------------	-----------------------------------

## 17.4. getServiceParamMap()

```
public java.util.Map getServiceParamMap();
```

Getter for property 'serviceParamMap'.

#### Parameters

<i>return</i>	Value for property 'serviceParamMap'.
---------------	---------------------------------------

## 17.5. getTarget()

```
public String getTarget();
```

Getter for property 'target'.

#### Parameters

<i>return</i>	Value for property 'target'.
---------------	------------------------------

## 17.6. setResult(Object)

```
public void setResult(Object result);
```

Setter for property 'result'.

#### Parameters

<i>result</i>	Value to set for property 'result'.
---------------	-------------------------------------

## 17.7. setServiceName(String)

```
public void setServiceName(String serviceName);
```

Setter for property 'serviceName'.

#### Parameters

<i>serviceName</i>	Value to set for property 'serviceName'.
--------------------	--

## 17.8. setServiceParamMap(Map)

```
public void setServiceParamMap(java.util.Map serviceParamMap);
```

Setter for property 'serviceParamMap'.

#### Parameters

serviceParamMap	Value to set for property 'serviceParamMap'.
-----------------	--

## 17.9. setTarget(String)

```
public void setTarget(String target);
```

Setter for property 'target'.

### Parameters

target	Value to set for property 'target'.
--------	-------------------------------------

# 18. Class PipeConnectionEvent

Event object corresponds to the connect/disconnect events among providers/consumers and pipes.

## 18.1. Synopsis

```
public class PipeConnectionEvent extends, java.util.EventObject {  
    // Public Static Fields
```

```
        public static final int CONSUMER_CONNECT_PULL = 3;
```

```
        public static final int CONSUMER_CONNECT_PUSH = 4;
```

```
        public static final int CONSUMER_DISCONNECT = 5;
```

```
        public static final int PROVIDER_CONNECT_PULL = 0;
```

```
        public static final int PROVIDER_CONNECT_PUSH = 1;
```

```
        public static final int PROVIDER_DISCONNECT = 2;
```

```
    // Public Constructors
```

```
        public PipeConnectionEvent(Object source);
```

```
    // Public Methods
```

```
        public void addTask(Runnable task);
```

```
        public IConsumer getConsumer();
```

```
        public java.util.Map getParamMap();
```

```
        public IProvider getProvider();
```

```
        public int getType();
```

```
        public void setConsumer(IConsumer consumer);
```

```
        public void setParamMap(java.util.Map paramMap);
```

```
        public void setProvider(IProvider provider);
```

```
public void setType(int type);
}
```

**Methods inherited from java.util.EventObject:** getSource , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

**Fields inherited from java.util.EventObject:** source

## 18.2. PipeConnectionEvent(Object)

```
public PipeConnectionEvent(Object source);
```

Construct an object with the specific pipe as the source

### Parameters

source	A pipe that triggers this event.
--------	----------------------------------

## 18.3. CONSUMER\_CONNECT\_PULL

```
public static final int CONSUMER_CONNECT_PULL = 3;
```

A consumer connects as pull mode.

## 18.4. CONSUMER\_CONNECT\_PUSH

```
public static final int CONSUMER_CONNECT_PUSH = 4;
```

A consumer connects as push mode.

## 18.5. CONSUMER\_DISCONNECT

```
public static final int CONSUMER_DISCONNECT = 5;
```

A consumer disconnects.

## 18.6. PROVIDER\_CONNECT\_PULL

```
public static final int PROVIDER_CONNECT_PULL = 0;
```

A provider connects as pull mode.

## 18.7. PROVIDER\_CONNECT\_PUSH

```
public static final int PROVIDER_CONNECT_PUSH = 1;
```

A provider connects as push mode.

## 18.8. PROVIDER\_DISCONNECT

```
public static final int PROVIDER_DISCONNECT = 2;
```

A provider disconnects.

## 18.9. addTask(Runnable)

```
public void addTask(Runnable task);
```

Add task to list

Parameters

task	Task to add
------	-------------

## 18.10. getConsumer()

```
public IConsumer getConsumer();
```

Return pipe connection consumer

Parameters

return	Consumer
--------	----------

## 18.11. getParamMap()

```
public java.util.Map getParamMap();
```

Return event parameters as Map

Parameters

return	Event parameters as Map
--------	-------------------------

## 18.12. getProvider()

```
public IProvider getProvider();
```

Return pipe connection provider

Parameters

return	Provider
--------	----------

## 18.13. getType()

```
public int getType();
```

Return event type

Parameters

return	Event type
--------	------------

## 18.14. setConsumer(IConsumer)

```
public void setConsumer(IConsumer consumer);
```

Setter for pipe connection consumer

### Parameters

consumer	Consumer
----------	----------

## 18.15. setParamMap(Map)

```
public void setParamMap(java.util.Map paramMap);
```

Setter for event parameters map

### Parameters

paramMap	Event parameters as Map
----------	-------------------------

## 18.16. setProvider(IProvider)

```
public void setProvider(IProvider provider);
```

Setter for pipe connection provider

### Parameters

provider	Provider
----------	----------

## 18.17. setType(int)

```
public void setType(int type);
```

Setter for event type

### Parameters

type	Event type
------	------------

# 19. Class PipeUtils

Helper class for pipe structure.

## 19.1. Synopsis

```
public class PipeUtils {
// Public Constructors

    public PipeUtils();

// Public Static Methods

    public static void connect(IProvider provider,
```

```

        IPipe pipe,
        IConsumer consumer);

    public static void disconnect(IProvider provider,
                                 IPipe pipe,
                                 IConsumer consumer);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 19.2. connect(IProvider, IPipe, IConsumer)

```

public static void connect(IProvider provider,
                           IPipe pipe,
                           IConsumer consumer);

```

Connect a provider/consumer with a pipe.

Parameters	
provider	Provider
pipe	Pipe that used to establish connection
consumer	Consumer

## 19.3. disconnect(IProvider, IPipe, IConsumer)

```

public static void disconnect(IProvider provider,
                             IPipe pipe,
                             IConsumer consumer);

```

Disconnect a provider/consumer from a pipe.

Parameters	
provider	Provider
pipe	Pipe to disconnect from
consumer	Consumer

# 1. Class MidiPlayer

Plays a midi file provided on command line

## 1.1. Synopsis

```
public class MidiPlayer {  
    // Public Constructors  
  
    public MidiPlayer(java.io.File midiFile);  
  
    // Public Static Methods  
  
    public static void main(String[] args);  
  
}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

# 2. Class SharedMidiObject

```
public class SharedMidiObject {  
    // Protected Fields  
  
    protected javax.sound.midi.MidiDevice dev ;  
  
    protected String deviceName ;  
  
    protected org.red5.server.api.so.ISharedObject so ;  
  
    // Public Constructors  
  
    public SharedMidiObject(String deviceName,  
                           org.red5.server.api.so.ISharedObject so);  
  
    // Public Static Methods  
  
    public static javax.sound.midi.MidiDevice getMidiDevice(String name);  
  
    // Public Methods  
  
    public void close();  
  
    public boolean connect();  
  
}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

# 3. Class SharedMidiObject.MidiReceiver

```

public class SharedMidiObject.MidiReceiver implements javax.sound.midi.Receiver {
// Public Constructors

    public SharedMidiObject.MidiReceiver();

// Public Methods

    public void close();

    public void send(javax.sound.midi.MidiMessage midi,
                    long time);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 3.1. close()

```
public void close();
```

**Specified by:** Method `close` in interface `Receiver`

### 3.2. send(MidiMessage, long)

```
public void send(javax.sound.midi.MidiMessage midi,
                 long time);
```

**Specified by:** Method `send` in interface `Receiver`

## 4. Class Test

```

public class Test {
// Protected Fields

    protected static org.slf4j.Logger log;

// Public Constructors

    public Test()
        throws Exception;

// Public Static Methods

    public static javax.sound.midi.MidiDevice getMidiDevice(String name);

    public static void main(String[] args)
        throws Exception;

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 4.1. Test()

```
public Test()
throws Exception;
```

Constructs a new Test.

## 5. Class Test.MyReceiver

```
public class Test.MyReceiver implements javax.sound.midi.Receiver {
// Public Constructors

    public Test.MyReceiver();

// Public Methods

    public void close();

    public void send(javax.sound.midi.MidiMessage midi,
                    long time);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 5.1. close()

```
public void close();
```

**Specified by:** Method `close` in interface `Receiver`

### 5.2. send(MidiMessage, long)

```
public void send(javax.sound.midi.MidiMessage midi,
                 long time);
```

**Specified by:** Method `send` in interface `Receiver`

# 1. Interface IConnectionEventQueue

Queue of connection events

## 1.1. Synopsis

```
public interface IConnectionEventQueue {  
    // Public Methods  
  
    public boolean hasEventsWaiting(org.red5.server.api.IConnection conn);  
  
    public java.util.Iterator<org.red5.server.api.event.IEvent> pickupEvents(org.red5.server.api.IConn  
}  
}
```

## 1.2. hasEventsWaiting(IConnection)

```
public boolean hasEventsWaiting(org.red5.server.api.IConnection conn);
```

Whether queue has waiting connection events

### Parameters

conn	Connection
------	------------

return	true if queue has waiting events for connection, false otherwise
--------	--

## 1.3. pickupEvents(IConnection)

```
public java.util.Iterator<org.red5.server.api.event.IEvent> pickupEvents(org.red5.server.api.IConnec
```

Return iterator over waiting events

### Parameters

conn	Connection
------	------------

return	Iterator over events
--------	----------------------

# 1. Class BaseMRTMPConnection

```
public class BaseMRTMPConnection implements org.red5.server.net.mrtmp.IMRTMPConnection {
    // Public Constructors

    public BaseMRTMPConnection();

    // Public Methods

    public void close();

    public void connect(int clientId);

    public void disconnect(int clientId);

    public org.apache.mina.common.IoSession getIoSession();

    public void setIoSession(org.apache.mina.common.IoSession ioSession);

    public void write(int clientId,
                      org.red5.server.net.rtmp.message.Packet packet);

}
```

**Direct known subclasses:** org.red5.server.net.mrtmp.MRTMPEdgeConnection , org.red5.server.net.mrtmp.MRTMPOriginConnection

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 1.1. connect(int)

```
public void connect(int clientId);
```

**Specified by:** Method connect in interface IMRTMPConnection

Send connect message to other side

### Parameters

clientId
----------

**Description copied from interface: connect**

## 1.2. disconnect(int)

```
public void disconnect(int clientId);
```

**Specified by:** Method disconnect in interface IMRTMPConnection

Send disconnect message to other side

### Parameters

clientId	
----------	--

**Description copied from interface: disconnect****1.3. write(int, Packet)**

```
public void write(int clientId,
                  org.red5.server.net.rtmp.message.Packet packet);
```

**Specified by:** Method write in interface IMRTMPConnection

Send RTMP packet to other side

Parameters	
------------	--

clientId	
----------	--

packet	
--------	--

**Description copied from interface: write****2. Class EdgeMRTMPHandler**

```
public class EdgeMRTMPHandler extends org.apache.mina.common.IoHandlerAdapter
    implements org.red5.server.net.rtmp.message.Constants {
    // Public Constructors

    public EdgeMRTMPHandler();

    // Public Methods

    public void messageReceived(org.apache.mina.common.IoSession session,
                               Object message)
        throws Exception;

    public void messageSent(org.apache.mina.common.IoSession session,
                           Object message)
        throws Exception;

    public void sessionClosed(org.apache.mina.common.IoSession session)
        throws Exception;

    public void sessionCreated(org.apache.mina.common.IoSession session)
        throws Exception;

    public void setCodecFactory(org.apache.mina.filter.codec.ProtocolCodecFactory codecFactory);

    public void setMrtmpManager(IMRTMPEdgeManager mrtmpManager);

    public void setRtmpConnManager(org.red5.server.net.rtmp.IRTMPConnManager rtmpConnManager);

}
```

**Methods inherited from org.apache.mina.common.IoHandlerAdapter:**

exceptionCaught , messageReceived , messageSent , sessionClosed , sessionCreated ,  
 sessionIdle , sessionOpened

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 3. Interface IMRTMPConnection

```
public interface IMRTMPConnection {
// Public Methods

    public void close();

    public void connect(int clientId);

    public void disconnect(int clientId);

    public void write(int clientId,
                     org.red5.server.net.rtmp.message.Packet packet);

}
```

### 3.1. connect(int)

```
public void connect(int clientId);
```

Send connect message to other side

#### Parameters

clientId
----------

### 3.2. disconnect(int)

```
public void disconnect(int clientId);
```

Send disconnect message to other side

#### Parameters

clientId
----------

### 3.3. write(int, Packet)

```
public void write(int clientId,
                  org.red5.server.net.rtmp.message.Packet packet);
```

Send RTMP packet to other side

#### Parameters

clientId
----------

packet
--------

## 4. Interface IMRTMPEdgeManager

A tag interface.

### 4.1. Synopsis

```
public interface IMRTMPEdgeManager extends, org.?red5.?server.?net.?mrtmp.?IMRTMPManager {  
}
```

## 5. Interface IMRTMPManager

```
public interface IMRTMPManager {  
    // Public Methods  
  
    public IMRTMPConnection lookupMRTMPConnection(org.red5.server.net.rtmp.RTMPConnection conn);  
  
    public boolean registerConnection(IMRTMPConnection conn);  
  
    public boolean unregisterConnection(IMRTMPConnection conn);  
}
```

### 5.1. lookupMRTMPConnection(RTMPConnection)

```
public IMRTMPConnection lookupMRTMPConnection(org.red5.server.net.rtmp.RTMPConnection conn);
```

Map a client to an Edge/Origin MRTMP connection. On Edge, the server will find an Origin connection per routing logic. On Origin, the server will send to the original in-coming connection if the client connection type is persistent. Or the latest in-coming connection will be used.

#### Parameters

conn	
<i>return</i>	

### 5.2. registerConnection(IMRTMPConnection)

```
public boolean registerConnection(IMRTMPConnection conn);
```

Register a MRTMP connection so that it can be later been looked up.

#### Parameters

conn	
<i>return</i>	whether the registration is successful

## 5.3. unregisterConnection(IMRTMPConnection)

```
public boolean unregisterConnection(IMRTMPConnection conn);
```

Unregister a MRTMP connection.

### Parameters

conn	
------	--

<i>return</i>	whether the registration is successful
---------------	--

## 6. Interface IMRTMPOriginManager

```
public interface IMRTMPOriginManager extends, org.?red5.?server.?net.?mrtmp.?IMRTMPManager {
    // Public Methods

    public void associate(org.red5.server.net.rtmp.RTMPConnection rtmpConn,
                         IMRTMPConnection mrtmpConn);

    public void dissociate(org.red5.server.net.rtmp.RTMPConnection rtmpConn);

}
```

### 6.1. associate(RTMPConnection, IMRTMPConnection)

```
public void associate(org.red5.server.net.rtmp.RTMPConnection rtmpConn,
                     IMRTMPConnection mrtmpConn);
```

Associate the client to a MRTMP connection so that the packet will be sent via this MRTMP connection. The association has different impacts on persistent and polling connections. For persistent connection, the mapping is static while for polling connection, the mapping is dynamic and might not be honored.

### Parameters

clientId	
----------	--

conn	
------	--

### 6.2. dissociate(RTMPConnection)

```
public void dissociate(org.red5.server.net.rtmp.RTMPConnection rtmpConn);
```

Deassociate the client from the MRTMP connection previously associated to.

### Parameters

rtmpConn	
----------	--

## 7. Class MRTMPClient

```

public class MRTMPClient implements, java.lang.Runnable {
// Public Constructors

    public MRTMPClient();

// Public Methods

    public org.apache.mina.common.IoHandler getIoHandler();

    public int getPort();

    public String getServer();

    public void run();

    public void setIoHandler(org.apache.mina.common.IoHandler ioHandler);

    public void setPort(int port);

    public void setServer(String address);

    public void start();

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 8. Class MRTMPEdgeConnection

```

public class MRTMPEdgeConnection extends, org.red5.server.net.mrmp.BaseMRTMPConnection {
// Public Constructors

    public MRTMPEdgeConnection();

}

```

**Methods inherited from org.red5.server.net.mrmp.BaseMRTMPConnection:** close, connect, disconnect, getIoSession, setIoSession, write

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 9. Class MRTMPMinaTransport

```

public class MRTMPMinaTransport {
// Public Constructors

    public MRTMPMinaTransport();

// Public Methods

```

```

    public void setAddress(String address);

    public void setEventThreadsCore(int eventThreadsCore);

    public void setEventThreadsKeepalive(int eventThreadsKeepalive);

    public void setEventThreadsMax(int eventThreadsMax);

    public void setEventThreadsQueue(int eventThreadsQueue);

    public void setIoHandler(org.apache.mina.common.IoHandlerAdapter rtmpIOHandler);

    public void setIoThreads(int ioThreads);

    public void setIsLoggingTraffic(boolean isLoggingTraffic);

    public void setPort(int port);

    public void setReceiveBufferSize(int receiveBufferSize);

    public void setSendBufferSize(int sendBufferSize);

    public void setTcpNoDelay(boolean tcpNoDelay);

    public void setUseHeapBuffers(boolean useHeapBuffers);

    public void start()
        throws Exception;

    public void stop();

    public String toString();
}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

## 10. Class MRTMPOriginConnection

```

public class MRTMPOriginConnection extends, org.?red5.?server.?net.?mrtmp.?BaseMRTMPConnection {
// Public Constructors

    public MRTMPOriginConnection();

// Public Methods

    public void connect(int clientId);

}

```

**Methods inherited from org.red5.server.net.mrtmp.BaseMRTMPConnection:** close ,  
connect , disconnect , getIoSession , setIoSession , write

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 11. Class MRTMPPacket

```
public class MRTMPPacket {
// Public Static Fields

    public static final short CLOSE = 1;

    public static final int COMMON_HEADER_LENGTH = 20;

    public static final short CONNECT = 0;

    public static final short JAVA_ENCODING = 0;

    public static final short RTMP = 2;

    public static final int RTMP_HEADER_LENGTH = 24;

// Public Constructors

    public MRTMPPacket();

// Public Methods

    public org.red5.server.net.mrtmp.MRTMPPacket.Body getBody();

    public org.red5.server.net.mrtmp.MRTMPPacket.Header getHeader();

    public void setBody(org.red5.server.net.mrtmp.MRTMPPacket.Body body);

    public void setHeader(org.red5.server.net.mrtmp.MRTMPPacket.Header header);

    public String toString();

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 12. Class MRTMPPacket.Body

```
public static class MRTMPPacket.Body {
// Public Constructors

    public MRTMPPacket.Body();

// Public Methods

    public org.apache.mina.common.ByteBuffer getRawBuf();

    public void setRawBuf(org.apache.mina.common.ByteBuffer rawBuf);
```

}

**Direct known subclasses:** org.?red5.?server.?net.?mrtmp.?MRTMPPacket.?RTMPBody**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 13. Class MRTMPPacket.Header

```
public static class MRTMPPacket.Header {
// Public Constructors

    public MRTMPPacket.Header();

// Public Methods

    public short getBodyEncoding();

    public int getBodyLength();

    public int getClientId();

    public int getHeaderLength();

    public short getType();

    public boolean isDynamic();

    public void setBodyEncoding(short bodyEncoding);

    public void setBodyLength(int bodyLength);

    public void setClientId(int clientId);

    public void setDynamic(boolean dynamic);

    public void setHeaderLength(int headerLength);

    public void setType(short type);

}
```

**Direct known subclasses:** org.?red5.?server.?net.?mrtmp.?MRTMPPacket.?RTMPHeader**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 14. Class MRTMPPacket.RTMPBody

```
public static class MRTMPPacket.RTMPBody extends, org.?red5.?server.?net.?mrtmp.?MRTMPPacket.?Body {
// Public Constructors
```

```

public MRTMPPacket.RTMPBody();

// Public Methods

public org.red5.server.net.rtmp.message.Packet getRtmpPacket();

public void setRtmpPacket(org.red5.server.net.rtmp.message.Packet rtmpPacket);

}

```

**Methods inherited from org.red5.server.net.mrtmp.MRTMPPacket.Body:** `getRawBuf` , `setRawBuf`

**Methods inherited from java.lang.Object:** `clone` , `equals` , `finalize` , `getClass` , `hashCode` , `notify` , `notifyAll` , `toString` , `wait`

## 15. Class MRTMPPacket.RTMPHeader

```

public static class MRTMPPacket.RTMPHeader extends org.?red5.?server.?net.?mrtmp.?MRTMPPacket.?Header
// Public Constructors

public MRTMPPacket.RTMPHeader();

// Public Methods

public int getRtmpType();

public void setRtmpType(int rtmpType);

}

```

**Methods inherited from org.red5.server.net.mrtmp.MRTMPPacket.Header:**

`getBodyEncoding` , `getBodyLength` , `getClientId` , `getHeaderLength` , `getType` , `isDynamic` , `setBodyEncoding` , `setBodyLength` , `setClientId` , `setDynamic` , `setHeaderLength` , `setType`

**Methods inherited from java.lang.Object:** `clone` , `equals` , `finalize` , `getClass` , `hashCode` , `notify` , `notifyAll` , `toString` , `wait`

## 16. Class OriginMRTMPHandler

```

public class OriginMRTMPHandler extends org.?apache.?mina.?common.?IoHandlerAdapter {
// Public Constructors

public OriginMRTMPHandler();

// Public Methods

public void closeConnection(org.red5.server.net.rtmp.RTMPOriginConnection conn);

public void messageReceived(org.apache.mina.common.IoSession session,
                           Object message)
                           throws Exception;

```

```

public void messageSent(org.apache.mina.common.IoSession session,
                      Object message)
                      throws Exception;

public void sessionClosed(org.apache.mina.common.IoSession session)
                         throws Exception;

public void sessionCreated(org.apache.mina.common.IoSession session)
                           throws Exception;

public void setCodecFactory(org.apache.mina.filter.codec.ProtocolCodecFactory codecFactory);

public void setHandler(org.red5.server.net.rtmp.IRTMPHandler handler);

public void setMrtmpManager(IMRTMPOriginManager mrtmpManager);

// Protected Methods

protected int getSessionId(org.apache.mina.common.IoSession session);

}

```

**Methods inherited from org.apache.mina.common.IoHandlerAdapter:**

exceptionCaught , messageReceived , messageSent , sessionClosed , sessionCreated ,  
sessionIdle , sessionOpened

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 17. Class SimpleMRTMPEdgeManager

A simple Edge connection manager that only manages one Edge/Origin connection.

### 17.1. Synopsis

```

public class SimpleMRTMPEdgeManager implements org.?red5.?server.?net.?mrtmp.?IMRTMPEdgeManager {
// Public Constructors

    public SimpleMRTMPEdgeManager();

// Public Methods

    public IMRTMPConnection lookupMRTMPConnection(org.red5.server.net.rtmp.RTMPConnection conn);

    public boolean registerConnection(IMRTMPConnection conn);

    public void setRtmpConnManager(org.red5.server.net.rtmp.IRTMPConnManager rtmpConnManager);

    public boolean unregisterConnection(IMRTMPConnection conn);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 18. Class SimpleMRTMPOriginManager

```

public class SimpleMRTMPOriginManager implements org.red5.server.net.mrtmp.IMRTMPOriginManager {
    // Public Constructors

    public SimpleMRTMPOriginManager();

    // Public Methods

    public void associate(org.red5.server.net.rtmp.RTMPConnection rtmpConn,
                         IMRTMPConnection mrtmpConn);

    public void dissociate(org.red5.server.net.rtmp.RTMPConnection rtmpConn);

    public IMRTMPConnection lookupMRTMPConnection(org.red5.server.net.rtmp.RTMPConnection rtmpConn);

    public boolean registerConnection(IMRTMPConnection conn);

    public void setOriginMRTMPHandler(OriginMRTMPHandler originMRTMPHandler);

    public boolean unregisterConnection(IMRTMPConnection conn);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 18.1. associate(RTMPConnection, IMRTMPConnection)

```

public void associate(org.red5.server.net.rtmp.RTMPConnection rtmpConn,
                     IMRTMPConnection mrtmpConn);

```

**Specified by:** Method associate in interface IMRTMPOriginManager

Associate the client to a MRTMP connection so that the packet will be sent via this MRTMP connection. The association has different impacts on persistent and polling connections. For persistent connection, the mapping is static while for polling connection, the mapping is dynamic and might not be honored.

#### Parameters

clientId	
conn	

**Description copied from interface: associate**

### 18.2. dissociate(RTMPConnection)

```

public void dissociate(org.red5.server.net.rtmp.RTMPConnection rtmpConn);

```

**Specified by:** Method dissociate in interface IMRTMPOriginManager

Deassociate the client from the MRTMP connection previously associated to.

**Parameters**

rtmpConn	
----------	--

**Description copied from interface: dissociate**

## 1. Class MRTMPCodecFactory

```
public class MRTMPCodecFactory implements org.apache.mina.filter.codec.ProtocolCodecFactory {
    // Public Constructors

    public MRTMPCodecFactory();

    // Public Methods

    public org.apache.mina.filter.codec.ProtocolDecoder getDecoder()
        throws Exception;

    public org.apache.mina.filter.codec.ProtocolEncoder getEncoder()
        throws Exception;

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2. Class MRTMPProtocolDecoder

```
public class MRTMPProtocolDecoder implements org.apache.mina.filter.codec.ProtocolDecoder {
    // Public Constructors

    public MRTMPProtocolDecoder();

    // Public Methods

    public void decode(org.apache.mina.common.IoSession session,
                      org.apache.mina.common.ByteBuffer in,
                      org.apache.mina.filter.codec.ProtocolDecoderOutput out)
        throws Exception;

    public org.red5.server.net.mrtmp.MRTMPPacket.Body decodeBody(org.apache.mina.common.ByteBuffer buf
                                                                org.red5.server.net.mrtmp.MRTMPPacket);

    public org.red5.server.net.mrtmp.MRTMPPacket.Header decodeHeader(org.apache.mina.common.ByteBuffer buf);

    public void dispose(org.apache.mina.common.IoSession session)
        throws Exception;

    public void finishDecode(org.apache.mina.common.IoSession session,
                           org.apache.mina.filter.codec.ProtocolDecoderOutput out)
        throws Exception;

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 3. Class MRTMPProtocolEncoder

```
public class MRTMPProtocolEncoder implements org.apache.mina.filter.codec.ProtocolEncoder {  
    // Public Constructors  
  
    public MRTMPProtocolEncoder();  
  
    // Public Methods  
  
    public void dispose(org.apache.mina.common.IoSession session)  
        throws Exception;  
  
    public void encode(org.apache.mina.common.IoSession session,  
                      Object message,  
                      org.apache.mina.filter.codec.ProtocolEncoderOutput out)  
        throws Exception;  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

# 1. Class SocketPolicyHandler

Provides the socket policy file.

## 1.1. Synopsis

```
public class SocketPolicyHandler extends org.apache.mina.common.IoHandlerAdapter {  
    // Protected Fields  
  
    protected static org.slf4j.Logger log ;  
  
    // Public Constructors  
  
    public SocketPolicyHandler();  
  
    // Public Methods  
  
    public void exceptionCaught(org.apache.mina.common.IoSession session,  
                                Throwable ex)  
        throws Exception;  
  
    public String getHost();  
  
    public int getPort();  
  
    public void messageReceived(org.apache.mina.common.IoSession session,  
                               Object message)  
        throws Exception;  
  
    public void setHost(String host);  
  
    public void setPort(int port);  
  
    public void start();  
  
    public void stop();  
}
```

### Methods inherited from org.apache.mina.common.IoHandlerAdapter:

exceptionCaught , messageReceived , messageSent , sessionClosed , sessionCreated ,  
sessionIdle , sessionOpened

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

### See Also

[http://www.adobe.com/devnet/flashplayer/articles/socket\\_policy\\_files.html](http://www.adobe.com/devnet/flashplayer/articles/socket_policy_files.html)

# 1. Class BaseProtocolEncoder

Base class for all protocol encoders.

## 1.1. Synopsis

```
public abstract class BaseProtocolEncoder implements org.red5.server.net.protocol.SimpleProtocol
// Public Constructors

    public BaseProtocolEncoder();

// Protected Methods

    protected org.red5.server.net.rtmp.status.StatusObject generateErrorResult(String code,
        Throwable error);

}
```

**Direct known subclasses:** org.red5.server.net.remoting.codec.RemotingProtocolEncoder , org.red5.server.net.rtmp.codec.RTMPPProtocolEncoder

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 1.2. generateErrorResult(String, Throwable)

```
protected org.red5.server.net.rtmp.status.StatusObject generateErrorResult(String code,
    Throwable error);
```

Generate error object to return for given exception.

### Parameters

call	
<i>return</i>	

# 2. Exception HandshakeFailedException

```
public class HandshakeFailedException extends org.red5.server.net.protocol.ProtocolException {
// Public Constructors

    public HandshakeFailedException(String message);

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 2.1. HandshakeFailedException(String)

```
public HandshakeFailedException(String message);
```

Create handshake failed exception with given message

### Parameters

message
---------

## 3. Exception ProtocolException

```
public class ProtocolException extends java.lang.RuntimeException {
// Public Constructors

    public ProtocolException(String message);

    public ProtocolException(String message,
                           Throwable cause);

}
```

**Direct known subclasses:** org.red5.server.net.protocol.HandshakeFailedException

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 3.1. ProtocolException(String)

```
public ProtocolException(String message);
```

Create protocol exception with given message.

### Parameters

message
---------

## 3.2. ProtocolException(String, Throwable)

```
public ProtocolException(String message,
                        Throwable cause);
```

Create protocol exception with given message and cause.

### Parameters

message	
cause	

## 4. Class ProtocolState

Represents current state of protocol.

### 4.1. Synopsis

```
public class ProtocolState {
    // Public Static Fields

    public static byte DECODER_BUFFER ;
    public static byte DECODER_CONTINUE ;
    public static byte DECODER_OK ;
    public static final String SESSION_KEY = "protocol_state";

    // Public Constructors

    public ProtocolState();

    // Public Methods

    public void bufferDecoding(int amount);
    public boolean canContinueDecoding();
    public boolean canStartDecoding(int remaining);
    public void continueDecoding();
    public int getDecoderBufferAmount();
    public boolean hasDecodedObject();
    public void startDecoding();
}
```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?codec.?RTMP

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 4.2. DECODER\_BUFFER

```
public static byte DECODER_BUFFER ;
```

Decoder is buffering state constant.

## 4.3. DECODER\_CONTINUE

```
public static byte DECODER_CONTINUE ;
```

Decoding continues state constant.

## 4.4. DECODER\_OK

```
public static byte DECODER_OK ;
```

Decoding finished successfully state constant.

## 4.5. SESSION\_KEY

```
public static final String SESSION_KEY = "protocol_state";
```

Session key constant.

## 4.6. bufferDecoding(int)

```
public void bufferDecoding(int amount);
```

Specifies buffer decoding amount

### Parameters

amount	Buffer decoding amount
--------	------------------------

## 4.7. canContinueDecoding()

```
public boolean canContinueDecoding();
```

Checks whether decoding process can be continued.

### Parameters

return	true if decoding can be continued, false otherwise
--------	--

## 4.8. canStartDecoding(int)

```
public boolean canStartDecoding(int remaining);
```

Checks whether remaining buffer size is greater or equal than buffer amount and so if it makes sense to start decoding.

### Parameters

remaining	Remaining buffer size
-----------	-----------------------

return	true if there is data to decode, false otherwise
--------	--

## 4.9. continueDecoding()

```
public void continueDecoding();
```

Set decoding state as "needed to be continued".

## 4.10. getDecoderBufferAmount()

```
public int getDecoderBufferAmount();
```

Returns current buffer amount.

### Parameters

<i>return</i>	Buffer amount
---------------	---------------

## 4.11. hasDecodedObject()

```
public boolean hasDecodedObject();
```

Checks whether decoding is complete.

### Parameters

<i>return</i>	true if decoding has finished, false otherwise
---------------	--

## 4.12. startDecoding()

```
public void startDecoding();
```

Starts decoding. Sets state to "ready" and clears buffer amount.

# 5. Interface SimpleProtocolCodecFactory

Simple protocol codec factory

## 5.1. Synopsis

```
public interface SimpleProtocolCodecFactory {
    // Public Methods

    public SimpleProtocolDecoder getSimpleDecoder();

    public SimpleProtocolEncoder getSimpleEncoder();

}
```

## 5.2. getSimpleDecoder()

```
public SimpleProtocolDecoder getSimpleDecoder();
```

Getter for simple decoder.

### Parameters

<i>return</i>	Value for property 'simpleDecoder'.
---------------	-------------------------------------

### 5.3. getSimpleEncoder()

```
public SimpleProtocolEncoder getSimpleEncoder();
```

Getter for simple encoder.

#### Parameters

<i>return</i>	Value for property 'simpleEncoder'.
---------------	-------------------------------------

## 6. Interface SimpleProtocolDecoder

Simple protocol decoder

### 6.1. Synopsis

```
public interface SimpleProtocolDecoder {
    // Public Methods

    public Object decode(ProtocolState state,
                        org.apache.mina.common.ByteBuffer in)
                        throws Exception;

    public java.util.List decodeBuffer(ProtocolState state,
                                      org.apache.mina.common.ByteBuffer buffer);

}
```

### 6.2. decode(ProtocolState, ByteBuffer)

```
public Object decode(ProtocolState state,
                    org.apache.mina.common.ByteBuffer in)
                    throws Exception;
```

#### Parameters

state	Stores state for the protocol, ProtocolState is just a marker interface
in	ByteBuffer of data to be decoded
<i>return</i>	one of three possible values. null : the object could not be decoded, or some data was skipped, just continue. ProtocolState : the decoder was unable to decode the whole object, refer to the protocol state Object : something was decoded, continue

ProtocolCodecException  
org.apache.mina.filter.codec.ProtocolCodecException

### 6.3. decodeBuffer(ProtocolState, ByteBuffer)

```
public java.util.List decodeBuffer(ProtocolState state,
```

```
org.apache.mina.common.ByteBuffer buffer);
```

Decode all available objects in buffer.

#### Parameters

state	Stores state for the protocol
buffer	ByteBuffer of data to be decoded
<i>return</i>	a list of decoded objects, may be empty if nothing could be decoded

## 7. Interface SimpleProtocolEncoder

Every protocol encoder should implement this

### 7.1. Synopsis

```
public interface SimpleProtocolEncoder {
    // Public Methods

    public org.apache.mina.common.ByteBuffer encode(ProtocolState state,
                                                    Object out)
        throws Exception;

}
```

### 7.2. encode(ProtocolState, Object)

```
public org.apache.mina.common.ByteBuffer encode(ProtocolState state,
                                                Object out)
    throws Exception;
```

Encodes object with given protocol state to byte buffer

#### Parameters

state	Protocol state
out	Object to encode
<i>return</i>	Byte buffer with encoded data

Exception

Any decoding exception

# 1. Class DebugProxyHandler

```
public class DebugProxyHandler extends org.apache.mina.common.IoHandlerAdapter
    implements org.springframework.context.ResourceLoaderAware {
// Protected Fields

    protected static org.slf4j.Logger log;

// Public Constructors

    public DebugProxyHandler();

// Public Methods

    public void exceptionCaught(org.apache.mina.common.IoSession session,
                               Throwable cause)
        throws Exception;

    public void messageReceived(org.apache.mina.common.IoSession session,
                               Object in);

    public void sessionCreated(org.apache.mina.common.IoSession session)
        throws Exception;

    public void sessionOpened(org.apache.mina.common.IoSession session)
        throws Exception;

    public void setCodecFactory(org.apache.mina.filter.codec.ProtocolCodecFactory codecFactory);

    public void setDumpTo(String dumpTo);

    public void setForward(String forward);

    public void setResourceLoader(org.springframework.core.io.ResourceLoader loader);

}
```

## Methods inherited from org.apache.mina.common.IoHandlerAdapter:

exceptionCaught , messageReceived , messageSent , sessionClosed , sessionCreated ,  
sessionId , sessionOpened

## Methods inherited from java.lang.Object:

clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 1.1. exceptionCaught(IoSession, Throwable)

```
public void exceptionCaught(org.apache.mina.common.IoSession session,
                           Throwable cause)
        throws Exception;
```

## 1.2. messageReceived(IoSession, Object)

```
public void messageReceived(org.apache.mina.common.IoSession session,
                           Object in);
```

### 1.3. sessionCreated(IoSession)

```
public void sessionCreated(org.apache.mina.common.IoSession session)
    throws Exception;
```

### 1.4. sessionOpened(IoSession)

```
public void sessionOpened(org.apache.mina.common.IoSession session)
    throws Exception;
```

### 1.5. setCodecFactory(ProtocolCodecFactory)

```
public void setCodecFactory(org.apache.mina.filter.codec.ProtocolCodecFactory codecFactory);
```

Setter for property 'codecFactory'.

#### Parameters

codecFactory	Value to set for property 'codecFactory'.
--------------	---

### 1.6. setDumpTo(String)

```
public void setDumpTo(String dumpTo);
```

Setter for property 'dumpTo'.

#### Parameters

dumpTo	Value to set for property 'dumpTo'.
--------	-------------------------------------

### 1.7. setForward(String)

```
public void setForward(String forward);
```

Setter for property 'forward'.

#### Parameters

forward	Value to set for property 'forward'.
---------	--------------------------------------

### 1.8. setResourceLoader(ResourceLoader)

```
public void setResourceLoader(org.springframework.core.io.ResourceLoader loader);
```

**Specified by:** Method `setResourceLoader` in interface `ResourceLoaderAware`

## 2. Class NetworkDumpFilter

Network dump filter, performs raw data and headers dump on message receive

### 2.1. Synopsis

```
public class NetworkDumpFilter extends org.apache.mina.common.IoFilterAdapter {
    // Protected Fields
```

```

protected java.nio.channels.WritableByteChannel headers ;

protected static org.slf4j.Logger log ;

protected java.nio.channels.WritableByteChannel raw ;

// Public Constructors

public NetworkDumpFilter(java.nio.channels.WritableByteChannel headers,
                         java.nio.channels.WritableByteChannel raw);

// Public Methods

public void messageReceived(org.apache.mina.common.IoFilter.NextFilter next,
                           org.apache.mina.common.IoSession session,
                           Object message)
throws Exception;

public void sessionClosed(org.apache.mina.common.IoFilter.NextFilter next,
                        org.apache.mina.common.IoSession session)
throws Exception;

}

```

**Methods inherited from org.apache.mina.common.IoFilterAdapter:** destroy , exceptionCaught , filterClose , filterWrite , init , messageReceived , messageSent , onPostAdd , onPostRemove , onPreAdd , onPreRemove , sessionClosed , sessionCreated , sessionIdle , sessionOpened

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 2.2. NetworkDumpFilter(WritableByteChannel, WritableByteChannel)

```

public NetworkDumpFilter(java.nio.channels.WritableByteChannel headers,
                        java.nio.channels.WritableByteChannel raw);

```

Create network dump filter from given dump channels

### Parameters

headers	Channel to dump headers
raw	Channel to dump raw data

## 2.3. headers

```

protected java.nio.channels.WritableByteChannel headers ;

```

Headers byte channel

## 2.4. log

```

protected static org.slf4j.Logger log ;

```

Logger

## 2.5. raw

```
protected java.nio.channels.WritableByteChannel raw ;
```

Raw data byte channel

## 2.6. messageReceived(IoFilter.NextFilter, IoSession, Object)

```
public void messageReceived(org.apache.mina.common.IoFilter.NextFilter next,
                           org.apache.mina.common.IoSession session,
                           Object message)
                     throws Exception;
```

## 2.7. sessionClosed(IoFilter.NextFilter, IoSession)

```
public void sessionClosed(org.apache.mina.common.IoFilter.NextFilter next,
                         org.apache.mina.common.IoSession session)
                     throws Exception;
```

# 3. Class ProxyFilter

Proxy filter

## 3.1. Synopsis

```
public class ProxyFilter extends org.apache.mina.common.IoFilterAdapter {
    // Public Static Fields

    public static final String FORWARD_KEY = "proxy_forward_key";

    // Protected Fields

    protected static org.slf4j.Logger log;

    protected String name;

    // Public Constructors

    public ProxyFilter(String name);

    // Public Methods

    public void messageReceived(org.apache.mina.common.IoFilter.NextFilter next,
                               org.apache.mina.common.IoSession session,
                               Object message)
                     throws Exception;

    public void sessionClosed(org.apache.mina.common.IoFilter.NextFilter next,
                            org.apache.mina.common.IoSession session)
                     throws Exception;

}
```

**Methods inherited from org.apache.mina.common.IoFilterAdapter:** destroy , exceptionCaught , filterClose , filterWrite , init , messageReceived , messageSent , onPostAdd , onPostRemove , onPreAdd , onPreRemove , sessionClosed , sessionCreated , sessionIdle , sessionOpened

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 3.2. ProxyFilter(String)

```
public ProxyFilter(String name);
```

Create proxy filter with given name

### Parameters

name
------

## 3.3. FORWARD\_KEY

```
public static final String FORWARD_KEY = "proxy_forward_key";
```

Forwarding key constant

## 3.4. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 3.5. name

```
protected String name ;
```

Filter name

## 3.6. messageReceived(IoFilter.NextFilter, IoSession, Object)

```
public void messageReceived(org.apache.mina.common.IoFilter.NextFilter next,
                           org.apache.mina.common.IoSession session,
                           Object message)
                           throws Exception;
```

## 3.7. sessionClosed(IoFilter.NextFilter, IoSession)

```
public void sessionClosed(org.apache.mina.common.IoFilter.NextFilter next,
                           org.apache.mina.common.IoSession session)
                           throws Exception;
```

# 1. Class FlexMessagingService

Service that can execute compatibility Flex messages.

## 1.1. Synopsis

```
public class FlexMessagingService {  
    // Public Static Fields  
  
    public static final String SERVICE_NAME = "flexMessaging";  
  
    // Protected Fields  
  
    protected java.util.Map<java.lang.String, java.lang.Object> endpoints ;  
  
    protected static org.slf4j.Logger log ;  
  
    protected org.red5.server.api.service.IServiceInvoker serviceInvoker ;  
  
    // Public Constructors  
  
    public FlexMessagingService();  
  
    // Public Static Methods  
  
    public static org.red5.compatibility.flex.messaging.messages.ErrorMessage returnError(org.red5.compatibility.flex.messaging.messages.ErrorMessage fault, org.red5.compatibility.flex.messaging.messages.ErrorMessage fault, org.red5.compatibility.flex.messaging.messages.ErrorMessage fault, org.red5.compatibility.flex.messaging.messages.ErrorMessage fault);  
  
    public static org.red5.compatibility.flex.messaging.messages.ErrorMessage returnError(org.red5.compatibility.flex.messaging.messages.ErrorMessage fault, org.red5.compatibility.flex.messaging.messages.ErrorMessage fault, org.red5.compatibility.flex.messaging.messages.ErrorMessage fault, org.red5.compatibility.flex.messaging.messages.ErrorMessage fault, Throwable error);  
  
    // Public Methods  
  
    public org.red5.compatibility.flex.messaging.messages.AsyncMessage handleRequest(org.red5.compatibility.flex.messaging.messages.AsyncMessage request);  
    public org.red5.compatibility.flex.messaging.messages.ErrorMessage handleRequest(org.red5.compatibility.flex.messaging.messages.ErrorMessage request);  
    public org.red5.compatibility.flex.messaging.messages.AsyncMessage handleRequest(org.red5.compatibility.flex.messaging.messages.AsyncMessage request, org.red5.compatibility.flex.messaging.messages.ErrorMessage fault);  
    public org.red5.compatibility.flex.messaging.messages.AsyncMessage handleRequest(org.red5.compatibility.flex.messaging.messages.AsyncMessage request, org.red5.compatibility.flex.messaging.messages.ErrorMessage fault, org.red5.compatibility.flex.messaging.messages.ErrorMessage fault);  
    public void setEndpoints(java.util.Map<java.lang.String, java.lang.Object> endpoints);  
    public void setServiceInvoker(org.red5.server.api.service.IServiceInvoker serviceInvoker);  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. endpoints

```
protected java.util.Map<java.lang.String, java.lang.Object> endpoints ;
```

Configured endpoints.

### 1.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

### 1.4. SERVICE\_NAME

```
public static final String SERVICE_NAME = "flexMessaging";
```

Name of the service.

### 1.5. serviceInvoker

```
protected org.red5.server.api.service.IServiceInvoker serviceInvoker ;
```

Service invoker to use.

### 1.6. handleRequest(AbstractMessage)

```
public org.red5.compatibility.flex.messaging.messages.ErrorMessage handleRequest(org.red5.compatibility.flex.messaging.messages.ErrorMessage msg);
```

Fallback method to handle arbitrary messages.

#### Parameters

msg	
<i>return</i>	

### 1.7. handleRequest(CommandMessage)

```
public org.red5.compatibility.flex.messaging.messages.AsyncMessage handleRequest(org.red5.compatibility.flex.messaging.messages.AsyncMessage msg);
```

Handle command message request.

#### Parameters

msg	
<i>return</i>	

### 1.8. handleRequest(DataMessage)

```
public org.red5.compatibility.flex.messaging.messages.AsyncMessage handleRequest(org.red5.compatibility.flex.messaging.messages.AsyncMessage msg);
```

Handle messages related to shared objects.

#### Parameters

msg	
<i>return</i>	

## 1.9. handleRequest(RemotingMessage)

```
public org.red5.compatibility.flex.messaging.messages.AsyncMessage handleRequest(org.red5.compatibility.flex.messaging.messages.AsyncMessage msg)
```

Handle request coming from `mx:RemoteObject` tags.

### Parameters

msg	
<i>return</i>	

### See Also

Adobe Livedocs (external) [<http://livedocs.adobe.com/flex/2/langref/mx/rpc/remoting/mxml/RemoteObject.html>]

## 1.10. returnError(AbstractMessage, String, String, String)

```
public static org.red5.compatibility.flex.messaging.messages.ErrorMessage returnError(org.red5.compatibility.flex.messaging.messages.ErrorMessage request, String faultCode, String faultString, String faultDetail)
```

Construct error message.

### Parameters

request	
<i>faultCode</i>	
<i>faultString</i>	
<i>faultDetail</i>	
<i>return</i>	

## 1.11. returnError(AbstractMessage, String, String, Throwable)

```
public static org.red5.compatibility.flex.messaging.messages.ErrorMessage returnError(org.red5.compatibility.flex.messaging.messages.ErrorMessage request, String faultCode, String faultString, Throwable error)
```

Construct error message from exception.

### Parameters

request	
<i>faultCode</i>	
<i>faultString</i>	
<i>error</i>	
<i>return</i>	

## 1.12. setEndpoints(Map<String, Object>)

```
public void setEndpoints(java.util.Map<java.lang.String, java.lang.Object> endpoints);
```

Setup available end points.

### Parameters

endPoints	
-----------	--

## 1.13. setServiceInvoker(IServiceInvoker)

```
public void setServiceInvoker(org.red5.server.api.service.IServiceInvoker serviceInvoker);
```

Set the service invoker to use.

### Parameters

serviceInvoker	
----------------	--

## 2. Interface IRemotingCallback

Callback for asynchronous remoting calls.

### 2.1. Synopsis

```
public interface IRemotingCallback {
    // Public Methods

    public void errorReceived(RemotingClient client,
                           String method,
                           Object[] params,
                           Throwable error);

    public void resultReceived(RemotingClient client,
                           String method,
                           Object[] params,
                           Object result);

}
```

## 2.2. errorReceived(RemotingClient, String, Object[], Throwable)

```
public void errorReceived(RemotingClient client,
                           String method,
                           Object[] params,
                           Throwable error);
```

An error occurred while performing the remoting call.

### Parameters

client	Remoting client
method	Remoting method

params	Call parameters
error	Call result

## 2.3. resultReceived(RemotingClient, String, Object[], Object)

```
public void resultReceived(RemotingClient client,
                           String method,
                           Object[] params,
                           Object result);
```

The result of a remoting call has been received.

Parameters	
client	Remoting client
method	Remote method name
params	Call parameters
result	Call result

## 3. Class RemotingClient

Client interface for remoting calls.

### 3.1. Synopsis

```
public class RemotingClient {
// Public Static Fields

    public static final int DEFAULT_TIMEOUT = 30000;

// Protected Fields

    protected java.util.Map<java.lang.String, org.red5.server.net.remoting.RemotingHeader> headers ;
    protected static org.slf4j.Logger log ;
    protected static org.red5.server.pooling.ThreadPool threadPool ;

// Public Constructors

    public RemotingClient();

    public RemotingClient(String url);

    public RemotingClient(String url,
                          int timeout);

// Public Methods

    public void addHeader(String name,
                         boolean required,
                         Object value);

    public Object invokeMethod(String method,
```

```

        Object[] params);

public void invokeMethod(String method,
                        Object[] methodParams,
                        IRemotingCallback callback);

public void removeHeader(String name);

public void resetCredentials();

public void setCredentials(String userid,
                          String password);

public void setThreadPool(org.red5.server.pooling.ThreadPool threadPool);

// Protected Methods

protected void processHeaders(org.apache.mina.common.ByteBuffer in);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 3.2. RemotingClient()

```
public RemotingClient();
```

Dummy constructor used by the spring configuration.

### 3.3. RemotingClient(String)

```
public RemotingClient(String url);
```

Create new remoting client for the given url.

#### Parameters

url	URL to connect to
-----	-------------------

### 3.4. RemotingClient(String, int)

```
public RemotingClient(String url,
                      int timeout);
```

Create new remoting client for the given url and given timeout.

#### Parameters

url	URL to connect to
timeout	Timeout for one request in milliseconds

### 3.5. DEFAULT\_TIMEOUT

```
public static final int DEFAULT_TIMEOUT = 30000;
```

Default timeout to use.

### 3.6. headers

```
protected java.util.Map<java.lang.String, org.red5.server.net.remoting.RemotingHeader> headers ;
```

Headers to send to the server.

### 3.7. log

```
protected static org.slf4j.Logger log ;
```

Logger

### 3.8. threadPool

```
protected static org.red5.server.pooling.ThreadPool threadPool ;
```

Thread pool to use for asynchronous requests.

### 3.9. addHeader(String, boolean, Object)

```
public void addHeader(String name,
                      boolean required,
                      Object value);
```

Send an additional header to the server.

#### Parameters

name	Header name
required	Header required?
value	Header body

### 3.10. invokeMethod(String, Object[])

```
public Object invokeMethod(String method,
                           Object[] params);
```

Invoke a method synchronously on the remoting server.

#### Parameters

method	Method name
params	Parameters passed to method
return	the result of the method call

### 3.11. invokeMethod(String, Object[], IRemotingCallback)

```
public void invokeMethod(String method,
                        Object[] methodParams,
```

```
IRemotingCallback callback);
```

Invoke a method asynchronously on the remoting server.

#### Parameters

method	Method name
methodParams	Parameters passed to method
callback	Callback

### 3.12. processHeaders(ByteBuffer)

```
protected void processHeaders(org.apache.mina.common.ByteBuffer in);
```

Process any headers sent in the response.

#### Parameters

in	Byte buffer with response data
----	--------------------------------

### 3.13. removeHeader(String)

```
public void removeHeader(String name);
```

Stop sending a given header.

#### Parameters

name	Header name
------	-------------

### 3.14. resetCredentials()

```
public void resetCredentials();
```

Stop sending authentication data.

### 3.15. setCredentials(String, String)

```
public void setCredentials(String userid,
                           String password);
```

Send authentication data with each remoting request.

#### Parameters

userid	User identifier
password	Password

### 3.16. setThreadPool(ThreadPool)

```
public void setThreadPool(org.red5.server.pooling.ThreadPool threadPool);
```

Set the thread pool to use for asynchronous requests.

## Parameters

threadPool	The thread pool
------------	-----------------

## 4. Class RemotingClient.RemotingWorker

Worker class that is used for asynchronous remoting calls.

### 4.1. Synopsis

```
public static class RemotingClient.RemotingWorker {
    // Public Constructors

    public RemotingClient.RemotingWorker();

    // Public Methods

    public void executeTask(RemotingClient client,
                           String method,
                           Object[] params,
                           IRemotingCallback callback);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 4.2. executeTask(RemotingClient, String, Object[], IRemotingCallback)

```
public void executeTask(RemotingClient client,
                       String method,
                       Object[] params,
                       IRemotingCallback callback);
```

Execute task.

## Parameters

client	Remoting client
method	Method name
params	Parameters to pass to method on call
callback	Callback

## 5. Class RemotingConnection

Connection class so the Red5 object works in methods invoked through remoting.  
Attributes are stored in the session of the implementing servlet container.

### 5.1. Synopsis

```
public class RemotingConnection implements, org.red5.server.api.remoting.IRemotingConnection {
    // Protected Fields
```

## Package org.?red5.?server.?net.?remoting

```
protected java.util.List<org.red5.server.api.remoting.IRemotingHeader> headers ;  
  
protected org.red5.server.net.remoting.message.RemotingPacket packet ;  
  
protected javax.servlet.http.HttpServletRequest request ;  
  
protected org.red5.server.api.IScope scope ;  
  
protected javax.servlet.http.HttpSession session ;  
  
// Public Constructors  
  
public RemotingConnection(javax.servlet.http.HttpServletRequest request,  
                           org.red5.server.api.IScope scope,  
                           org.red5.server.net.remoting.message.RemotingPacket packet);  
  
// Public Methods  
  
public void addHeader(String name,  
                      Object value);  
  
public void addHeader(String name,  
                      Object value,  
                      boolean mustUnderstand);  
  
public void close();  
  
public boolean connect(org.red5.server.api.IScope scope);  
  
public boolean connect(org.red5.server.api.IScope scope,  
                      Object[] params);  
  
public void dispatchEvent(Object event);  
  
public void dispatchEvent(org.red5.server.api.event.IEvent event);  
  
public Object getAttribute(String name);  
  
public Object getAttribute(String name,  
                           Object defaultValue);  
  
public java.util.Set<java.lang.String> getAttributeNames();  
  
public java.util.Map<java.lang.String, java.lang.Object> getAttributes();  
  
public java.util.Iterator<org.red5.server.api.IBasicScope> getBasicScopes();  
  
public Boolean getBoolAttribute(String name);  
  
public Byte getByteAttribute(String name);  
  
public org.red5.server.api.IClient getClient();  
  
public long getClientBytesRead();  
  
public java.util.Map<java.lang.String, java.lang.Object> getConnectParams();
```

```
public Double getDoubleAttribute(String name);

public long getDroppedMessages();

public org.red5.server.api.IConnection.Encoding getEncoding();

public java.util.Collection<org.red5.server.api.remoting.IRemotingHeader> getHeaders();

public String getHost();

public Integer getIntAttribute(String name);

public int getLastPingTime();

public java.util.List getListAttribute(String name);

public Long getLongAttribute(String name);

public java.util.Map getMapAttribute(String name);

public String getPath();

public long getPendingMessages();

public long getPendingVideoMessages();

public long getReadBytes();

public long getReadMessages();

public String getRemoteAddress();

public java.util.List<java.lang.String> getRemoteAddresses();

public int getRemotePort();

public org.red5.server.api.IScope getScope();

public String getSessionId();

public java.util.Set getSetAttribute(String name);

public Short getShortAttribute(String name);

public String getStringAttribute(String name);

public String getType();

public long getWrittenBytes();

public long getWrittenMessages();

public boolean handleEvent(org.red5.server.api.event.IEvent event);
```

```

public boolean hasAttribute(String name);

public void initialize(org.red5.server.api.IClient client);

public boolean isConnected();

public void notifyEvent(org.red5.server.api.event.IEvent event);

public void ping();

public boolean removeAttribute(String name);

public void removeAttributes();

public void removeHeader(String name);

public boolean setAttribute(String name,
                           Object value);

public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);

public void setAttributes(org.red5.server.api.IAttributeStore values);

public String toString();

// Protected Methods

protected void setPacket(org.red5.server.net.remoting.message.RemotingPacket packet);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 5.2. RemotingConnection(HttpServletRequest, IScope, RemotingPacket)

```

public RemotingConnection(javax.servlet.http.HttpServletRequest request,
                         org.red5.server.api.IScope scope,
                         org.red5.server.net.remoting.message.RemotingPacket packet);

```

Create servlet connection from request and scope.

### Parameters

request	Servlet request
scope	Scope

## 5.3. headers

```

protected java.util.List<org.red5.server.api.remoting.IRemotingHeader> headers ;

```

Headers to be returned to the client.

## 5.4. packet

```
protected org.red5.server.net.remoting.message.RemotingPacket packet ;
```

Remoting packet that triggered the connection.

## 5.5. request

```
protected javax.servlet.http.HttpServletRequest request ;
```

Servlet request

## 5.6. scope

```
protected org.red5.server.api.IScope scope ;
```

Scope

## 5.7. session

```
protected javax.servlet.http.HttpSession session ;
```

Session used to store properties.

## 5.8. addHeader(String, Object)

```
public void addHeader(String name,  
                      Object value);
```

**Specified by:** Method addHeader in interface IRemotingConnection

Tell the client to add a header with all further requests. This is returned to the client as response for the next request received.

## 5.9. addHeader(String, Object, boolean)

```
public void addHeader(String name,  
                      Object value,  
                      boolean mustUnderstand);
```

**Specified by:** Method addHeader in interface IRemotingConnection

Tell the client to add a header with all further requests. This is returned to the client as response for the next request received.

## 5.10. close()

```
public void close();
```

## 5.11. connect(IScope)

```
public boolean connect(org.red5.server.api.IScope scope);
```

## 5.12. connect(IScope, Object[])

```
public boolean connect(org.red5.server.api.IScope scope,  
                      Object[] params);
```

## 5.13. dispatchEvent(IEvent)

```
public void dispatchEvent(org.red5.server.api.event.IEvent event);
```

## 5.14. getAttribute(String)

```
public Object getAttribute(String name);
```

## 5.15. getAttribute(String, Object)

```
public Object getAttribute(String name,  
                           Object defaultValue);
```

## 5.16. getAttributeNames()

```
public java.util.Set<java.lang.String> getAttributeNames();
```

## 5.17. getAttributes()

```
public java.util.Map<java.lang.String, java.lang.Object> getAttributes();
```

## 5.18. getBasicScopes()

```
public java.util.Iterator<org.red5.server.api.IBasicScope> getBasicScopes();
```

## 5.19. getBoolAttribute(String)

```
public Boolean getBoolAttribute(String name);
```

## 5.20. getByteAttribute(String)

```
public Byte getByteAttribute(String name);
```

## 5.21. getClient()

```
public org.red5.server.api.IClient getClient();
```

## 5.22. getClientBytesRead()

```
public long getClientBytesRead();
```

## 5.23. getConnectParams()

```
public java.util.Map<java.lang.String, java.lang.Object> getConnectParams();
```

## 5.24. getDoubleAttribute(String)

```
public Double getDoubleAttribute(String name);
```

## 5.25. getDroppedMessages()

```
public long getDroppedMessages();
```

## 5.26. getEncoding()

```
public org.red5.server.api.IConnection.Encoding getEncoding();
```

Return encoding (AMF0 or AMF3).

Parameters	
<i>return</i>	Encoding, currently AMF0

## 5.27. getHeaders()

```
public java.util.Collection<org.red5.server.api.remoting.IRemotingHeader> getHeaders();
```

**Specified by:** Method `getHeaders` in interface `IRemotingConnection`

Return headers to send.

## 5.28. getHost()

```
public String getHost();
```

## 5.29. getIntAttribute(String)

```
public Integer getIntAttribute(String name);
```

## 5.30. getLastPingTime()

```
public int getLastPingTime();
```

## 5.31. getListAttribute(String)

```
public java.util.List getListAttribute(String name);
```

## 5.32. getLongAttribute(String)

```
public Long getLongAttribute(String name);
```

## 5.33. getMapAttribute(String)

```
public java.util.Map getMapAttribute(String name);
```

## 5.34. getPath()

```
public String getPath();
```

## 5.35. getPendingMessages()

```
public long getPendingMessages();
```

## 5.36. getPendingVideoMessages()

```
public long getPendingVideoMessages();
```

Return pending video messages number.

#### Parameters

return	Pending video messages number
--------	-------------------------------

### 5.37. getReadBytes()

```
public long getReadBytes();
```

### 5.38. getReadMessages()

```
public long getReadMessages();
```

### 5.39. getRemoteAddress()

```
public String getRemoteAddress();
```

### 5.40. getRemoteAddresses()

```
public java.util.List<java.lang.String> getRemoteAddresses();
```

### 5.41. getRemotePort()

```
public int getRemotePort();
```

### 5.42. getScope()

```
public org.red5.server.api.IScope getScope();
```

### 5.43. getSessionId()

```
public String getSessionId();
```

### 5.44. getSetAttribute(String)

```
public java.util.Set getSetAttribute(String name);
```

### 5.45. getShortAttribute(String)

```
public Short getShortAttribute(String name);
```

### 5.46. getStringAttribute(String)

```
public String getStringAttribute(String name);
```

### 5.47. getType()

```
public String getType();
```

### 5.48. getWrittenBytes()

```
public long getWrittenBytes();
```

## 5.49. getWrittenMessages()

```
public long getWrittenMessages();
```

## 5.50. handleEvent(IEvent)

```
public boolean handleEvent(org.red5.server.api.event.IEvent event);
```

## 5.51. hasAttribute(String)

```
public boolean hasAttribute(String name);
```

## 5.52. initialize(IClient)

```
public void initialize(org.red5.server.api.IClient client);
```

## 5.53. isConnected()

```
public boolean isConnected();
```

## 5.54. notifyEvent(IEvent)

```
public void notifyEvent(org.red5.server.api.event.IEvent event);
```

## 5.55. ping()

```
public void ping();
```

## 5.56. removeAttribute(String)

```
public boolean removeAttribute(String name);
```

## 5.57. removeAttributes()

```
public void removeAttributes();
```

## 5.58. removeHeader(String)

```
public void removeHeader(String name);
```

**Specified by:** Method removeHeader in interface IRemotingConnection

Tell the client to no longer send a header with all further requests. This is returned to the client as response for the next request received.

## 5.59. setAttribute(String, Object)

```
public boolean setAttribute(String name,  
                           Object value);
```

## 5.60. setAttributes(IAttributeStore)

```
public void setAttributes(org.red5.server.api.IAttributeStore values);
```

## 5.61. setAttributes(Map<String, Object>)

```
public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);
```

## 5.62. setPacket(RemotingPacket)

```
protected void setPacket(org.red5.server.net.remoting.message.RemotingPacket packet);
```

Update the current packet.

### Parameters

packet

## 5.63. toString()

```
public String toString();
```

Return string representation of the connection.

### Parameters

*return*

# 6. Class RemotingHeader

Remoting header to be sent to a server. Informations about predefined headers can be found at <http://www.osflash.org/amf/envelopes/remoting/headers>

## 6.1. Synopsis

```
public class RemotingHeader implements org.red5.server.api.remoting.IRemotingHeader {
    // Protected Fields

    protected Object data;

    protected String name;

    protected boolean required;

    // Public Constructors

    public RemotingHeader(String name,
                          boolean required,
                          Object data);

    // Public Methods

    public boolean getMustUnderstand();

    public String getName();

    public Object getValue();

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 6.2. RemotingHeader(String, boolean, Object)

```
public RemotingHeader(String name,
                      boolean required,
                      Object data);
```

Create a new header to be sent through remoting.

### Parameters

name	Header name
required	Header required?
data	Header data

## 6.3. data

```
protected Object data ;
```

The actual data of the header.

## 6.4. name

```
protected String name ;
```

The name of the header.

## 6.5. required

```
protected boolean required ;
```

Is this header required?

## 6.6. getMustUnderstand()

```
public boolean getMustUnderstand();
```

**Specified by:** Method getMustUnderstand in interface IRemotingHeader

Return boolean flag if receiver must process header before handling other headers or messages.

## 6.7. getName()

```
public String getName();
```

**Specified by:** Method getName in interface IRemotingHeader

Return name of header.

## 6.8. getValue()

```
public Object getValue();
```

**Specified by:** Method `getValue` in interface `IRemotingHeader`

Return value of header.

# 1. Class RemotingCodecFactory

Factory for remoting codec

## 1.1. Synopsis

```
public class RemotingCodecFactory implements org.red5.server.net.protocol.SimpleProtocolCodecFactory

// Protected Fields

protected RemotingProtocolDecoder decoder ;

protected org.red5.io.object.Deserializer deserializer ;

protected RemotingProtocolEncoder encoder ;

protected org.red5.io.object.Serializer serializer ;

// Public Constructors

public RemotingCodecFactory();

// Public Methods

public org.red5.server.net.protocol.SimpleProtocolDecoder getSimpleDecoder();

public org.red5.server.net.protocol.SimpleProtocolEncoder getSimpleEncoder();

public void init();

public void setDeserializer(org.red5.io.object.Deserializer deserializer);

public void setSerializer(org.red5.io.object.Serializer serializer);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. decoder

```
protected RemotingProtocolDecoder decoder ;
```

Remoting protocol decoder

## 1.3. deserializer

```
protected org.red5.io.object.Deserializer deserializer ;
```

Deserializer

## 1.4. encoder

```
protected RemotingProtocolEncoder encoder ;
```

Remoting protocol encoder

## 1.5. serializer

```
protected org.red5.io.object.Serializer serializer ;
```

Serializers

## 1.6. getSimpleDecoder()

```
public org.red5.server.net.protocol.SimpleProtocolDecoder getSimpleDecoder();
```

**Specified by:** Method getSimpleDecoder in interface SimpleProtocolCodecFactory

Getter for simple decoder.

## 1.7. getSimpleEncoder()

```
public org.red5.server.net.protocol.SimpleProtocolEncoder getSimpleEncoder();
```

**Specified by:** Method getSimpleEncoder in interface SimpleProtocolCodecFactory

Getter for simple encoder.

## 1.8. init()

```
public void init();
```

Initialization, creates and binds encoder and decoder to serializer and deserializer

## 1.9. setDeserializer(Deserializer)

```
public void setDeserializer(org.red5.io.object.Deserializer deserializer);
```

Setter for deserializer.

### Parameters

deserializer	Deserializer.
--------------	---------------

## 1.10. setSerializer(Serializer)

```
public void setSerializer(org.red5.io.object.Serializer serializer);
```

Setter for serializer.

### Parameters

serializer	Sserializer.
------------	--------------

## 2. Class RemotingProtocolDecoder

```
public class RemotingProtocolDecoder implements org.?red5.?server.?net.?protocol.?SimpleProtocolDecoder
// Protected Fields
```

```

protected static org.slf4j.Logger ioLog ;

protected static org.slf4j.Logger log ;

// Public Constructors

public RemotingProtocolDecoder();

// Public Methods

public Object decode(org.red5.server.net.protocol.ProtocolState state,
                     org.apache.mina.common.ByteBuffer in)
    throws Exception;

public java.util.List<java.lang.Object> decodeBuffer(org.red5.server.net.protocol.ProtocolState state,
                                                       org.apache.mina.common.ByteBuffer buffer);

public void dispose(org.apache.mina.common.IoSession session)
    throws Exception;

public void setDeserializer(org.red5.io.object.Deserializer deserializer);

// Protected Methods

protected java.util.List<org.red5.server.net.remoting.message.RemotingCall> decodeCalls(org.apache.mina.common.ByteBuffer in);

protected java.util.Map<java.lang.String, java.lang.Object> readHeaders(org.apache.mina.common.ByteBuffer in);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

## 2.1. ioLog

```
protected static org.slf4j.Logger ioLog ;
```

I/O logger

## 2.2. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 2.3. decode(ProtocolState, ByteBuffer)

```
public Object decode(org.red5.server.net.protocol.ProtocolState state,
                     org.apache.mina.common.ByteBuffer in)
    throws Exception;
```

**Specified by:** Method decode in interface SimpleProtocolDecoder

## 2.4. decodeBuffer(ProtocolState, ByteBuffer)

```
public java.util.List<java.lang.Object> decodeBuffer(org.red5.server.net.protocol.ProtocolState state,
```

```
org.apache.mina.common.ByteBuffer buffer);
```

**Specified by:** Method decodeBuffer in interface SimpleProtocolDecoder

Decode all available objects in buffer.

## 2.5. decodeCalls(ByteBuffer)

```
protected java.util.List<org.red5.server.net.remoting.message.RemotingCall> decodeCalls(org.apache.mina.common.ByteBuffer buffer);
```

Decode calls.

### Parameters

in	Input data as byte buffer
<i>return</i>	List of pending calls

## 2.6. dispose(IoSession)

```
public void dispose(org.apache.mina.common.IoSession session)
throws Exception;
```

Disposes session. Not yet implemented.

### Parameters

session	Session to dispose
---------	--------------------

### Exception

Any exception can be raised on disposal

## 2.7. readHeaders(ByteBuffer)

```
protected java.util.Map<java.lang.String, java.lang.Object> readHeaders(org.apache.mina.common.ByteBuffer buffer);
```

Read remoting headers.

### Parameters

in	Input data as byte buffer
----	---------------------------

## 2.8. setDeserializer(Deserializer)

```
public void setDeserializer(org.red5.io.object.Deserializer deserializer);
```

Setter for deserializer.

### Parameters

deserializer	Deserializer
--------------	--------------

## 3. Class RemotingProtocolEncoder

Remoting protocol encoder.

### 3.1. Synopsis

```
public class RemotingProtocolEncoder extends org.red5.server.net.protocol.BaseProtocolEncoder
    implements org.red5.server.net.protocol.SimpleProtocolEncoder {
    // Protected Fields

    protected static org.slf4j.Logger ioLog;

    protected static org.slf4j.Logger log;

    // Public Constructors

    public RemotingProtocolEncoder();

    // Public Methods

    public void dispose(org.apache.mina.common.IoSession ioSession)
        throws Exception;

    public org.apache.mina.common.ByteBuffer encode(org.red5.server.net.protocol.ProtocolState state,
                                                    Object message)
        throws Exception;

    public void setSerializer(org.red5.io.object.Serializer serializer);

}
```

**Methods inherited from org.red5.server.net.protocol.BaseProtocolEncoder:**

generateErrorResult

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 3.2. ioLog

```
protected static org.slf4j.Logger ioLog;
```

I/O logger

### 3.3. log

```
protected static org.slf4j.Logger log;
```

Logger

### 3.4. dispose(IoSession)

```
public void dispose(org.apache.mina.common.IoSession ioSession)
    throws Exception;
```

Dispose I/O session, not implemented yet.

Parameters

ioSession	I/O session
-----------	-------------

Exception

Exception

### 3.5. encode(ProtocolState, Object)

```
public org.apache.mina.common.ByteBuffer encode(org.red5.server.net.protocol.ProtocolState state,  
                                              Object message)  
throws Exception;
```

**Specified by:** Method encode in interface SimpleProtocolEncoder

Encodes object with given protocol state to byte buffer

### 3.6. setSerializer(Serializer)

```
public void setSerializer(org.red5.io.object.Serializer serializer);
```

Setter for serializer.

#### Parameters

serializer	New serializer
------------	----------------

# 1. Class RemotingCall

Remoting method call, specific pending call.

## 1.1. Synopsis

```
public class RemotingCall extends org.red5.server.service.PendingCall {  
    // Public Static Fields  
  
    public static final String HANDLER_ERROR = "/onStatus";  
  
    public static final String HANDLER_SUCCESS = "/onResult";  
  
    // Public Fields  
  
    public String clientCallback;  
  
    public boolean isAMF3;  
  
    public boolean isMessaging;  
  
    // Public Constructors  
  
    public RemotingCall(String serviceName,  
                        String serviceMethod,  
                        Object[] args,  
                        String callback,  
                        boolean isAMF3,  
                        boolean isMessaging);  
  
    // Public Methods  
  
    public String getClientResponse();  
  
    public Object getClientResult();  
  
    public void setClientCallback(String clientCallback);  
}
```

**Methods inherited from org.red5.server.service.PendingCall:** getCallbacks , getResult , readExternal , registerCallback , setResult , unregisterCallback , writeExternal

**Methods inherited from org.red5.server.service.Call:** getArguments , getException , getServiceMethodName , getServiceName , getStatus , isSuccess , setArguments , setException , setServiceMethodName , setServiceName , setStatus , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

**Fields inherited from org.red5.server.service.Call:** arguments , exception , serviceMethodName , serviceName , status , STATUS\_ACCESS\_DENIED , STATUS\_APP\_SHUTTING\_DOWN , STATUS\_GENERAL\_EXCEPTION , STATUS\_INVOCATION\_EXCEPTION , STATUS\_METHOD\_NOT\_FOUND , STATUS\_PENDING , STATUS\_SERVICE\_NOT\_FOUND , STATUS\_SUCCESS\_NULL , STATUS\_SUCCESS\_RESULT , STATUS\_SUCCESS\_VOID

## 1.2. RemotingCall(String, String, Object[], String, boolean, boolean)

```
public RemotingCall(String serviceName,
                    String serviceMethod,
                    Object[] args,
                    String callback,
                    boolean isAMF3,
                    boolean isMessaging);
```

Create remoting call from service name, method name, list of arguments and callback name.

Parameters	
serviceName	Service name
serviceMethod	Service method name
args	Parameters passed to method
callback	Name of client callback
isAMF3	Does the client support AMF3?
isMessaging	Is this a Flex messaging request?

## 1.3. clientCallback

```
public String clientCallback ;
```

Client callback name

## 1.4. HANDLER\_ERROR

```
public static final String HANDLER_ERROR = "/onStatus";
```

Handler error postfix constant

## 1.5. HANDLER\_SUCCESS

```
public static final String HANDLER_SUCCESS = "/onResult";
```

Handler success postfix constant

## 1.6. getClientResponse()

```
public String getClientResponse();
```

Getter for client response.

Parameters	
return	Client response

## 1.7. getClientResult()

```
public Object getClientResult();
```

Getter for client result.

### Parameters

<i>return</i>	Client result
---------------	---------------

## 1.8. setClientCallback(String)

```
public void setClientCallback(String clientCallback);
```

Setter for client callback.

### Parameters

clientCallback	Client callback
----------------	-----------------

## 2. Class RemotingPacket

Packet of remote calls. Used by RemoteProtocolDecoder.

### 2.1. Synopsis

```
public class RemotingPacket {
    // Protected Fields

    protected java.util.List<org.red5.server.net.remoting.message.RemotingCall> calls ;
    protected java.nio.ByteBuffer data ;
    protected java.util.Map<java.lang.String, java.lang.Object> headers ;
    protected javax.servlet.http.HttpServletRequest request ;
    protected String scopePath ;

    // Public Constructors

    public RemotingPacket(java.util.Map<java.lang.String, java.lang.Object> headers,
                         java.util.List<org.red5.server.net.remoting.message.RemotingCall> calls);

    // Public Methods

    public java.util.List<org.red5.server.net.remoting.message.RemotingCall> getCalls();
    public org.red5.server.api.IConnection.Encoding getEncoding();
    public java.util.Map<java.lang.String, java.lang.Object> getHeaders();
    public String getScopePath();
    public void setScopePath(String path);
}
```

```
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. RemotingPacket(Map<String, Object>, List<RemotingCall>)

```
public RemotingPacket(java.util.Map<java.lang.String, java.lang.Object> headers,
                      java.util.List<org.red5.server.net.remoting.message.RemotingCall> calls);
```

Create remoting packet from list of pending calls

### Parameters

calls	List of call objects
-------	----------------------

## 2.3. calls

```
protected java.util.List<org.red5.server.net.remoting.message.RemotingCall> calls;
```

List of calls

## 2.4. data

```
protected java.nio.ByteBuffer data;
```

Byte buffer data

## 2.5. headers

```
protected java.util.Map<java.lang.String, java.lang.Object> headers;
```

Headers sent with request.

## 2.6. request

```
protected javax.servlet.http.HttpServletRequest request;
```

HTTP request object

## 2.7. scopePath

```
protected String scopePath;
```

Scope path

## 2.8. getCalls()

```
public java.util.List<org.red5.server.net.remoting.message.RemotingCall> getCalls();
```

Getter for calls.

**Parameters**

<i>return</i>	List of remote calls
---------------	----------------------

**2.9. getEncoding()**

```
public org.red5.server.api.IConnection.Encoding getEncoding();
```

Return the encoding of the included calls.

**Parameters**

<i>return</i>
---------------

**2.10. getHeaders()**

```
public java.util.Map<java.lang.String, java.lang.Object> getHeaders();
```

Get the headers sent with the request.

**Parameters**

<i>return</i>
---------------

**2.11. getScopePath()**

```
public String getScopePath();
```

Getter for property scope path.

**Parameters**

<i>return</i>	Scope path to set
---------------	-------------------

**2.12. setScopePath(String)**

```
public void setScopePath(String path);
```

Setter for scope path.

**Parameters**

path	Value to set for property 'scopePath'.
------	--

# 1. Class BaseRTMPClientHandler

Base class for clients (RTMP and RTMPT)

## 1.1. Synopsis

```
public abstract class BaseRTMPClientHandler extends org.red5.server.net.rtmp.BaseRTMPHandler {  
    // Protected Fields  
  
    protected org.red5.server.net.rtmp.codec.RTMPCodecFactory codecFactory ;  
  
    protected RTMPClientConnManager connManager ;  
  
    protected Object[] connectArguments ;  
  
    protected org.red5.server.api.service.IPendingServiceCallback connectCallback ;  
  
    protected java.util.Map<java.lang.String, java.lang.Object> connectionParams ;  
  
    protected org.red5.server.api.service.IServiceInvoker serviceInvoker ;  
  
    protected Object serviceProvider ;  
  
    protected java.util.Map<java.lang.String, org.red5.server.so.ClientSharedObject> sharedObjects ;  
  
    protected org.red5.server.api.event.IEventDispatcher streamEventDispatcher ;  
  
    // Protected Constructors  
  
    protected BaseRTMPClientHandler();  
  
    // Public Methods  
  
    public void connect(String server,  
                       int port,  
                       String application);  
  
    public void connect(String server,  
                       int port,  
                       String application,  
                       org.red5.server.api.service.IPendingServiceCallback connectCallback);  
  
    public void connect(String server,  
                       int port,  
                       java.util.Map<java.lang.String, java.lang.Object> connectionParams);  
  
    public void connect(String server,  
                       int port,  
                       java.util.Map<java.lang.String, java.lang.Object> connectionParams,  
                       org.red5.server.api.service.IPendingServiceCallback connectCallback);  
  
    public void connect(String server,  
                       int port,  
                       java.util.Map<java.lang.String, java.lang.Object> connectionParams,  
                       org.red5.server.api.service.IPendingServiceCallback connectCallback,  
                       Object[] connectCallArguments);  
  
    public void connectionClosed(RTMPConnection conn,
```

## Package org.red5.server.net.rtmp

```
    org.red5.server.net.rtmp.codec.RTMP state);  
  
    public void connectionOpened(RTMPConnection conn,  
                                org.red5.server.net.rtmp.codec.RTMP state);  
  
    public void createStream(org.red5.server.api.service.IPendingServiceCallback callback);  
  
    public void disconnect();  
  
    public org.red5.server.net.rtmp.codec.RTMCCodecFactory getCodecFactory();  
  
    public RTMPClientConnManager getConnManager();  
  
    public synchronized org.red5.server.api.so.IClientSharedObject getSharedObject(String name,  
                                boolean persistent);  
  
    public void handleException(Throwable throwable);  
  
    public void invoke(String method,  
                      Object[] params,  
                      org.red5.server.api.service.IPendingServiceCallback callback);  
  
    public void invoke(String method,  
                      org.red5.server.api.service.IPendingServiceCallback callback);  
  
    public java.util.Map<java.lang.String, java.lang.Object> makeDefaultConnectionParams(String server  
                                                int port,  
                                                String applic  
  
    public void play(int streamId,  
                    String name,  
                    int start,  
                    int length);  
  
    public void publish(int streamId,  
                      String name,  
                      String mode,  
                      INetStreamEventHandler handler);  
  
    public void publishStreamData(int streamId,  
                                org.red5.server.messaging.IMessage message);  
  
    public void setCodecFactory(org.red5.server.net.rtmp.codec.RTMCCodecFactory factory);  
  
    public void setConnectionClosedHandler(Runnable connectionClosedHandler);  
  
    public void setExceptionHandler(ClientExceptionHandler exceptionHandler);  
  
    public void setServiceProvider(Object serviceProvider);  
  
    public void setStreamEventDispatcher(org.red5.server.api.event.IEventDispatcher streamEventDispatcher);  
  
// Protected Methods  
  
    protected void onChunkSize(RTMPConnection conn,  
                             Channel channel,  
                             org.red5.server.net.rtmp.message.Header source,  
                             org.red5.server.net.rtmp.event.ChunkSize chunkSize);
```

```

protected void onInvoke(RTMPConnection conn,
                      Channel channel,
                      org.red5.server.net.rtmp.message.Header source,
                      org.red5.server.net.rtmp.event.Notify invoke,
                      org.red5.server.net.rtmp.codec.RTMP rtmp);

protected void onPing(RTMPConnection conn,
                     Channel channel,
                     org.red5.server.net.rtmp.message.Header source,
                     org.red5.server.net.rtmp.event.Ping ping);

protected void onSharedObject(RTMPConnection conn,
                            Channel channel,
                            org.red5.server.net.rtmp.message.Header source,
                            org.red5.server.so.SharedObjectMessage object);

protected abstract void startConnector(String server,
                                       int port);

}

```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?RTMPClient , org.?red5.?server.?net.?rtmpt.?RTMPTClient

#### Methods inherited from org.red5.server.net.rtmp.BaseRTMPHandler:

connectionClosed , connectionOpened , getHostname , getStreamId ,  
 handlePendingCallResult , messageReceived , messageSent , onChunkSize , onInvoke ,  
 onPing , onSharedObject , onStreamBytesRead , setApplicationContext

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
 , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.BaseRTMPHandler:** appCtx , log

## 1.2. connectArguments

```
protected Object[] connectArguments ;
```

Connect call arguments

## 1.3. connectCallback

```
protected org.red5.server.api.service.IPendingServiceCallback connectCallback ;
```

Connection callback

## 1.4. connectionParams

```
protected java.util.Map<java.lang.String, java.lang.Object> connectionParams ;
```

Connection parameters

## 1.5. serviceInvoker

```
protected org.red5.server.api.service.IServiceInvoker serviceInvoker ;
```

Service invoker

## 1.6. serviceProvider

```
protected Object serviceProvider ;
```

Service provider

## 1.7. sharedObjects

```
protected java.util.Map<java.lang.String, org.red5.server.so.ClientSharedObject> sharedObjects ;
```

Shared objects map

## 1.8. connect(String, int, Map<String, Object>)

```
public void connect(String server,
                    int port,
                    java.util.Map<java.lang.String, java.lang.Object> connectionParams);
```

Connect RTMP client to server via given port and with given connection parameters

Parameters	
server	Server
port	Connection port
connectionParams	Connection parameters

## 1.9. connect(String, int, Map<String, Object>, IPendingServiceCallback)

```
public void connect(String server,
                    int port,
                    java.util.Map<java.lang.String, java.lang.Object> connectionParams,
                    org.red5.server.api.service.IPendingServiceCallback connectCallback);
```

Connect RTMP client to server's application via given port

Parameters	
server	Server
port	Connection port
connectionParams	Connection parameters
connectCallback	Connection callback

## 1.10. connect(String, int, Map<String, Object>, IPendingServiceCallback, Object[])

```
public void connect(String server,
```

```
int port,
java.util.Map<java.lang.String, java.lang.Object> connectionParams,
org.red5.server.api.service.IPendingServiceCallback connectCallback,
Object[] connectCallArguments);
```

Connect RTMP client to server's application via given port

Parameters	
server	Server
port	Connection port
connectionParams	Connection parameters
connectCallback	Connection callback
connectCallArguments	Arguments for 'connect' call

## 1.11. **connect(String, int, String)**

```
public void connect(String server,
                    int port,
                    String application);
```

Connect RTMP client to server's application via given port

Parameters	
server	Server
port	Connection port
application	Application at that server

## 1.12. **connect(String, int, String, IPendingServiceCallback)**

```
public void connect(String server,
                    int port,
                    String application,
                    org.red5.server.api.service.IPendingServiceCallback connectCallback);
```

Connect RTMP client to server's application via given port with given connection callback

Parameters	
server	Server
port	Connection port
application	Application at that server
connectCallback	Connection callback

## 1.13. **connectionOpened(RTMPConnection, RTMP)**

```
public void connectionOpened(RTMPConnection conn,
                            org.red5.server.net.rtmp.codec.RTMP state);
```

## 1.14. disconnect()

```
public void disconnect();
```

Disconnect the first connection in the connection map

## 1.15. getCodecFactory()

```
public org.red5.server.net.rtmp.codec.RTMPCodecFactory getCodecFactory();
```

Getter for codec factory

### Parameters

<i>return</i>	Codec factory
---------------	---------------

## 1.16. getSharedObject(String, boolean)

```
public synchronized org.red5.server.api.so.IClientSharedObject getSharedObject(String name,
                                                                           boolean persistent);
```

Connect to client shared object.

### Parameters

name	Client shared object name
persistent	SO persistence flag
<i>return</i>	Client shared object instance

## 1.17. invoke(String, IPendingServiceCallback)

```
public void invoke(String method,
                   org.red5.server.api.service.IPendingServiceCallback callback);
```

Invoke a method on the server.

### Parameters

method	Method name
callback	Callback handler

## 1.18. invoke(String, Object[], IPendingServiceCallback)

```
public void invoke(String method,
                   Object[] params,
                   org.red5.server.api.service.IPendingServiceCallback callback);
```

Invoke a method on the server and pass parameters.

### Parameters

method	Method
--------	--------

params	Method call parameters
callback	Callback object

## 1.19. makeDefaultConnectionParams(String, int, String)

```
public java.util.Map<java.lang.String, java.lang.Object> makeDefaultConnectionParams(String server,
                                                                                     int port,
                                                                                     String applicat
```

### Parameters

server	Server
port	Connection port
application	Application at that server
return	default connection parameters

## 1.20. onChunkSize(RTMPConnection, Channel, Header, ChunkSize)

```
protected void onChunkSize(RTMPConnection conn,
                           Channel channel,
                           org.red5.server.net.rtmp.message.Header source,
                           org.red5.server.net.rtmp.event.ChunkSize chunkSize);
```

## 1.21. onInvoke(RTMPConnection, Channel, Header, Notify, RTMP)

```
protected void onInvoke(RTMPConnection conn,
                       Channel channel,
                       org.red5.server.net.rtmp.message.Header source,
                       org.red5.server.net.rtmp.event.Notify invoke,
                       org.red5.server.net.rtmp.codec.RTMP rtmp);
```

## 1.22. onPing(RTMPConnection, Channel, Header, Ping)

```
protected void onPing(RTMPConnection conn,
                      Channel channel,
                      org.red5.server.net.rtmp.message.Header source,
                      org.red5.server.net.rtmp.event.Ping ping);
```

## 1.23. onSharedObject(RTMPConnection, Channel, Header, SharedObjectMessage)

```
protected void onSharedObject(RTMPConnection conn,
                            Channel channel,
                            org.red5.server.net.rtmp.message.Header source,
                            org.red5.server.so.SharedObjectMessage object);
```

## 1.24. setCodecFactory(RTMPCodecFactory)

```
public void setCodecFactory(org.red5.server.net.rtmp.codec.RTMPCodecFactory factory);
```

Setter for codec factory

### Parameters

factory	Codec factory to use
---------	----------------------

## 1.25. setServiceProvider(Object)

```
public void setServiceProvider(Object serviceProvider);
```

Register object that provides methods that can be called by the server.

### Parameters

serviceProvider	Service provider
-----------------	------------------

## 1.26. setStreamEventDispatcher(IEventDispatcher)

```
public void setStreamEventDispatcher(org.red5.server.api.event.IEventDispatcher streamEventDispatcher);
```

Setter for stream event dispatcher (useful for saving playing stream to file)

### Parameters

streamEventDispatcher	
-----------------------	--

## 1.27. startConnector(String, int)

```
protected abstract void startConnector(String server,
                                      int port);
```

Start network connection to server

### Parameters

server	Server
port	Connection port

## 2. Class BaseRTMPHandler

Base class for all RTMP handlers.

### 2.1. Synopsis

```
public abstract class BaseRTMPHandler implements org.?red5.?server.?net.?rtmp.?IRTMPHandler, org.?red5.?server.?net.?rtmp.?IStreamHandler {
    // Protected Fields

    protected org.springframework.context.ApplicationContext appCtx;

    protected static org.slf4j.Logger log;

    // Public Constructors

    public BaseRTMPHandler();

    // Public Static Methods

    public static int getStreamId();
```

## Package org.?red5.?server.?net.?rtmp

```
// Public Methods

    public void connectionClosed(RTMPConnection conn,
                                org.red5.server.net.rtmp.codec.RTMP state);

    public void connectionOpened(RTMPConnection conn,
                                org.red5.server.net.rtmp.codec.RTMP state);

    public void messageReceived(RTMPConnection conn,
                               org.red5.server.net.protocol.ProtocolState state,
                               Object in)
        throws Exception;

    public void messageSent(RTMPConnection conn,
                           Object message);

    public void setApplicationContext(org.springframework.context.ApplicationContext appCtx)
        throws BeansException;

// Protected Methods

    protected String getHostname(String url);

    protected void handlePendingCallResult(RTMPConnection conn,
                                         org.red5.server.net.rtmp.event.Notify invoke);

    protected abstract void onChunkSize(RTMPConnection conn,
                                       Channel channel,
                                       org.red5.server.net.rtmp.message.Header source,
                                       org.red5.server.net.rtmp.event.ChunkSize chunkSize);

    protected abstract void onInvoke(RTMPConnection conn,
                                    Channel channel,
                                    org.red5.server.net.rtmp.message.Header source,
                                    org.red5.server.net.rtmp.event.Notify invoke,
                                    org.red5.server.net.rtmp.codec.RTMP rtmp);

    protected abstract void onPing(RTMPConnection conn,
                                 Channel channel,
                                 org.red5.server.net.rtmp.message.Header source,
                                 org.red5.server.net.rtmp.event.Ping ping);

    protected abstract void onSharedObject(RTMPConnection conn,
                                         Channel channel,
                                         org.red5.server.net.rtmp.message.Header source,
                                         org.red5.server.so.SharedObjectMessage object);

    protected void onStreamBytesRead(RTMPConnection conn,
                                    Channel channel,
                                    org.red5.server.net.rtmp.message.Header source,
                                    org.red5.server.net.rtmp.event.BytesRead streamBytesRead);

}
```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?BaseRTMPClientHandler , org.?red5.?server.?net.?rtmp.?RTMPHandler

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 2.2. appCtx

```
protected org.springframework.context.ApplicationContext appCtx ;
```

Application context

## 2.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 2.4. connectionClosed(RTMPConnection, RTMP)

```
public void connectionClosed(RTMPConnection conn,
                             org.red5.server.net.rtmp.codec.RTMP state);
```

**Specified by:** Method connectionClosed in interface IRTMPHandler

Connection closed

## 2.5. connectionOpened(RTMPConnection, RTMP)

```
public void connectionOpened(RTMPConnection conn,
                            org.red5.server.net.rtmp.codec.RTMP state);
```

**Specified by:** Method connectionOpened in interface IRTMPHandler

Connection open event

## 2.6. getHostname(String)

```
protected String getHostname(String url);
```

Return hostname for URL.

### Parameters

url	URL
return	Hostname from that URL

## 2.7. getStreamId()

```
public static int getStreamId();
```

Getter for stream ID.

### Parameters

return	Stream ID
--------	-----------

## 2.8. handlePendingCallResult(RTMPConnection, Notify)

```
protected void handlePendingCallResult(RTMPConnection conn,
```

```
org.red5.server.net.rtmp.event.Notify invoke);
```

Handler for pending call result. Dispatches results to all pending call handlers.

#### Parameters

conn	Connection
invoke	Pending call result event context

## 2.9. messageReceived(RTMPConnection, ProtocolState, Object)

```
public void messageReceived(RTMPConnection conn,
                            org.red5.server.net.protocol.ProtocolState state,
                            Object in)
throws Exception;
```

**Specified by:** Method messageReceived in interface IRTMPHandler

Message received

## 2.10. messageSent(RTMPConnection, Object)

```
public void messageSent(RTMPConnection conn,
                        Object message);
```

**Specified by:** Method messageSent in interface IRTMPHandler

Message sent

## 2.11. onChunkSize(RTMPConnection, Channel, Header, ChunkSize)

```
protected abstract void onChunkSize(RTMPConnection conn,
                                    Channel channel,
                                    org.red5.server.net.rtmp.message.Header source,
                                    org.red5.server.net.rtmp.event.ChunkSize chunkSize);
```

Chunk size change event handler. Abstract, to be implemented in subclasses.

#### Parameters

conn	Connection
channel	Channel
source	Header
chunkSize	New chunk size

## 2.12. onInvoke(RTMPConnection, Channel, Header, Notify, RTMP)

```
protected abstract void onInvoke(RTMPConnection conn,
                                Channel channel,
                                org.red5.server.net.rtmp.message.Header source,
                                org.red5.server.net.rtmp.event.Notify invoke,
                                org.red5.server.net.rtmp.codec.RTMP rtmp);
```

Invocation event handler.

Parameters	
conn	Connection
channel	Channel
source	Header
invoke	Invocation event context
rtmp	RTMP connection state

## 2.13. onPing(RTMPConnection, Channel, Header, Ping)

```
protected abstract void onPing(RTMPConnection conn,
                               Channel channel,
                               org.red5.server.net.rtmp.message.Header source,
                               org.red5.server.net.rtmp.event.Ping ping);
```

Ping event handler.

Parameters	
conn	Connection
channel	Channel
source	Header
ping	Ping event context

## 2.14. onSharedObject(RTMPConnection, Channel, Header, SharedObjectMessage)

```
protected abstract void onSharedObject(RTMPConnection conn,
                                      Channel channel,
                                      org.red5.server.net.rtmp.message.Header source,
                                      org.red5.server.so.SharedObjectMessage object);
```

Shared object event handler.

Parameters	
conn	Connection
channel	Channel
source	Header
object	Shared object event context

## 2.15. onStreamBytesRead(RTMPConnection, Channel, Header, BytesRead)

```
protected void onStreamBytesRead(RTMPConnection conn,
                                Channel channel,
                                org.red5.server.net.rtmp.message.Header source,
```

```
org.red5.server.net.rtmp.event.BytesRead streamBytesRead);
```

Stream bytes read event handler.

Parameters	
conn	Connection
channel	Channel
source	Header
streamBytesRead	Bytes read event context

## 2.16. setApplicationContext(ApplicationContext)

```
public void setApplicationContext(org.springframework.context.ApplicationContext appCtx)
    throws BeansException;
```

**Specified by:** Method `setApplicationContext` in interface `ApplicationContextAware`

## 3. Class Channel

Identified connection that transfers packets.

### 3.1. Synopsis

```
public class Channel {
    // Protected Fields

    protected static org.slf4j.Logger log;

    // Public Constructors

    public Channel(RTMPConnection conn,
                   int channelId);

    // Public Methods

    public void close();

    public int getId();

    public void sendStatus(org.red5.server.net.rtmp.status.Status status);

    public void write(org.red5.server.net.rtmp.event.IRTMPEvent event);

    // Protected Methods

    protected RTMPConnection getConnection();

}
```

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

### 3.2. Channel(RTMPConnection, int)

```
public Channel(RTMPConnection conn,
              int channelId);
```

Creates channel from connection and channel id

#### Parameters

conn	Connection
channelId	Channel id

### 3.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

### 3.4. close()

```
public void close();
```

Closes channel with this id on RTMP connection.

### 3.5. getConnection()

```
protected RTMPConnection getConnection();
```

Getter for RTMP connection.

#### Parameters

return	RTMP connection
--------	-----------------

### 3.6. getId()

```
public int getId();
```

Getter for id.

#### Parameters

return	Channel ID
--------	------------

### 3.7. sendStatus(Status)

```
public void sendStatus(org.red5.server.net.rtmp.status.Status status);
```

Sends status notification.

#### Parameters

status	Status
--------	--------

### 3.8. write(IRTMPEvent)

```
public void write(org.red5.server.net.rtmp.event.IRTMPEvent event);
```

Writes packet from event data to RTMP connection.

#### Parameters

event	Event data
-------	------------

## 4. Interface ClientExceptionHandler

Client connection exception handler

### 4.1. Synopsis

```
public interface ClientExceptionHandler {
    // Public Methods

    public void handleException(Throwable throwable);

}
```

## 5. Class DeferredResult

Can be returned to delay returning the result of invoked methods.

### 5.1. Synopsis

```
public class DeferredResult {
    // Protected Fields

    protected static org.slf4j.Logger log;

    // Public Constructors

    public DeferredResult();

    // Public Methods

    public void setResult(Object result);

    public boolean wasSent();

    // Protected Methods

    protected void setChannel(Channel channel);

    protected void setInvokeId(int id);

    protected void setServiceCall(org.red5.server.api.service.IPendingServiceCall call);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 5.2. log

```
protected static org.slf4j.Logger log;
```

Logger

## 5.3. setChannel(Channel)

```
protected void setChannel(Channel channel);
```

Setter for channel.

### Parameters

channel	Channel
---------	---------

## 5.4. setInvokeId(int)

```
protected void setInvokeId(int id);
```

Setter for invoke Id.

### Parameters

id	Invocation object identifier
----	------------------------------

## 5.5. setResult(Object)

```
public void setResult(Object result);
```

Set the result of a method call and send to the caller.

### Parameters

result	deferred result of the method call
--------	------------------------------------

## 5.6. setServiceCall(IPendingServiceCall)

```
protected void setServiceCall(org.red5.server.api.service.IPendingServiceCall call);
```

Setter for service call.

### Parameters

call	Service call
------	--------------

## 5.7. wasSent()

```
public boolean wasSent();
```

Check if the result has been sent to the client.

#### Parameters

<i>return</i>	true if the result has been sent, otherwise false
---------------	---

## 6. Class EdgeRTMPHandler

```
public class EdgeRTMPHandler extends org.?red5.?server.?net.?rtmp.?RTMPHandler {
// Public Constructors

    public EdgeRTMPHandler();

// Public Methods

    public void connectionClosed(RTMPConnection conn,
                                org.red5.server.net.rtmp.codec.RTMP state);

    public void messageReceived(RTMPConnection conn,
                               org.red5.server.net.protocol.ProtocolState state,
                               Object in)
    throws Exception;

    public void messageSent(RTMPConnection conn,
                           Object message);

    public void setMRTMPManager(org.red5.server.net.mrtmp.IMRTMPManager mrtmpManager);

// Protected Methods

    protected boolean checkPermission(RTMPConnection conn);

    protected void forwardPacket(RTMPConnection conn,
                                org.red5.server.net.rtmp.message.Packet packet);

    protected void handleConnect(RTMPConnection conn,
                                Channel channel,
                                org.red5.server.net.rtmp.message.Header header,
                                org.red5.server.net.rtmp.event.Invoke invoke,
                                org.red5.server.net.rtmp.codec.RTMP rtmp);

    protected void onPing(RTMPConnection conn,
                         Channel channel,
                         org.red5.server.net.rtmp.message.Header source,
                         org.red5.server.net.rtmp.event.Ping ping);

    protected void sendConnectMessage(RTMPConnection conn);

}
```

**Methods inherited from org.red5.server.net.rtmp.RTMPHandler:** getStatus , invokeCall , onChunkSize , onInvoke , onPing , onSharedObject , setServer , setStatusObjectService

**Methods inherited from org.red5.server.net.rtmp.BaseRTMPHandler:**

connectionClosed , connectionOpened , getHostname , getStreamId , handlePendingCallResult , messageReceived , messageSent , onStreamBytesRead , setApplicationContext

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.RTMPHandler:** log , server , statusObjectService

**Fields inherited from org.red5.server.net.rtmp.BaseRTMPHandler:** appCtx

## 6.1. onPing(RTMPConnection, Channel, Header, Ping)

```
protected void onPing(RTMPConnection conn,
                      Channel channel,
                      org.red5.server.net.rtmp.message.Header source,
                      org.red5.server.net.rtmp.event.Ping ping);
```

Pass through all Ping events to origin except ping/pong

## 7. Class EdgeRTMPMinaConnection

```
public class EdgeRTMPMinaConnection extends org.?red5.?server.?net.?rtmp.?RTMPMinaConnection {
    // Public Constructors

    public EdgeRTMPMinaConnection();

    // Public Methods

    public void close();

    public void setMrtmpManager(org.red5.server.net.mrtmp.IMRTMPEdgeManager mrtmpManager);

    // Protected Methods

    protected void startWaitForHandshake(org.red5.server.api.scheduling.ISchedulingService service);

}
```

**Methods inherited from org.red5.server.net.rtmp.RTMPMinaConnection:** close , connect , getIoSession , getPendingMessages , getReadBytes , getWrittenBytes , invokeMethod , isConnected , onInactive , rawWrite , setIoSession , write

**Methods inherited from org.red5.server.net.rtmp.RTMPConnection:**

```
addClientStream , closeChannel , createOutputStream , createStreamName ,
deleteStreamById , getBandwidthConfigure , getChannel , getClientBytesRead ,
getEncoding , getId , getInvokeId , getLastPingTime , getNextAvailableChannelId ,
getParentBWControllable , getPendingCall , getPendingVideoMessages ,
getState , getStreamByChannelId , getStreamById , getStreamIdForChannel ,
getStreams , getUsedStreamCount , getVideoCodecFactory , invoke , isChannelUsed ,
messageDropped , messageReceived , messageSent , newBroadcastStream ,
newPlaylistSubscriberStream , newSingleItemSubscriberStream , notify , ping ,
pingReceived , receivedBytesRead , registerDeferredResult , registerPendingCall ,
registerStream , rememberStreamBufferDuration , removeClientStream , reserveStreamId ,
retrievePendingCall , setBandwidthConfigure , setId , setMaxHandshakeTimeout ,
setMaxInactivity , setPingInterval , setSchedulingService , setState
```

```
, setup , startRoundTripMeasurement , startWaitForHandshake , toString ,
unregisterDeferredResult , unregisterStream , unreserveStreamId , updateBytesRead ,
writingMessage
```

**Methods inherited from org.red5.server.BaseConnection:** dispatchEvent ,  
getBasicScopes , getClient , getConnectParams , getDroppedMessages , getHost ,  
getPath , getReadMessages , getRemoteAddress , getRemoteAddresses , getRemotePort ,  
getScope , getSessionId , getType , getWrittenMessages , handleEvent , initialize ,  
notifyEvent , registerBasicScope , unregisterBasicScope

**Methods inherited from org.red5.server.AttributeStore:** filterNull ,  
getAttribute ,  
getAttributeNames ,  
getAttributes ,  
getBoolAttribute ,  
getByteAttribute ,  
getDoubleAttribute ,  
getIntAttribute ,  
getListAttribute ,  
getLongAttribute ,  
getMapAttribute ,  
getSetAttribute ,  
getShortAttribute ,  
getStringAttribute ,  
hasAttribute ,  
removeAttribute ,  
removeAttributes ,  
setAttribute ,  
setAttributes

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notifyAll , wait

**Fields inherited from org.red5.server.net.rtmp.RTMPMinaConnection:** ioSession ,  
log

**Fields inherited from org.red5.server.net.rtmp.RTMPConnection:** clientId  
, deferredResults , encoding , invokeId , keepAliveJobName , lastPingSent  
, lastPingTime , lastPongReceived , maxInactivity , oName , pendingCalls ,  
pingInterval , state , streamBuffers

**Fields inherited from org.red5.server.BaseConnection:** basicScopes , client ,  
closed , droppedMessages , host , params , path , readMessages , remoteAddress ,  
remoteAddresses , remotePort , scope , sessionId , type , writtenMessages

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 8. Class EdgeRTMPMinaIoHandler

```
public class EdgeRTMPMinaIoHandler extends org.?red5.?server.?net.?rtmp.?RTMPMinaIoHandler {
// Public Constructors

    public EdgeRTMPMinaIoHandler();

// Public Methods

    public void setRtmpConnManager(IRTMPConnManager rtmpConnManager);

// Protected Methods

    protected RTMPMinaConnection createRTMPMinaConnection();

}
```

**Methods inherited from org.red5.server.net.rtmp.RTMPMinaHandler:**

createRTMPMinaConnection , exceptionCaught , messageReceived , messageSent

```
, rawBufferRecieved , sessionClosed , sessionCreated , sessionOpened ,
setApplicationContext , setCodecFactory , setHandler , setMode , setRtmpConnManager
```

**Methods inherited from org.apache.mina.common.IoHandlerAdapter:** sessionIdle

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.RTMPMinaloHandler:** appCtx ,  
codecFactory , handler , log , mode , rtmpConnManager

## 9. Interface INetStreamEventHandler

Notify handler for client-side stream

### 9.1. Synopsis

```
public interface INetStreamEventHandler {
// Public Methods

    public void onStreamEvent(org.red5.server.net.rtmp.event.Notify notify);

}
```

## 10. Interface IRTMPConnManager

```
public interface IRTMPConnManager {
// Public Methods

    public RTMPConnection createConnection(Class connCls);

    public RTMPConnection getConnection(int clientId);

    public RTMPConnection removeConnection(int clientId);

    public java.util.Collection<org.red5.server.net.rtmp.RTMPConnection> removeConnections();

}
```

## 11. Interface IRTMPHandler

RTMP events handler

### 11.1. Synopsis

```
public interface IRTMPHandler {
// Public Methods

    public void connectionClosed(RTMPConnection conn,
```

```

        org.red5.server.net.rtmp.codec.RTMP state);

public void connectionOpened(RTMPConnection conn,
                            org.red5.server.net.rtmp.codec.RTMP state);

public void messageReceived(RTMPConnection conn,
                           org.red5.server.net.protocol.ProtocolState state,
                           Object message)
throws Exception;

public void messageSent(RTMPConnection conn,
                       Object message);

}

```

## 11.2. connectionClosed(RTMPConnection, RTMP)

```

public void connectionClosed(RTMPConnection conn,
                            org.red5.server.net.rtmp.codec.RTMP state);

```

Connection closed

### Parameters

conn	Connection
state	RTMP state

## 11.3. connectionOpened(RTMPConnection, RTMP)

```

public void connectionOpened(RTMPConnection conn,
                            org.red5.server.net.rtmp.codec.RTMP state);

```

Connection open event

### Parameters

conn	Connection
state	RTMP state

## 11.4. messageReceived(RTMPConnection, ProtocolState, Object)

```

public void messageReceived(RTMPConnection conn,
                           org.red5.server.net.protocol.ProtocolState state,
                           Object message)
throws Exception;

```

Message received

### Parameters

conn	Connection
state	RTMP state

message	Message
---------	---------

Exception  
Exception

## 11.5. messageSent(RTMPConnection, Object)

```
public void messageSent(RTMPConnection conn,
                        Object message);
```

Message sent

### Parameters

conn	Connection
message	Message

## 12. Class RTMPClient

RTMP client object. Initial client mode code by Christian Eckerle.

### 12.1. Synopsis

```
public class RTMPClient extends org.?red5.?server.?net.?rtmp.?BaseRTMPClientHandler {
    // Public Constructors

    public RTMPClient();

    // Public Methods

    public java.util.Map<java.lang.String, java.lang.Object> makeDefaultConnectionParams(String server,
                                                                                           int port,
                                                                                           String applica
    // Protected Methods

    protected void startConnector(String server,
                                 int port);

}
```

#### Methods inherited from org.red5.server.net.rtmp.BaseRTMPClientHandler:

connect , connectionClosed , connectionOpened , createStream , disconnect ,  
getCodecFactory , getConnManager , getSharedObject , handleException , invoke ,  
makeDefaultConnectionParams , onChunkSize , onInvoke , onPing , onSharedObject ,  
play , publish , publishStreamData , setCodecFactory , setConnectionClosedHandler ,  
setExceptionHandler , setServiceProvider , setStreamEventDispatcher , startConnector

**Methods inherited from org.red5.server.net.rtmp.BaseRTMPHandler:** getHostname  
, getStreamId , handlePendingCallResult , messageReceived , messageSent ,  
onStreamBytesRead , setApplicationContext

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.BaseRTMPClientHandler:**

codecFactory , connectArguments , connectCallback , connectionParams , connManager , serviceInvoker , serviceProvider , sharedObjects , streamEventDispatcher

**Fields inherited from org.red5.server.net.rtmp.BaseRTMPHandler:** appCtx , log

## 12.2. RTMPClient()

```
public RTMPClient();
```

Constructs a new RTMPClient.

## 13. Class RTMPClientConnManager

```
public class RTMPClientConnManager implements, org.?red5.?server.?net.?rtmp.?IRTMPConnManager {
// Protected Fields

protected RTMPConnection rtmpConn ;

// Public Constructors

public RTMPClientConnManager();

// Public Methods

public synchronized RTMPConnection createConnection(Class connCls);

public synchronized RTMPConnection getConnection();

public synchronized RTMPConnection getConnection(int clientId);

public synchronized RTMPConnection removeConnection(int clientId);

public synchronized java.util.Collection<org.red5.server.net.rtmp.RTMPConnection> removeConnection()

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 14. Class RTMPConnManager

```
public class RTMPConnManager implements, org.?red5.?server.?net.?rtmp.?IRTMPConnManager, org.?springfr...
```

// Public Constructors

```
public RTMPConnManager();
```

// Public Methods

```
public RTMPConnection createConnection(Class connCls);
```

```
public RTMPConnection createConnectionInstance(Class cls)
throws Exception;
```

```

    public RTMPConnection getConnection(int clientId);

    public RTMPConnection removeConnection(int clientId);

    public java.util.Collection<org.red5.server.net.rtmp.RTMPConnection> removeConnections();

    public void setApplicationContext(org.springframework.context.ApplicationContext appCtx)
        throws BeansException;

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 15. Class RTMPConnection

RTMP connection. Stores information about client streams, data transfer channels, pending RPC calls, bandwidth configuration, used encoding (AMF0/AMF3), connection state (is alive, last ping time and ping result) and session.

### 15.1. Synopsis

```

public abstract class RTMPConnection extends, org.red5.server.BaseConnection
    implements, org.red5.server.api.stream.IStreamCapableConnection, org.red5.server.api.serv
// Protected Fields

    protected int clientId ;

    protected java.util.HashSet<org.red5.server.net.rtmp.DeferredResult> deferredResults ;

    protected org.red5.server.api.IConnection.Encoding encoding ;

    protected java.util.concurrent.atomic.AtomicInteger invokeId ;

    protected String keepAliveJobName ;

    protected long lastPingSent ;

    protected int lastPingTime ;

    protected long lastPongReceived ;

    protected static org.slf4j.Logger log ;

    protected int maxInactivity ;

    protected javax.management.ObjectName oName ;

    protected java.util.concurrent.ConcurrentMap<java.lang.Integer, org.red5.server.api.service.IPending
    protected int pingInterval ;

    protected org.red5.server.net.rtmp.codec.RTMP state ;

```

## Package org.?red5.?server.?net.?rtmp

```
protected java.util.Map<java.lang.Integer, java.lang.Integer> streamBuffers ;  
  
// Public Constructors  
  
public RTMPConnection(String type);  
  
// Public Methods  
  
public void addClientStream(org.red5.server.api.stream.IClientStream stream);  
  
public void close();  
  
public void closeChannel(int channelId);  
  
public boolean connect(org.red5.server.api.IScope newScope,  
                      Object[] params);  
  
public org.red5.server.stream.OutputStream createOutputStream(int streamId);  
  
public void deleteStreamById(int streamId);  
  
public org.red5.server.api.IBandwidthConfigure getBandwidthConfigure();  
  
public Channel.getChannel(int channelId);  
  
public long getClientBytesRead();  
  
public org.red5.server.api.IConnection.Encoding getEncoding();  
  
public int getId();  
  
public int getInvokeId();  
  
public int getLastPingTime();  
  
public synchronized int getNextAvailableChannelId();  
  
public org.red5.server.api.IBWControllable getParentBWControllable();  
  
public long getPendingVideoMessages(int streamId);  
  
public long getReadBytes();  
  
public org.red5.server.net.rtmp.codec.RTMP getState();  
  
public org.red5.server.api.stream.IClientStream getStreamByChannelId(int channelId);  
  
public org.red5.server.api.stream.IClientStream getStreamById(int id);  
  
public int getStreamIdForChannel(int channelId);  
  
public org.red5.server.stream.VideoCodecFactory getVideoCodecFactory();  
  
public long getWrittenBytes();  
  
public void invoke(String method);
```

## Package org.?red5.?server.?net.?rtmp

```
public void invoke(String method,
                   Object[] params);

public void invoke(String method,
                   Object[] params,
                   org.red5.server.api.service.IPendingServiceCallback callback);

public void invoke(String method,
                   org.red5.server.api.service.IPendingServiceCallback callback);

public void invoke(org.red5.server.api.service.IServiceCall call);

public void invoke(org.red5.server.api.service.IServiceCall call,
                  int channel);

public boolean isChannelUsed(int channelId);

public org.red5.server.api.stream.IClientBroadcastStream newBroadcastStream(int streamId);

public org.red5.server.api.stream.IPlaylistSubscriberStream newPlaylistSubscriberStream(int streamId);

public org.red5.server.api.stream.ISingleItemSubscriberStream newSingleItemSubscriberStream(int streamId);

public void notify(String method);

public void notify(String method,
                  Object[] params);

public void notify(org.red5.server.api.service.IServiceCall call);

public void notify(org.red5.server.api.service.IServiceCall call,
                  int channel);

public void ping();

public void ping(org.red5.server.net.rtmp.event.Ping ping);

public abstract void rawWrite(org.apache.mina.common.ByteBuffer out);

public void receivedBytesRead(int bytes);

public void registerPendingCall(int invokeId,
                               org.red5.server.api.service.IPendingServiceCall call);

public void removeClientStream(int streamId);

public int reserveStreamId();

public void setBandwidthConfigure(org.red5.server.api.IBandwidthConfigure config);

public void setId(int clientId);

public void setMaxHandshakeTimeout(int maxHandshakeTimeout);

public void setMaxInactivity(int maxInactivity);
```

## Package org.?red5.?server.?net.?rtmp

```
public void setPingInterval(int pingInterval);

public void setSchedulingService(org.red5.server.api.scheduling.ISchedulingService schedulingService);

public void setState(org.red5.server.net.rtmp.codec.RTMP state);

public void setup(String host,
                  String path,
                  String sessionId,
                  java.util.Map<java.lang.String, java.lang.Object> params);

public void startRoundTripMeasurement();

public String toString();

public void unreserveStreamId(int streamId);

public abstract void write(org.red5.server.net.rtmp.message.Packet out);

// Protected Methods

protected String createStreamName();

protected org.red5.server.api.service.IPendingServiceCall getPendingCall(int invokeId);

protected java.util.Collection<org.red5.server.api.stream.IClientStream> getStreams();

protected int getUsedStreamCount();

protected void messageDropped();

protected void messageReceived();

protected void messageSent(org.red5.server.net.rtmp.message.Packet message);

protected abstract void onInactive();

protected void pingReceived(org.red5.server.net.rtmp.event.Ping pong);

protected void registerDeferredResult(DeferredResult result);

protected void registerStream(org.red5.server.api.stream.IClientStream stream);

protected void rememberStreamBufferDuration(int streamId,
                                            int bufferDuration);

protected org.red5.server.api.service.IPendingServiceCall retrievePendingCall(int invokeId);

protected void startWaitForHandshake(org.red5.server.api.scheduling.ISchedulingService service);

protected void unregisterDeferredResult(DeferredResult result);

protected void unregisterStream(org.red5.server.api.stream.IClientStream stream);

protected void updateBytesRead();
```

```

protected void writingMessage(org.red5.server.net.rtmp.message.Packet message);

}

```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?RTMPMinaConnection , org.?red5.?server.?net.?rtmp.?RTMPOriginConnection , org.?red5.?server.?net.?rtmpt.?BaseRTMPTConnection

**Methods inherited from org.red5.server.BaseConnection:** close , connect , dispatchEvent , getBasicScopes , getClient , getClientBytesRead , getConnectParams , getDroppedMessages , getHost , getPath , getPendingMessages , getPendingVideoMessages , getReadBytes , getReadMessages , getRemoteAddress , getRemoteAddresses , getRemotePort , getScope , getSessionId , getType , getWrittenBytes , getWrittenMessages , handleEvent , initialize , isConnected , notifyEvent , registerBasicScope , unregisterBasicScope

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttribute , getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute , getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute , getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute , hasAttribute , removeAttribute , removeAttributes , setAttribute , setAttributes

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.BaseConnection:** basicScopes , client , closed , droppedMessages , host , log , params , path , readMessages , remoteAddress , remoteAddresses , remotePort , scope , sessionId , type , writtenMessages

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 15.2. RTMPConnection(String)

```
public RTMPConnection(String type);
```

Creates anonymous RTMP connection without scope.

Parameters	
type	Connection type

## 15.3. deferredResults

```
protected java.util.HashSet<org.red5.server.net.rtmp.DeferredResult> deferredResults ;
```

**See Also**

[org.red5.server.net.rtmp.DeferredResult](#)

Deferred results set.

## 15.4. encoding

```
protected org.red5.server.api.IConnection.Encoding encoding ;
```

AMF version, AMF0 by default.

## 15.5. invokeId

```
protected java.util.concurrent.atomic.AtomicInteger invokeId ;
```

Identifier for remote calls.

## 15.6. keepAliveJobName

```
protected String keepAliveJobName ;
```

Name of quartz job that keeps connection alive.

## 15.7. lastPingSent

```
protected long lastPingSent ;
```

Timestamp when last ping command was sent.

## 15.8. lastPingTime

```
protected int lastPingTime ;
```

Last ping timestamp.

## 15.9. lastPongReceived

```
protected long lastPongReceived ;
```

Timestamp when last ping result was received.

## 15.10. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 15.11. maxInactivity

```
protected int maxInactivity ;
```

Maximum time in ms after a client is disconnected because of inactivity.

## 15.12. oName

```
protected javax.management.ObjectName oName ;
```

MBean object name used for de/registration purposes.

## 15.13. pendingCalls

```
protected java.util.concurrent.ConcurrentMap<java.lang.Integer, org.red5.server.api.service.IPendingCall>;
```

Hash map that stores pending calls and ids as pairs.

## 15.14. pingInterval

```
protected int pingInterval ;
```

Ping interval in ms to detect dead clients.

## 15.15. streamBuffers

```
protected java.util.Map<java.lang.Integer, java.lang.Integer> streamBuffers ;
```

Remembered stream buffer durations.

## 15.16. close()

```
public void close();
```

## 15.17. closeChannel(int)

```
public void closeChannel(int channelId);
```

Closes channel.

### Parameters

channelId	Channel id
-----------	------------

## 15.18. createOutputStream(int)

```
public org.red5.server.stream.OutputStream createOutputStream(int streamId);
```

Creates output stream object from stream id. Output stream consists of audio, data and video channels.

### Parameters

streamId	Stream id
----------	-----------

return	Output stream object
--------	----------------------

### See Also

[org.red5.server.stream.OutputStream](#)

## 15.19. createStreamName()

```
protected String createStreamName();
```

Generates new stream name.

### Parameters

<i>return</i>	New stream name
---------------	-----------------

## 15.20. deleteStreamById(int)

```
public void deleteStreamById(int streamId);
```

**Specified by:** Method deleteStreamById in interface IStreamCapableConnection

Deletes the stream with the given id.

## 15.21. getBandwidthConfigure()

```
public org.red5.server.api.IBandwidthConfigure getBandwidthConfigure();
```

## 15.22. getChannel(int)

```
public Channel getChannel(int channelId);
```

Return channel by id.

### Parameters

channelId	Channel id
<i>return</i>	Channel by id

## 15.23. getClientBytesRead()

```
public long getClientBytesRead();
```

Get number of bytes the client reported to have received.

### Parameters

<i>return</i>	Number of bytes
---------------	-----------------

## 15.24. getEncoding()

```
public org.red5.server.api.IConnection.Encoding getEncoding();
```

Return AMF protocol encoding used by this connection.

### Parameters

<i>return</i>	AMF encoding used by connection
---------------	---------------------------------

## 15.25. getInvokeId()

```
public int getInvokeId();
```

Generate next invoke id.

### Parameters

<i>return</i>	Next invoke id for RPC
---------------	------------------------

## 15.26. getLastPingTime()

```
public int getLastPingTime();
```

## 15.27. getNextAvailableChannelId()

```
public synchronized int getNextAvailableChannelId();
```

Getter for next available channel id.

### Parameters

<i>return</i>	Next available channel id
---------------	---------------------------

## 15.28. getParentBWControllable()

```
public org.red5.server.api.IBWControllable getParentBWControllable();
```

## 15.29. getPendingCall(int)

```
protected org.red5.server.api.service.IPendingServiceCall getPendingCall(int invokeId);
```

Get pending call service by id.

### Parameters

invokeld	Pending call service id
<i>return</i>	Pending call service object

## 15.30. getPendingVideoMessages(int)

```
public long getPendingVideoMessages(int streamId);
```

**Specified by:** Method `getPendingVideoMessages` in interface `IStreamCapableConnection`

Total number of video messages that are pending to be sent to a stream.

## 15.31. getReadBytes()

```
public long getReadBytes();
```

## 15.32. getStreamByChannelId(int)

```
public org.red5.server.api.stream.IClientStream getStreamByChannelId(int channelId);
```

Return stream by given channel id.

### Parameters

channelId	Channel id
<i>return</i>	Stream that channel belongs to

### 15.33. getStreamById(int)

```
public org.red5.server.api.stream.IClientStream getStreamById(int id);
```

**Specified by:** Method `getStreamById` in interface `IStreamCapableConnection`

Get a stream by its id.

### 15.34. getStreamIdForChannel(int)

```
public int getStreamIdForChannel(int channelId);
```

Return stream id for given channel id.

Parameters	
channelId	Channel id
<i>return</i>	ID of stream that channel belongs to

### 15.35. getStreams()

```
protected java.util.Collection<org.red5.server.api.stream.IClientStream> getStreams();
```

Getter for client streams.

Parameters	
<i>return</i>	Client streams as array

### 15.36. getUsedStreamCount()

```
protected int getUsedStreamCount();
```

Getter for used stream count.

Parameters	
<i>return</i>	Value for property 'usedStreamCount'.

### 15.37. getVideoCodecFactory()

```
public org.red5.server.stream.VideoCodecFactory getVideoCodecFactory();
```

Getter for video codec factory.

Parameters	
<i>return</i>	Video codec factory

## 15.38. getWrittenBytes()

```
public long getWrittenBytes();
```

## 15.39. invoke(IServiceCall)

```
public void invoke(org.red5.server.api.service(IServiceCall call);
```

**Specified by:** Method invoke in interface IServiceCapableConnection

Invokes service using remoting call object

## 15.40. invoke(IServiceCall, int)

```
public void invoke(org.red5.server.api.service(IServiceCall call,
                int channel);
```

**Specified by:** Method invoke in interface IServiceCapableConnection

Invoke service using call and channel

## 15.41. invoke(String)

```
public void invoke(String method);
```

**Specified by:** Method invoke in interface IServiceCapableConnection

Invoke method by name

## 15.42. invoke(String, IPendingServiceCallback)

```
public void invoke(String method,
                  org.red5.server.api.service.IPendingServiceCallback callback);
```

**Specified by:** Method invoke in interface IServiceCapableConnection

Invoke method by name with callback

## 15.43. invoke(String, Object[])

```
public void invoke(String method,
                  Object[] params);
```

**Specified by:** Method invoke in interface IServiceCapableConnection

Invoke method with parameters

## 15.44. invoke(String, Object[], IPendingServiceCallback)

```
public void invoke(String method,
                  Object[] params,
                  org.red5.server.api.service.IPendingServiceCallback callback);
```

**Specified by:** Method invoke in interface IServiceCapableConnection

## 15.45. isChannelUsed(int)

```
public boolean isChannelUsed(int channelId);
```

Checks whether channel is used.

### Parameters

channelId	Channel id
<i>return</i>	<code>true</code> if channel is in use, <code>false</code> otherwise

## 15.46. messageDropped()

```
protected void messageDropped();
```

Increases number of dropped messages.

## 15.47. messageReceived()

```
protected void messageReceived();
```

Increases number of read messages by one. Updates number of bytes read.

## 15.48. messageSent(Packet)

```
protected void messageSent(org.red5.server.net.rtmp.message.Packet message);
```

Mark message as sent.

### Parameters

message	Message to mark
---------	-----------------

## 15.49. newBroadcastStream(int)

```
public org.red5.server.api.stream.IClientBroadcastStream newBroadcastStream(int streamId);
```

**Specified by:** Method newBroadcastStream in interface IStreamCapableConnection

Create a broadcast stream.

## 15.50. newPlaylistSubscriberStream(int)

```
public org.red5.server.api.stream.IPlaylistSubscriberStream newPlaylistSubscriberStream(int streamId);
```

**Specified by:** Method newPlaylistSubscriberStream in interface IStreamCapableConnection

Create a stream that can play a list.

## 15.51. newSingleItemSubscriberStream(int)

```
public org.red5.server.api.stream.ISingleItemSubscriberStream newSingleItemSubscriberStream(int streamId);
```

**Specified by:** Method newSingleItemSubscriberStream in interface IStreamCapableConnection

Create a stream that can play only one item. To be implemented.

## 15.52. notify(IServiceCall)

```
public void notify(org.red5.server.api.service(IServiceCall call);
```

**Specified by:** Method notify in interface IServiceCapableConnection

## 15.53. notify(IServiceCall, int)

```
public void notify(org.red5.server.api.service(IServiceCall call,
                  int channel);
```

**Specified by:** Method notify in interface IServiceCapableConnection

## 15.54. notify(String)

```
public void notify(String method);
```

**Specified by:** Method notify in interface IServiceCapableConnection

## 15.55. notify(String, Object[])

```
public void notify(String method,
                  Object[] params);
```

**Specified by:** Method notify in interface IServiceCapableConnection

## 15.56. onInactive()

```
protected abstract void onInactive();
```

Inactive state event handler.

## 15.57. ping()

```
public void ping();
```

## 15.58. ping(Ping)

```
public void ping(org.red5.server.net.rtmp.event.Ping ping);
```

Handler for ping event.

### Parameters

ping	Ping event context
------	--------------------

## 15.59. pingReceived(Ping)

```
protected void pingReceived(org.red5.server.net.rtmp.event.Ping pong);
```

Marks that pingback was received.

#### Parameters

pong	Ping object
------	-------------

## 15.60. rawWrite(ByteBuffer)

```
public abstract void rawWrite(org.apache.mina.common.ByteBuffer out);
```

Write raw byte buffer.

#### Parameters

out	Byte buffer
-----	-------------

## 15.61. receivedBytesRead(int)

```
public void receivedBytesRead(int bytes);
```

Read number of received bytes.

#### Parameters

bytes	Number of bytes
-------	-----------------

## 15.62. registerDeferredResult(DeferredResult)

```
protected void registerDeferredResult(DeferredResult result);
```

Registers deferred result.

#### Parameters

result	Result to register
--------	--------------------

## 15.63. registerPendingCall(int, IPendingServiceCall)

```
public void registerPendingCall(int invokeId,
                               org.red5.server.api.service.IPendingServiceCall call);
```

Register pending call (remote function call that is yet to finish).

#### Parameters

invokeld	Deferred operation id
call	Call service

## 15.64. registerStream(IClientStream)

```
protected void registerStream(org.red5.server.api.stream.IClientStream stream);
```

Store a stream in the connection.

#### Parameters

stream
--------

### 15.65. reserveStreamId()

```
public int reserveStreamId();
```

**Specified by:** Method reserveStreamId in interface IStreamCapableConnection

Return a reserved stream id for use. According to FCS/FMS regulation, the base is 1.

### 15.66. retrievePendingCall(int)

```
protected org.red5.server.api.service.IPendingServiceCall retrievePendingCall(int invokeId);
```

Retrieve pending call service by id. The call will be removed afterwards.

#### Parameters

invokeld	Pending call service id
<i>return</i>	Pending call service object

### 15.67. setBandwidthConfigure(IBandwidthConfigure)

```
public void setBandwidthConfigure(org.red5.server.api.IBandwidthConfigure config);
```

### 15.68. setMaxHandshakeTimeout(int)

```
public void setMaxHandshakeTimeout(int maxHandshakeTimeout);
```

Set maximum time to wait for valid handshake in milliseconds.

#### Parameters

maxHandshakeTimeout	Maximum time in milliseconds
---------------------	------------------------------

### 15.69. setMaxInactivity(int)

```
public void setMaxInactivity(int maxInactivity);
```

Setter for maximum inactivity.

#### Parameters

maxInactivity	Maximum time in ms after which a client is disconnected in case of inactivity.
---------------	--

### 15.70. setPingInterval(int)

```
public void setPingInterval(int pingInterval);
```

Setter for ping interval.

#### Parameters

pingInterval	Interval in ms to ping clients. Set to 0 to disable ghost detection code.
--------------	---

## 15.71. setSchedulingService(ISchedulingService)

```
public void setSchedulingService(org.red5.server.api.scheduling.ISchedulingService schedulingService);
```

Sets the scheduling service.

#### Parameters

schedulingService
-------------------

## 15.72. setup(String, String, String, Map<String, Object>)

```
public void setup(String host,
                  String path,
                  String sessionId,
                  java.util.Map<java.lang.String, java.lang.Object> params);
```

Initialize connection.

#### Parameters

host	Connection host
path	Connection path
sessionId	Connection session id
params	Params passed from client

## 15.73. startRoundTripMeasurement()

```
public void startRoundTripMeasurement();
```

Starts measurement.

## 15.74. startWaitForHandshake(ISchedulingService)

```
protected void startWaitForHandshake(org.red5.server.api.scheduling.ISchedulingService service);
```

Start waiting for a valid handshake.

#### Parameters

service	The scheduling service to use
---------	-------------------------------

## 15.75. toString()

```
public String toString();
```

## 15.76. unregisterDeferredResult(DeferredResult)

```
protected void unregisterDeferredResult(DeferredResult result);
```

Unregister deferred result

### Parameters

result	Result to unregister
--------	----------------------

## 15.77. unregisterStream(IClientStream)

```
protected void unregisterStream(org.red5.server.api.stream.IClientStream stream);
```

Remove a stream from the connection.

### Parameters

stream
--------

## 15.78. unreserveStreamId(int)

```
public void unreserveStreamId(int streamId);
```

**Specified by:** Method unreserveStreamId in interface IStreamCapableConnection

Unreserve this id for future use.

## 15.79. updateBytesRead()

```
protected void updateBytesRead();
```

Update number of bytes to read next value.

## 15.80. write(Packet)

```
public abstract void write(org.red5.server.net.rtmp.message.Packet out);
```

Write packet.

### Parameters

out	Packet
-----	--------

## 15.81. writingMessage(Packet)

```
protected void writingMessage(org.red5.server.net.rtmp.message.Packet message);
```

Mark message as being written.

### Parameters

message

Message to mark

## 16. Class RTMPHandler

RTMP events handler.

### 16.1. Synopsis

```

public class RTMPHandler extends, org.?red5.?server.?net.?rtmp.?BaseRTMPHandler {
// Protected Fields

    protected static org.slf4j.Logger log ;

    protected org.red5.server.api.IServer server ;

    protected org.red5.server.net.rtmp.status.StatusObjectService statusObjectService ;

// Public Constructors

    public RTMPHandler();

// Public Methods

    public org.red5.server.net.rtmp.status.StatusObject getStatus(String code);

    public void setServer(org.red5.server.api.IServer server);

    public void setStatusObjectService(org.red5.server.net.rtmp.status.StatusObjectService statusObjectService);

// Protected Methods

    protected void invokeCall(RTMPConnection conn,
                            org.red5.server.api.service.IServiceCall call);

    protected void onChunkSize(RTMPConnection conn,
                            Channel channel,
                            org.red5.server.net.rtmp.message.Header source,
                            org.red5.server.net.rtmp.event.ChunkSize chunkSize);

    protected void onInvoke(RTMPConnection conn,
                            Channel channel,
                            org.red5.server.net.rtmp.message.Header source,
                            org.red5.server.net.rtmp.event.Notify invoke,
                            org.red5.server.net.rtmp.codec.RTMP rtmp);

    protected void onPing(RTMPConnection conn,
                            Channel channel,
                            org.red5.server.net.rtmp.message.Header source,
                            org.red5.server.net.rtmp.event.Ping ping);

    protected void onSharedObject(RTMPConnection conn,
                                Channel channel,
                                org.red5.server.net.rtmp.message.Header source,
                                org.red5.server.so.SharedObjectMessage object);

}

```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?EdgeRTMPHandler , org.?red5.?server.?net.?rtmpt.?RTMPTHandler

**Methods inherited from org.red5.server.net.rtmp.BaseRTMPHandler:**

connectionClosed , connectionOpened , getHostname , getStreamId ,  
 handlePendingCallResult , messageReceived , messageSent , onChunkSize , onInvoke ,  
 onPing , onSharedObject , onStreamBytesRead , setApplicationContext

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
 , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.BaseRTMPHandler:** appCtx , log

## 16.2. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 16.3. server

```
protected org.red5.server.api.IServer server ;
```

Red5 server instance.

## 16.4. statusObjectService

```
protected org.red5.server.net.rtmp.status.StatusObjectService statusObjectService ;
```

Status object service.

## 16.5. invokeCall(RTMPConnection, IServiceCall)

```
protected void invokeCall(RTMPConnection conn,  

                         org.red5.server.api.service(IServiceCall call);
```

Remoting call invocation handler.

### Parameters

conn	RTMP connection
call	Service call

## 16.6. onChunkSize(RTMPConnection, Channel, Header, ChunkSize)

```
protected void onChunkSize(RTMPConnection conn,  

                          Channel channel,  

                          org.red5.server.net.rtmp.message.Header source,  

                          org.red5.server.net.rtmp.event.ChunkSize chunkSize);
```

## 16.7. onInvoke(RTMPConnection, Channel, Header, Notify, RTMP)

```
protected void onInvoke(RTMPConnection conn,  

                      Channel channel,
```

```
org.red5.server.net.rtmp.message.Header source,
org.red5.server.net.rtmp.event.Notify invoke,
org.red5.server.net.rtmp.codec.RTMP rtmp);
```

## 16.8. onPing(RTMPConnection, Channel, Header, Ping)

```
protected void onPing(RTMPConnection conn,
                      Channel channel,
                      org.red5.server.net.rtmp.message.Header source,
                      org.red5.server.net.rtmp.event.Ping ping);
```

## 16.9. onSharedObject(RTMPConnection, Channel, Header, SharedObjectMessage)

```
protected void onSharedObject(RTMPConnection conn,
                             Channel channel,
                             org.red5.server.net.rtmp.message.Header source,
                             org.red5.server.so.SharedObjectMessage object);
```

## 16.10. setServer(I Server)

```
public void setServer(org.red5.server.api.I Server server);
```

Setter for server object.

### Parameters

server	Red5 server instance
--------	----------------------

## 16.11. setStatusObjectService(StatusObjectService)

```
public void setStatusObjectService(org.red5.server.net.rtmp.status.StatusObjectService statusObjectS
```

Setter for status object service.

### Parameters

statusObjectService	Status object service.
---------------------	------------------------

## 17. Class RTMPHandshake

Generates the second 1536 byte chunk in the RTMP handshake response for compatibility with Flash 9,0,124,0. Clients that require this send a nonzero value as the fifth byte of the handshake request.

This class is based on the Ruby handshaking code from Takuma Mori.

### 17.1. Synopsis

```
public class RTMPHandshake {
// Public Static Fields

    public static final byte[] HANDSHAKE_PAD_BYTES ;
```

```
// Protected Fields

protected static org.slf4j.Logger log ;

protected java.util.Random random ;

// Public Constructors

public RTMPHandshake();

// Public Static Methods

public static byte[] getHandshakeBytes();

// Public Methods

public byte[] calculateHMAC_SHA256(byte[] input,
                                     byte[] key);

public org.apache.mina.common.ByteBuffer generateResponse(org.apache.mina.common.ByteBuffer input)

// Protected Methods

protected byte[] getNewKeyPart(org.apache.mina.common.ByteBuffer input);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 18. Class RTMPMinaConnection

```
public class RTMPMinaConnection extends org.?red5.?server.?net.?rtmp.?RTMPConnection
    implements org.?red5.?server.?net.?rtmp.?RTMPMinaConnectionMBean {
// Protected Fields

protected org.apache.mina.common.IoSession ioSession ;

protected static org.slf4j.Logger log ;

// Public Constructors

public RTMPMinaConnection();

// Public Methods

public void close();

public boolean connect(org.red5.server.api.IScope newScope,
                      Object[] params);

public org.apache.mina.common.IoSession getIoSession();

public long getPendingMessages();

public long getReadBytes();

public long getWrittenBytes();
```

```

    public void invokeMethod(String method);

    public boolean isConnected();

    public void rawWrite(org.apache.mina.common.ByteBuffer out);

    public void setIoSession(org.apache.mina.common.IoSession protocolSession);

    public void write(org.red5.server.net.rtmp.message.Packet out);

// Protected Methods

    protected void onInactive();

}

```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?EdgeRTMPMinaConnection

**Methods inherited from org.red5.server.net.rtmp.RTMPConnection:** addClientStream, close, closeChannel, connect, createOutputStream, createStreamName, deleteStreamById, getBandwidthConfigure, getChannel, getClientBytesRead, getEncoding, getId, getInvokeId, getLastPingTime, getNextAvailableChannelId, getParentBWControllable, getPendingCall, getPendingVideoMessages, getReadBytes, getState, getStreamByChannelId, getStreamById, getStreamIdForChannel, getStreams, getUsedStreamCount, getVideoCodecFactory, getWrittenBytes, invoke, isChannelUsed, messageDropped, messageReceived, messageSent, newBroadcastStream, newPlaylistSubscriberStream, newSingleItemSubscriberStream, notify, onInactive, ping, pingReceived, rawWrite, receivedBytesRead, registerDeferredResult, registerPendingCall, registerStream, rememberStreamBufferDuration, removeClientStream, reserveStreamId, retrievePendingCall, setBandwidthConfigure, setId, setMaxHandshakeTimeout, setMaxInactivity, setPingInterval, setSchedulingService, setState, setup, startRoundTripMeasurement, startWaitForHandshake, toString, unregisterDeferredResult, unregisterStream, unreserveStreamId, updateBytesRead, write, writingMessage

**Methods inherited from org.red5.server.BaseConnection:** dispatchEvent, getBasicScopes, getClient, getConnectParams, getDroppedMessages, getHost, getPath, getPendingMessages, getReadMessages, getRemoteAddress, getRemoteAddresses, getRemotePort, getScope, getSessionId, getType, getWrittenMessages, handleEvent, initialize, isConnected, notifyEvent, registerBasicScope, unregisterBasicScope

**Methods inherited from org.red5.server.AttributeStore:** filterNull, getAttribute, getAttributeNames, getAttributes, getBoolAttribute, getByteAttribute, getDoubleAttribute, getIntAttribute, getListAttribute, getLongAttribute, getMapAttribute, getSetAttribute, getShortAttribute, getStringAttribute, hasAttribute, removeAttribute, removeAttributes, setAttribute, setAttributes

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notifyAll, wait

**Fields inherited from org.red5.server.net.rtmp.RTMPConnection:** clientId, deferredResults, encoding, invokeId, keepAliveJobName, lastPingSent,

```
lastPingTime , lastPongReceived , log , maxInactivity , oName , pendingCalls ,
pingInterval , state , streamBuffers
```

**Fields inherited from org.red5.server.BaseConnection:** basicScopes , client , closed , droppedMessages , host , params , path , readMessages , remoteAddress , remoteAddresses , remotePort , scope , sessionId , type , writtenMessages

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 18.1. RTMPMinaConnection()

```
public RTMPMinaConnection();
```

Constructs a new RTMPMinaConnection.

## 18.2. ioSession

```
protected org.apache.mina.common.IoSession ioSession ;
```

MINA I/O session, connection between two endpoints

## 18.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 18.4. close()

```
public void close();
```

**Specified by:** Method close in interface RTMPMinaConnectionMBean

## 18.5. getIoSession()

```
public org.apache.mina.common.IoSession getIoSession();
```

Return MINA I/O session.

### Parameters

return	MINA O/I session, connection between two endpoints
--------	--

## 18.6. getPendingMessages()

```
public long getPendingMessages();
```

**Specified by:** Method getPendingMessages in interface RTMPMinaConnectionMBean

## 18.7. getReadBytes()

```
public long getReadBytes();
```

**Specified by:** Method getReadBytes in interface RTMPMinaConnectionMBean

## 18.8. getWrittenBytes()

```
public long getWrittenBytes();
```

**Specified by:** Method getWrittenBytes in interface RTMPMinaConnectionMBean

## 18.9. isConnected()

```
public boolean isConnected();
```

**Specified by:** Method isConnected in interface RTMPMinaConnectionMBean

## 18.10. onInactive()

```
protected void onInactive();
```

## 18.11. rawWrite(ByteBuffer)

```
public void rawWrite(org.apache.mina.common.ByteBuffer out);
```

## 18.12. setIoSession(IoSession)

```
public void setIoSession(org.apache.mina.common.IoSession protocolSession);
```

Setter for MINA I/O session (connection).

### Parameters

protocolSession	Protocol session
-----------------	------------------

## 18.13. write(Packet)

```
public void write(org.red5.server.net.rtmp.message.Packet out);
```

# 19. Interface RTMPMinaConnectionMBean

Base abstract class for connections. Adds connection specific functionality like work with clients to AttributeStore.

## 19.1. Synopsis

```
public interface RTMPMinaConnectionMBean {
// Public Methods

    public void close();

    public java.util.Map<java.lang.String, java.lang.Object> getConnectParams();

    public long getDroppedMessages();

    public String getHost();
```

```

public String getPath();

public long getPendingMessages();

public long getPendingVideoMessages(int streamId);

public long getReadBytes();

public long getReadMessages();

public String getRemoteAddress();

public java.util.List<java.lang.String> getRemoteAddresses();

public int getRemotePort();

public String getSessionId();

public String getType();

public long getWrittenBytes();

public long getWrittenMessages();

public void invokeMethod(String method);

public boolean isConnected();

}

```

## 20. Class RTMPMinaloHandler

Handles all RTMP protocol events fired by the MINA framework.

### 20.1. Synopsis

```

public class RTMPMinaloHandler extends org.apache.mina.common.IoHandlerAdapter
    implements org.springframework.context.ApplicationContextAware {
// Protected Fields

    protected org.springframework.context.ApplicationContext appCtx;

    protected org.apache.mina.filter.codec.ProtocolCodecFactory codecFactory;

    protected IRTMPHandler handler;

    protected static org.slf4j.Logger log;

    protected boolean mode;

    protected IRTMPConnManager rtmpConnManager;

// Public Constructors

```

```

public RTMPMinaIoHandler();

// Public Methods

public void exceptionCaught(org.apache.mina.common.IoSession session,
                           Throwable cause)
                           throws Exception;

public void messageReceived(org.apache.mina.common.IoSession session,
                           Object in)
                           throws Exception;

public void messageSent(org.apache.mina.common.IoSession session,
                       Object message)
                       throws Exception;

public void sessionClosed(org.apache.mina.common.IoSession session)
                         throws Exception;

public void sessionCreated(org.apache.mina.common.IoSession session)
                          throws Exception;

public void sessionOpened(org.apache.mina.common.IoSession session)
                         throws Exception;

public void setApplicationContext(org.springframework.context.ApplicationContext appCtx)
                                throws BeansException;

public void setCodecFactory(org.apache.mina.filter.codec.ProtocolCodecFactory codecFactory);

public void setHandler(IRTMPHandler handler);

public void setMode(boolean mode);

public void setRtmpConnManager(IRTMPConnManager rtmpConnManager);

// Protected Methods

protected RTMPMinaConnection createRTMPMinaConnection();

protected void rawBufferReceived(org.red5.server.net.protocol.ProtocolState state,
                               org.apache.mina.common.ByteBuffer in,
                               org.apache.mina.common.IoSession session);

}

```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?EdgeRTMPMinaIoHandler

**Methods inherited from org.apache.mina.common.IoHandlerAdapter:**

exceptionCaught , messageReceived , messageSent , sessionClosed , sessionCreated ,  
sessionIdle , sessionOpened

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 20.2. appCtx

```
protected org.springframework.context.ApplicationContext appCtx ;
```

Application context

### 20.3. codecFactory

```
protected org.apache.mina.filter.codec.ProtocolCodecFactory codecFactory ;
```

RTMP protocol codec factory

### 20.4. handler

```
protected IRTMPHandler handler ;
```

RTMP events handler

### 20.5. log

```
protected static org.slf4j.Logger log ;
```

Logger

### 20.6. mode

```
protected boolean mode ;
```

Mode

### 20.7. exceptionCaught(IoSession, Throwable)

```
public void exceptionCaught(org.apache.mina.common.IoSession session,
                           Throwable cause)
                           throws Exception;
```

### 20.8. messageReceived(IoSession, Object)

```
public void messageReceived(org.apache.mina.common.IoSession session,
                           Object in)
                           throws Exception;
```

### 20.9. messageSent(IoSession, Object)

```
public void messageSent(org.apache.mina.common.IoSession session,
                       Object message)
                       throws Exception;
```

### 20.10. rawBufferRecieved(ProtocolState, ByteBuffer, IoSession)

```
protected void rawBufferRecieved(org.red5.server.net.protocol.ProtocolState state,
                                 org.apache.mina.common.ByteBuffer in,
                                 org.apache.mina.common.IoSession session);
```

Handle raw buffer receiving event.

**Parameters**

state	Protocol state
in	Data buffer
session	I/O session, that is, connection between two endpoints

**20.11. sessionClosed(IoSession)**

```
public void sessionClosed(org.apache.mina.common.IoSession session)
    throws Exception;
```

**20.12. sessionCreated(IoSession)**

```
public void sessionCreated(org.apache.mina.common.IoSession session)
    throws Exception;
```

**20.13. sessionOpened(IoSession)**

```
public void sessionOpened(org.apache.mina.common.IoSession session)
    throws Exception;
```

**20.14. setApplicationContext(ApplicationContext)**

```
public void setApplicationContext(org.springframework.context.ApplicationContext appCtx)
    throws BeansException;
```

**Specified by:** Method `setApplicationContext` in interface `ApplicationContextAware`

**20.15. setCodecFactory(ProtocolCodecFactory)**

```
public void setCodecFactory(org.apache.mina.filter.codec.ProtocolCodecFactory codecFactory);
```

Setter for codec factory.

**Parameters**

codecFactory	RTMP protocol codec factory
--------------	-----------------------------

**20.16. setHandler(IRTMPHandler)**

```
public void setHandler(IRTMPHandler handler);
```

Setter for handler.

**Parameters**

handler	RTMP events handler
---------	---------------------

**20.17. setMode(boolean)**

```
public void setMode(boolean mode);
```

Setter for mode.

#### Parameters

mode	true if handler should work in server mode, false otherwise
------	---

## 21. Class RTMPMinaTransport

Transport setup class configures socket acceptor and thread pools for RTMP in Mina. Note: This code originates from AsyncWeb, I've modified it for use with Red5. - Luke

### 21.1. Synopsis

```
public class RTMPMinaTransport implements, org.?red5.?server.?net.?rtmp.?RTMPMinaTransportMBean {
// Protected Fields

    protected org.apache.mina.transport.socket.nio.SocketAcceptor acceptor ;

    protected String address ;

    protected java.util.concurrent.ExecutorService eventExecutor ;

    protected int eventThreadsCore ;

    protected int eventThreadsKeepalive ;

    protected int eventThreadsMax ;

    protected int eventThreadsQueue ;

    protected org.apache.mina.common.IoHandlerAdapter ioHandler ;

    protected int ioThreads ;

    protected int jmxPollInterval ;

    protected javax.management.ObjectName oName ;

    protected int port ;

    protected int receiveBufferSize ;

    protected int sendBufferSize ;

    protected org.apache.mina.integration.jmx.IoServiceManager serviceManager ;

    protected javax.management.ObjectName serviceManagerObjectName ;

    protected boolean tcpNoDelay ;

    protected boolean useHeapBuffers ;

// Public Constructors

    public RTMPMinaTransport();
```

```
// Public Methods

    public int getJmxPollInterval();

    public void setAddress(String address);

    public void setEventThreadsCore(int eventThreadsCore);

    public void setEventThreadsKeepalive(int eventThreadsKeepalive);

    public void setEventThreadsMax(int eventThreadsMax);

    public void setEventThreadsQueue(int eventThreadsQueue);

    public void setIoHandler(org.apache.mina.common.IoHandlerAdapter rtmpIOHandler);

    public void setIoThreads(int ioThreads);

    public void setJmxPollInterval(int jmxPollInterval);

    public void setPort(int port);

    public void setReceiveBufferSize(int receiveBufferSize);

    public void setSendBufferSize(int sendBufferSize);

    public void setTcpNoDelay(boolean tcpNoDelay);

    public void setUseHeapBuffers(boolean useHeapBuffers);

    public void start()
        throws Exception;

    public void stop();

    public String toString();

// Protected Methods

    protected java.util.concurrent.BlockingQueue<java.lang.Runnable> threadQueue(int size);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 21.2. oName

```
protected javax.management.ObjectName oName ;
```

MBean object name used for de/registration purposes.

## 22. Interface RTMPMinaTransportMBean

```

public interface RTMPMinaTransportMBean {
// Public Methods

    public void setAddress(String address);

    public void setEventThreadsCore(int eventThreadsCore);

    public void setEventThreadsKeepalive(int eventThreadsKeepalive);

    public void setEventThreadsMax(int eventThreadsMax);

    public void setEventThreadsQueue(int eventThreadsQueue);

    public void setIoHandler(org.apache.mina.common.IoHandlerAdapter rtmpIOHandler);

    public void setIoThreads(int ioThreads);

    public void setPort(int port);

    public void setReceiveBufferSize(int receiveBufferSize);

    public void setSendBufferSize(int sendBufferSize);

    public void setTcpNoDelay(boolean tcpNoDelay);

    public void setUseHeapBuffers(boolean useHeapBuffers);

    public void start()
        throws Exception;

    public void stop();

}

```

## 23. Class RTMPOriginConnection

A pseudo-connection on Origin that represents a client on Edge. The connection is created behind a MRTMP connection so no handshake job or keep-alive job is necessary. No raw byte data write is needed either.

### 23.1. Synopsis

```

public class RTMPOriginConnection extends, org.?red5.?server.?net.?rtmp.?RTMPConnection {
// Public Constructors

    public RTMPOriginConnection(String type,
                                int clientId);

    public RTMPOriginConnection(String type,
                                int clientId,
                                int ioSessionId);

// Public Methods

```

```

    public synchronized void close();

    public int getIoSessionId();

    public org.red5.server.net.rtmp.codec.RTMP getState();

    public void rawWrite(org.apache.mina.common.ByteBuffer out);

    public synchronized void realClose();

    public void setHandler(org.red5.server.net.mrtmp.OriginMRTMPHandler handler);

    public void setMrtmpManager(org.red5.server.net.mrtmp.IMRTMPOriginManager mrtmpManager);

    public void startRoundTripMeasurement();

    public void write(org.red5.server.net.rtmp.message.Packet packet);

// Protected Methods

    protected void onInactive();

    protected void startWaitForHandshake(org.red5.server.api.scheduling.ISchedulingService service);

}

```

**Methods inherited from org.red5.server.net.rtmp.RTMPConnection:** addClientStream , close , closeChannel , connect , createOutputStream , createStreamName , deleteStreamById , getBandwidthConfigure , getChannel , getClientBytesRead , getEncoding , getId , getInvokeId , getLastPingTime , getNextAvailableChannelId , getParentBWControllable , getPendingCall , getPendingVideoMessages , getReadBytes , getState , getStreamByChannelId , getStreamById , getStreamIdForChannel , getStreams , getUsedStreamCount , getVideoCodecFactory , getWrittenBytes , invoke , isChannelUsed , messageDropped , messageReceived , messageSent , newBroadcastStream , newPlaylistSubscriberStream , newSingleItemSubscriberStream , notify , onInactive , ping , pingReceived , rawWrite , receivedBytesRead , registerDeferredResult , registerPendingCall , registerStream , rememberStreamBufferDuration , removeClientStream , reserveStreamId , retrievePendingCall , setBandwidthConfigure , setId , setMaxHandshakeTimeout , setMaxInactivity , setPingInterval , setSchedulingService , setState , setup , startRoundTripMeasurement , startWaitForHandshake , toString , unregisterDeferredResult , unregisterStream , unreserveStreamId , updateBytesRead , write , writingMessage

**Methods inherited from org.red5.server.BaseConnection:** dispatchEvent , getBasicScopes , getClient , getConnectParams , getDroppedMessages , getHost , getPath , getPendingMessages , getReadMessages , getRemoteAddress , getRemoteAddresses , getRemotePort , getScope , getSessionId , getType , getWrittenMessages , handleEvent , initialize , isConnected , notifyEvent , registerBasicScope , unregisterBasicScope

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttribute , getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute , getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute ,

```
getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute ,
hasAttribute , removeAttribute , removeAttributes , setAttribute , setAttributes
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notifyAll , wait

**Fields inherited from org.red5.server.net.rtmp.RTMPConnection:** clientId , deferredResults , encoding , invokeId , keepAliveJobName , lastPingSent , lastPingTime , lastPongReceived , log , maxInactivity , oName , pendingCalls , pingInterval , state , streamBuffers

**Fields inherited from org.red5.server.BaseConnection:** basicScopes , client , closed , droppedMessages , host , params , path , readMessages , remoteAddress , remoteAddresses , remotePort , scope , sessionId , type , writtenMessages

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 24. Class RTMPCUtils

RTMP utilities class.

### 24.1. Synopsis

```
public class RTMPCUtils implements org.?red5.?server.?net.?rtmp.?message.?Constants {
    // Public Constructors

    public RTMPCUtils();

    // Public Static Methods

    public static int decodeChannelId(int header,
                                      int byteCount);

    public static byte decodeHeaderSize(int header,
                                       int byteCount);

    public static void encodeHeaderByte(org.apache.mina.common.ByteBuffer out,
                                       byte headerSize,
                                       int channelId);

    public static int getHeaderLength(byte headerSize);

    public static int readMediumInt(org.apache.mina.common.ByteBuffer in);

    public static int readMediumIntOld(org.apache.mina.common.ByteBuffer in);

    public static int readReverseInt(org.apache.mina.common.ByteBuffer in);

    public static int readReverseIntOld(org.apache.mina.common.ByteBuffer in);

    public static int readUnsignedMediumInt(org.apache.mina.common.ByteBuffer in);

    public static int readUnsignedMediumIntOld(org.apache.mina.common.ByteBuffer in);
```

```

public static void writeMediumInt(org.apache.mina.common.ByteBuffer out,
                                int value);

public static void writeReverseInt(org.apache.mina.common.ByteBuffer out,
                                   int value);

public static void writeReverseIntOld(org.apache.mina.common.ByteBuffer out,
                                      int value);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 24.2. decodeChannelId(int, int)

```

public static int decodeChannelId(int header,
                                  int byteCount);

```

Decode channel id.

### Parameters

header	Header
<i>return</i>	Channel id

## 24.3. decodeHeaderSize(int, int)

```

public static byte decodeHeaderSize(int header,
                                    int byteCount);

```

Decode header size.

### Parameters

header	Header byte
<i>return</i>	Header size byte

## 24.4. encodeHeaderByte(ByteBuffer, byte, int)

```

public static void encodeHeaderByte(org.apache.mina.common.ByteBuffer out,
                                    byte headerSize,
                                    int channelId);

```

Encodes header size marker and channel id into header marker.

### Parameters

headerSize	Header size marker
channelId	Channel used
<i>return</i>	Header id

## 24.5. getHeaderLength(byte)

```
public static int getHeaderLength(byte headerSize);
```

Return header length from marker value.

### Parameters

headerSize	Header size marker value
<i>return</i>	Header length

## 24.6. readMediumInt(ByteBuffer)

```
public static int readMediumInt(org.apache.mina.common.ByteBuffer in);
```

### Parameters

in	
<i>return</i>	

## 24.7. readMediumIntOld(ByteBuffer)

```
public static int readMediumIntOld(org.apache.mina.common.ByteBuffer in);
```

### Parameters

in	
<i>return</i>	

## 24.8. readReverseInt(ByteBuffer)

```
public static int readReverseInt(org.apache.mina.common.ByteBuffer in);
```

Read integer in reversed order.

### Parameters

in	Input buffer
<i>return</i>	Integer

## 24.9. readReverseIntOld(ByteBuffer)

```
public static int readReverseIntOld(org.apache.mina.common.ByteBuffer in);
```

Read integer in reversed order.

### Parameters

in	Input buffer
<i>return</i>	Integer

## 24.10. readUnsignedMediumInt(ByteBuffer)

```
public static int readUnsignedMediumInt(org.apache.mina.common.ByteBuffer in);
```

### Parameters

in	
----	--

<i>return</i>	
---------------	--

## 24.11. readUnsignedMediumIntOld(ByteBuffer)

```
public static int readUnsignedMediumIntOld(org.apache.mina.common.ByteBuffer in);
```

### Parameters

in	
----	--

<i>return</i>	
---------------	--

## 24.12. writeMediumInt(ByteBuffer, int)

```
public static void writeMediumInt(org.apache.mina.common.ByteBuffer out,
                                 int value);
```

### Parameters

out	
-----	--

value	
-------	--

## 24.13. writeReverseInt(ByteBuffer, int)

```
public static void writeReverseInt(org.apache.mina.common.ByteBuffer out,
                                   int value);
```

Writes reversed integer to buffer.

### Parameters

out	Buffer
-----	--------

value	Integer to write
-------	------------------

## 24.14. writeReverseIntOld(ByteBuffer, int)

```
public static void writeReverseIntOld(org.apache.mina.common.ByteBuffer out,
                                      int value);
```

Writes reversed integer to buffer.

### Parameters

out	Buffer
-----	--------

value	Integer to write
-------	------------------

# 1. Interface IEventDecoder

Event decoder decodes event objects from incoming byte buffer.

## 1.1. Synopsis

```
public interface IEventDecoder {  
    // Public Methods  
  
    public org.red5.server.net.rtmp.event.AudioData decodeAudioData(org.apache.mina.common.ByteBuffer  
        in,  
        RTMP rtmp);  
  
    public org.red5.server.net.rtmp.event.BytesRead decodeBytesRead(org.apache.mina.common.ByteBuffer  
        in,  
        RTMP rtmp);  
  
    public org.red5.server.net.rtmp.event.ChunkSize decodeChunkSize(org.apache.mina.common.ByteBuffer  
        in,  
        RTMP rtmp);  
  
    public org.red5.server.net.rtmp.event.FlexMessage decodeFlexMessage(org.apache.mina.common.ByteBuffer  
        in,  
        RTMP rtmp);  
  
    public org.red5.server.so.ISharedObjectMessage decodeFlexSharedObject(org.apache.mina.common.ByteBuffer  
        in,  
        RTMP rtmp);  
  
    public org.red5.server.net.rtmp.event.Invoke decodeInvoke(org.apache.mina.common.ByteBuffer in,  
        RTMP rtmp);  
  
    public org.red5.server.net.rtmp.event.Notify decodeNotify(org.apache.mina.common.ByteBuffer in,  
        RTMP rtmp);  
  
    public org.red5.server.net.rtmp.event.Ping decodePing(org.apache.mina.common.ByteBuffer in);  
  
    public org.red5.server.so.ISharedObjectMessage decodeSharedObject(org.apache.mina.common.ByteBuffer  
        in,  
        RTMP rtmp);  
  
    public org.red5.server.net.rtmp.event.Unknown decodeUnknown(byte dataType,  
        org.apache.mina.common.ByteBuffer in);  
  
    public org.red5.server.net.rtmp.event.VideoData decodeVideoData(org.apache.mina.common.ByteBuffer  
        in);  
}
```

## 1.2. decodeAudioData(ByteBuffer)

```
public org.red5.server.net.rtmp.event.AudioData decodeAudioData(org.apache.mina.common.ByteBuffer in);
```

Decodes audio data event.

### Parameters

in	Byte buffer to decode
return	AudioData event

## 1.3. decodeBytesRead(ByteBuffer)

```
public org.red5.server.net.rtmp.event.BytesRead decodeBytesRead(org.apache.mina.common.ByteBuffer in);
```

Decodes BytesRead event.

#### Parameters

in	Byte buffer to decode
<i>return</i>	BytesRead event

## 1.4. decodeChunkSize(ByteBuffer)

```
public org.red5.server.net.rtmp.event.ChunkSize decodeChunkSize(org.apache.mina.common.ByteBuffer in,
```

Decodes chunk size event.

#### Parameters

in	Byte buffer to decode
<i>return</i>	ChunkSize event

## 1.5. decodeFlexMessage(ByteBuffer, RTMP)

```
public org.red5.server.net.rtmp.event.FlexMessage decodeFlexMessage(org.apache.mina.common.ByteBuffer in,
```

```
RTMP rtmp);
```

Decodes Flex message event.

#### Parameters

in	Byte buffer to decode
rtmp	RTMP protocol state
<i>return</i>	FlexMessage event

## 1.6. decodeFlexSharedObject(ByteBuffer, RTMP)

```
public org.red5.server.so.ISharedObjectMessage decodeFlexSharedObject(org.apache.mina.common.ByteBuffer in,
```

```
RTMP rtmp);
```

Decodes shared object message event from AMF3 encoding.

#### Parameters

in	Byte buffer to decode
rtmp	RTMP protocol state
<i>return</i>	ISharedObjectMessage event

## 1.7. decodeInvoke(ByteBuffer, RTMP)

```
public org.red5.server.net.rtmp.event.Invoke decodeInvoke(org.apache.mina.common.ByteBuffer in,
```

```
RTMP rtmp);
```

Decodes invocation event.

**Parameters**

<i>in</i>	Byte buffer to decode
<i>rtmp</i>	RTMP protocol state
<i>return</i>	Invoke event

**1.8. decodeNotify(ByteBuffer, RTMP)**

```
public org.red5.server.net.rtmp.event.Notify decodeNotify(org.apache.mina.common.ByteBuffer in,
                                                       RTMP rtmp);
```

Decodes notification event.

**Parameters**

<i>in</i>	Byte buffer to decode
<i>rtmp</i>	RTMP protocol state
<i>return</i>	Notify event

**1.9. decodePing(ByteBuffer)**

```
public org.red5.server.net.rtmp.event.Ping decodePing(org.apache.mina.common.ByteBuffer in);
```

Decodes ping event.

**Parameters**

<i>in</i>	Byte buffer to decode
<i>return</i>	Ping event

**1.10. decodeSharedObject(ByteBuffer, RTMP)**

```
public org.red5.server.so.ISharedObjectMessage decodeSharedObject(org.apache.mina.common.ByteBuffer
                                                               RTMP rtmp);
```

Decodes shared object message event.

**Parameters**

<i>in</i>	Byte buffer to decode
<i>rtmp</i>	RTMP protocol state
<i>return</i>	ISharedObjectMessage event

**1.11. decodeUnknown(byte, ByteBuffer)**

```
public org.red5.server.net.rtmp.event.Unknown decodeUnknown(byte dataType,
                                                             org.apache.mina.common.ByteBuffer in);
```

Decodes event of Unknown type.

**Parameters**

<b>dataType</b>	Data type
<b>in</b>	Byte buffer to decode
<b>return</b>	Unknown event

**1.12. decodeVideoData(ByteBuffer)**

```
public org.red5.server.net.rtmp.event.VideoData decodeVideoData(org.apache.mina.common.ByteBuffer in)
```

Decodes video data event.

**Parameters**

<b>in</b>	Byte buffer to decode
<b>return</b>	VideoData event

**2. Interface IEventEncoder**

Encodes events to byte buffer.

**2.1. Synopsis**

```
public interface IEventEncoder {
    // Public Methods

    public org.apache.mina.common.ByteBuffer encodeAudioData(org.red5.server.net.rtmp.event.AudioData
    public org.apache.mina.common.ByteBuffer encodeBytesRead(org.red5.server.net.rtmp.event.BytesRead
    public org.apache.mina.common.ByteBuffer encodeChunkSize(org.red5.server.net.rtmp.event.ChunkSize
    public org.apache.mina.common.ByteBuffer encodeFlexSharedObject(org.red5.server.so.ISharedObjectMe
                    RTMP rtmp);
    public org.apache.mina.common.ByteBuffer encodeInvoke(org.red5.server.net.rtmp.event.Invoke invoke
                    RTMP rtmp);
    public org.apache.mina.common.ByteBuffer encodeNotify(org.red5.server.net.rtmp.event.Notify notify
                    RTMP rtmp);
    public org.apache.mina.common.ByteBuffer encodePing(org.red5.server.net.rtmp.event.Ping ping);
    public org.apache.mina.common.ByteBuffer encodeSharedObject(org.red5.server.so.ISharedObjectMessag
                    RTMP rtmp);
    public org.apache.mina.common.ByteBuffer encodeUnknown(org.red5.server.net.rtmp.event.Unknown unkno
    public org.apache.mina.common.ByteBuffer encodeVideoData(org.red5.server.net.rtmp.event.VideoData
}
```

## 2.2. encodeAudioData(AudioData)

```
public org.apache.mina.common.ByteBuffer encodeAudioData(org.red5.server.net.rtmp.event.AudioData au
```

Encodes AudioData event to byte buffer.

### Parameters

audioData	AudioData event
<i>return</i>	Byte buffer

## 2.3. encodeBytesRead(BytesRead)

```
public org.apache.mina.common.ByteBuffer encodeBytesRead(org.red5.server.net.rtmp.event.BytesRead st
```

Encodes BytesRead event to byte buffer.

### Parameters

streamBytesRead	BytesRead event
<i>return</i>	Byte buffer

## 2.4. encodeChunkSize(ChunkSize)

```
public org.apache.mina.common.ByteBuffer encodeChunkSize(org.red5.server.net.rtmp.event.ChunkSize ch
```

Encodes ChunkSize event to byte buffer.

### Parameters

chunkSize	ChunkSize event
<i>return</i>	Byte buffer

## 2.5. encodeFlexSharedObject(ISharedObjectMessage, RTMP)

```
public org.apache.mina.common.ByteBuffer encodeFlexSharedObject(org.red5.server.so.ISharedObjectMessag
```

Encodes SharedObjectMessage event to byte buffer using AMF3 encoding.

### Parameters

so	ISharedObjectMessage event
rtmp	RTMP protocol state
<i>return</i>	Byte buffer

## 2.6. encodeInvoke(Invoke, RTMP)

```
public org.apache.mina.common.ByteBuffer encodeInvoke(org.red5.server.net.rtmp.event.Invoke invoke,
```

RTMP rtmp);

Encodes Invoke event to byte buffer.

Parameters	
invoke	Invoke event
rtmp	RTMP protocol state
<i>return</i>	Byte buffer

## 2.7. encodeNotify(Notify, RTMP)

```
public org.apache.mina.common.ByteBuffer encodeNotify(org.red5.server.net.rtmp.event.Notify notify,
                                                    RTMP rtmp);
```

Encodes Notify event to byte buffer.

Parameters	
notify	Notify event
rtmp	RTMP protocol state
<i>return</i>	Byte buffer

## 2.8. encodePing(Ping)

```
public org.apache.mina.common.ByteBuffer encodePing(org.red5.server.net.rtmp.event.Ping ping);
```

Encodes Ping event to byte buffer.

Parameters	
ping	Ping event
<i>return</i>	Byte buffer

## 2.9. encodeSharedObject(ISharedObjectMessage, RTMP)

```
public org.apache.mina.common.ByteBuffer encodeSharedObject(org.red5.server.so.ISharedObjectMessage
                                                            RTMP rtmp);
```

Encodes SharedObjectMessage event to byte buffer.

Parameters	
so	ISharedObjectMessage event
rtmp	RTMP protocol state
<i>return</i>	Byte buffer

## 2.10. encodeUnknown(Unknown)

```
public org.apache.mina.common.ByteBuffer encodeUnknown(org.red5.server.net.rtmp.event.Unknown unknown);
```

Encodes Unknown event to byte buffer.

**Parameters**

unknown	Unknown event
<i>return</i>	Byte buffer

**2.11. encodeVideoData(VideoData)**

```
public org.apache.mina.common.ByteBuffer encodeVideoData(org.red5.server.net.rtmp.event.VideoData vi
```

Encodes VideoData event to byte buffer.

**Parameters**

videoData	VideoData event
<i>return</i>	Byte buffer

**3. Class MulticastEventProcessor**

Processes multicast events.

**3.1. Synopsis**

```
public class MulticastEventProcessor {
// Public Constructors

    public MulticastEventProcessor();

// Public Static Methods

    public static org.apache.mina.common.ByteBuffer[] chunkBuffer(org.apache.mina.common.ByteBuffer bu
                                                               int size);

// Public Methods

    public void disposeCached(Object obj);

    public byte getCacheId();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

**3.2. chunkBuffer(ByteBuffer, int)**

```
public static org.apache.mina.common.ByteBuffer[] chunkBuffer(org.apache.mina.common.ByteBuffer buf
                                                               int size);
```

Breaks buffer into chunks of given size.

**Parameters**

buf	Byte buffer
size	Chunk size
<i>return</i>	Array of byte buffers, chunks

### 3.3. disposeCached(Object)

```
public void disposeCached(Object obj);
```

Disposes cached object.

#### Parameters

obj	Cached object
-----	---------------

### 3.4. getCacheld()

```
public byte getCacheld();
```

Getter for cache ID.

#### Parameters

<i>return</i>	Cache ID
---------------	----------

## 4. Class RTMP

RTMP is the RTMP protocol state representation.

### 4.1. Synopsis

```
public class RTMP extends, org.?red5.?server.?net.?protocol.?ProtocolState {
// Public Static Fields

    public static final int DEFAULT_CHUNK_SIZE = 128;

    public static final boolean MODE_CLIENT = true;

    public static final boolean MODE_SERVER = false;

    public static final byte STATE_CONNECT = 0;

    public static final byte STATE_CONNECTED = 2;

    public static final byte STATE_DISCONNECTED = 4;

    public static final byte STATE_EDGE_CONNECT_ORIGIN_SENT = 17;

    public static final byte STATE_EDGE_DISCONNECTING = 19;

    public static final byte STATE_ERROR = 3;

    public static final byte STATE_HANDSHAKE = 1;
```

## Package org.?red5.?server.?net.?rtmp.?codec

```
public static final byte STATE_ORIGIN_CONNECT_FORWARDED = 18;

// Public Constructors

public RTMP(boolean mode);

// Public Methods

public org.red5.server.api.IConnection.Encoding getEncoding();

public int getLastReadChannel();

public org.red5.server.net.rtmp.message.Header getLastReadHeader(int channelId);

public org.red5.server.net.rtmp.message.Packet getLastReadPacket(int channelId);

public int getLastWriteChannel();

public org.red5.server.net.rtmp.message.Header getLastWriteHeader(int channelId);

public org.red5.server.net.rtmp.message.Packet getLastWritePacket(int channelId);

public boolean getMode();

public int getReadChunkSize();

public byte getState();

public int getWriteChunkSize();

public boolean isDebug();

public void setDebug(boolean debug);

public void setEncoding(org.red5.server.api.IConnection.Encoding encoding);

public void setHandshake(org.apache.mina.common.ByteBuffer data,
                      int start,
                      int length);

public void setLastReadHeader(int channelId,
                            org.red5.server.net.rtmp.message.Header header);

public void setLastReadPacket(int channelId,
                            org.red5.server.net.rtmp.message.Packet packet);

public void setLastWriteHeader(int channelId,
                            org.red5.server.net.rtmp.message.Header header);

public void setLastWritePacket(int channelId,
                            org.red5.server.net.rtmp.message.Packet packet);

public void setReadChunkSize(int readChunkSize);

public void setState(byte state);

public void setWriteChunkSize(int writeChunkSize);
```

```

    public boolean validateHandshakeReply(org.apache.mina.common.ByteBuffer data,
                                         int start,
                                         int length);

}

```

**Methods inherited from org.red5.server.net.protocol.ProtocolState:** bufferDecoding , canContinueDecoding , canStartDecoding , continueDecoding , getDecoderBufferAmount , hasDecodedObject , startDecoding

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.protocol.ProtocolState:** DECODER\_BUFFER , DECODER\_CONTINUE , DECODER\_OK , SESSION\_KEY

## 4.2. RTMP(boolean)

```
public RTMP(boolean mode);
```

Creates RTMP object with initial mode.

### Parameters

mode	Initial mode
------	--------------

## 4.3. DEFAULT\_CHUNK\_SIZE

```
public static final int DEFAULT_CHUNK_SIZE = 128;
```

Default chunk size. Packets are read and written chunk-by-chunk.

## 4.4. MODE\_CLIENT

```
public static final boolean MODE_CLIENT = true;
```

Client mode.

## 4.5. MODE\_SERVER

```
public static final boolean MODE_SERVER = false;
```

Server mode.

## 4.6. STATE\_CONNECT

```
public static final byte STATE_CONNECT = 0;
```

Connect state.

## 4.7. STATE\_CONNECTED

```
public static final byte STATE_CONNECTED = 2;
```

Connected.

## 4.8. STATE\_DISCONNECTED

```
public static final byte STATE_DISCONNECTED = 4;
```

Disconnected.

## 4.9. STATE\_EDGE\_CONNECT\_ORIGIN\_SENT

```
public static final byte STATE_EDGE_CONNECT_ORIGIN_SENT = 17;
```

Sent the connect message to origin.

## 4.10. STATE\_EDGE\_DISCONNECTING

```
public static final byte STATE_EDGE_DISCONNECTING = 19;
```

Edge is disconnecting, waiting Origin close connection.

## 4.11. STATE\_ERROR

```
public static final byte STATE_ERROR = 3;
```

Error.

## 4.12. STATE\_HANDSHAKE

```
public static final byte STATE_HANDSHAKE = 1;
```

Handshake state. Server sends handshake request to client right after connection estabilished.

## 4.13. STATE\_ORIGIN\_CONNECT\_FORWARDED

```
public static final byte STATE_ORIGIN_CONNECT_FORWARDED = 18;
```

Forwarded client's connect call to origin.

## 4.14. getEncoding()

```
public org.red5.server.api.IConnection.Encoding getEncoding();
```

Getter for encoding version.

### Parameters

<i>return</i>	Encoding version
---------------	------------------

## 4.15. getLastReadChannel()

```
public int getLastReadChannel();
```

Return channel being read last.

#### Parameters

<i>return</i>	Last read channel
---------------	-------------------

### 4.16. getLastReadHeader(int)

```
public org.red5.server.net.rtmp.message.Header getLastReadHeader(int channelId);
```

Return last read header for channel.

#### Parameters

channelId	Channel id
-----------	------------

<i>return</i>	Last read header
---------------	------------------

### 4.17. getLastReadPacket(int)

```
public org.red5.server.net.rtmp.message.Packet getLastReadPacket(int channelId);
```

Return last read packet for channel.

#### Parameters

channelId	Channel id
-----------	------------

<i>return</i>	Last read packet for that channel
---------------	-----------------------------------

### 4.18. getLastWriteChannel()

```
public int getLastWriteChannel();
```

Getter for channel being written last.

#### Parameters

<i>return</i>	Last write channel
---------------	--------------------

### 4.19. getLastWriteHeader(int)

```
public org.red5.server.net.rtmp.message.Header getLastWriteHeader(int channelId);
```

Return last written header for channel.

#### Parameters

channelId	Channel id
-----------	------------

<i>return</i>	Last written header
---------------	---------------------

### 4.20. getLastWritePacket(int)

```
public org.red5.server.net.rtmp.message.Packet getLastWritePacket(int channelId);
```

Return packet that has been written last.

#### Parameters

channelId	Channel id
<i>return</i>	Packet that has been written last

### 4.21. getMode()

```
public boolean getMode();
```

Return current mode.

#### Parameters

<i>return</i>	Current mode
---------------	--------------

### 4.22. getReadChunkSize()

```
public int getReadChunkSize();
```

Getter for write chunk size. Data is being read chunk-by-chunk.

#### Parameters

<i>return</i>	Read chunk size
---------------	-----------------

### 4.23. getState()

```
public byte getState();
```

Return current state.

#### Parameters

<i>return</i>	State
---------------	-------

### 4.24. getWriteChunkSize()

```
public int getWriteChunkSize();
```

Getter for write chunk size. Data is being written chunk-by-chunk.

#### Parameters

<i>return</i>	Write chunk size
---------------	------------------

### 4.25. isDebug()

```
public boolean isDebug();
```

Getter for debug.

## Parameters

<i>return</i>	Debug state
---------------	-------------

**4.26. setDebug(boolean)**

```
public void setDebug(boolean debug);
```

Setter for debug.

## Parameters

debug	Debug flag new value
-------	----------------------

**4.27. setEncoding(IConnection.Encoding)**

```
public void setEncoding(org.red5.server.api.IConnection.Encoding encoding);
```

Setter for encoding version.

## Parameters

encoding	Encoding version
----------	------------------

**4.28. setHandshake(ByteBuffer, int, int)**

```
public void setHandshake(org.apache.mina.common.ByteBuffer data,
                        int start,
                        int length);
```

Store the handshake sent to the client.

## Parameters

data	Handshake data
length	Length of handshake to store

**4.29. setLastReadHeader(int, Header)**

```
public void setLastReadHeader(int channelId,
                             org.red5.server.net.rtmp.message.Header header);
```

Setter for last read header.

## Parameters

channelId	Channel id
header	Header

**4.30. setLastReadPacket(int, Packet)**

```
public void setLastReadPacket(int channelId,
```

```
org.red5.server.net.rtmp.message.Packet packet);
```

Setter for last read packet.

Parameters	
channelId	Channel id
packet	Packet

#### 4.31. setLastWriteHeader(int, Header)

```
public void setLastWriteHeader(int channelId,
                               org.red5.server.net.rtmp.message.Header header);
```

Setter for last written header.

Parameters	
channelId	Channel id
header	Header

#### 4.32. setLastWritePacket(int, Packet)

```
public void setLastWritePacket(int channelId,
                               org.red5.server.net.rtmp.message.Packet packet);
```

Setter for last written packet.

Parameters	
channelId	Channel id
packet	Last written packet

#### 4.33. setReadChunkSize(int)

```
public void setReadChunkSize(int readChunkSize);
```

Setter for read chunk size. Data is being read chunk-by-chunk.

Parameters	
readChunkSize	Value to set for property 'readChunkSize'.

#### 4.34. setState(byte)

```
public void setState(byte state);
```

Setter for state.

Parameters	
state	New state

## 4.35. setWriteChunkSize(int)

```
public void setWriteChunkSize(int writeChunkSize);
```

Setter for write chunk size.

### Parameters

writeChunkSize	Write chunk size
----------------	------------------

## 4.36. validateHandshakeReply(ByteBuffer, int, int)

```
public boolean validateHandshakeReply(org.apache.mina.common.ByteBuffer data,
                                      int start,
                                      int length);
```

Check if the handshake reply received from a client contains valid data.

### Parameters

data	
------	--

length	
--------	--

return	
--------	--

## 5. Class RTMPCodecFactory

RTMP codec factory creates RTMP encoders/decoders.

### 5.1. Synopsis

```
public class RTMPCodecFactory implements org.apache.mina.filter.codec.ProtocolCodecFactory, org.apache.mina.filter.codec.SimpleProtocolDecoder, org.apache.mina.filter.codec.SimpleProtocolEncoder {
    // Protected Fields
    protected RTMPMinaProtocolDecoder decoder;
    protected org.red5.io.object.Deserializer deserializer;
    protected RTMPMinaProtocolEncoder encoder;
    protected org.red5.io.object.Serializer serializer;
    // Public Constructors
    public RTMPCodecFactory();
    // Public Methods
    public org.apache.mina.filter.codec.ProtocolDecoder getDecoder();
    public org.apache.mina.filter.codec.ProtocolEncoder getEncoder();
    public org.red5.server.net.protocol.SimpleProtocolDecoder getSimpleDecoder();
    public org.red5.server.net.protocol.SimpleProtocolEncoder getSimpleEncoder();
}
```

```

    public void init();

    public void setDeserializer(org.red5.io.object.Deserializer deserializer);

    public void setSerializer(org.red5.io.object.Serializer serializer);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 5.2. decoder

```
protected RTMPMinaProtocolDecoder decoder;
```

Mina protocol decoder for RTMP.

## 5.3. deserializer

```
protected org.red5.io.object.Deserializer deserializer;
```

Deserializer.

## 5.4. encoder

```
protected RTMPMinaProtocolEncoder encoder;
```

Mina protocol encoder for RTMP.

## 5.5. serializer

```
protected org.red5.io.object.Serializer serializer;
```

Serializer.

## 5.6. getDecoder()

```
public org.apache.mina.filter.codec.ProtocolDecoder getDecoder();
```

**Specified by:** Method `getDecoder` in interface `ProtocolCodecFactory`

## 5.7. getEncoder()

```
public org.apache.mina.filter.codec.ProtocolEncoder getEncoder();
```

**Specified by:** Method `getEncoder` in interface `ProtocolCodecFactory`

## 5.8. getSimpleDecoder()

```
public org.red5.server.net.protocol.SimpleProtocolDecoder getSimpleDecoder();
```

**Specified by:** Method `getSimpleDecoder` in interface `SimpleProtocolCodecFactory`

Getter for simple decoder.

## 5.9. getSimpleEncoder()

```
public org.red5.server.net.protocol.SimpleProtocolEncoder getSimpleEncoder();
```

**Specified by:** Method getSimpleEncoder in interface SimpleProtocolCodecFactory

Getter for simple encoder.

## 5.10. init()

```
public void init();
```

Initialization

## 5.11. setDeserializer(Deserializer)

```
public void setDeserializer(org.red5.io.object.Deserializer deserializer);
```

Setter for deserializer.

Parameters

deserializer	Deserializer
--------------	--------------

## 5.12. setSerializer(Serializer)

```
public void setSerializer(org.red5.io.object.Serializer serializer);
```

Setter for serializer.

Parameters

serializer	Serializer
------------	------------

# 6. Class RTMPMinaCodecFactory

RTMP codec factory.

## 6.1. Synopsis

```
public class RTMPMinaCodecFactory implements org.apache.mina.filter.codec.ProtocolCodecFactory
// Protected Fields

protected RTMPMinaProtocolDecoder decoder ;

protected RTMPMinaProtocolEncoder encoder ;

// Public Constructors

public RTMPMinaCodecFactory();

// Public Methods

public org.apache.mina.filter.codec.ProtocolDecoder getDecoder();
```

```

public org.apache.mina.filter.codec.ProtocolEncoder getEncoder();

public RTMPMinaProtocolDecoder getMinaDecoder();

public RTMPMinaProtocolEncoder getMinaEncoder();

public org.red5.server.net.protocol.SimpleProtocolDecoder getSimpleDecoder();

public org.red5.server.net.protocol.SimpleProtocolEncoder getSimpleEncoder();

public void init();

public void setMinaDecoder(RTMPMinaProtocolDecoder decoder);

public void setMinaEncoder(RTMPMinaProtocolEncoder encoder);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 6.2. decoder

```
protected RTMPMinaProtocolDecoder decoder ;
```

RTMP Mina protocol decoder.

## 6.3. encoder

```
protected RTMPMinaProtocolEncoder encoder ;
```

RTMP Mina protocol encoder.

## 6.4. getDecoder()

```
public org.apache.mina.filter.codec.ProtocolDecoder getDecoder();
```

**Specified by:** Method getDecoder in interface ProtocolCodecFactory

## 6.5. getEncoder()

```
public org.apache.mina.filter.codec.ProtocolEncoder getEncoder();
```

**Specified by:** Method getEncoder in interface ProtocolCodecFactory

## 6.6. getMinaDecoder()

```
public RTMPMinaProtocolDecoder getMinaDecoder();
```

## 6.7. getMinaEncoder()

```
public RTMPMinaProtocolEncoder getMinaEncoder();
```

## 6.8. getSimpleDecoder()

```
public org.red5.server.net.protocol.SimpleProtocolDecoder getSimpleDecoder();
```

**Specified by:** Method getSimpleDecoder in interface SimpleProtocolCodecFactory

Getter for simple decoder.

## 6.9. getSimpleEncoder()

```
public org.red5.server.net.protocol.SimpleProtocolEncoder getSimpleEncoder();
```

**Specified by:** Method getSimpleEncoder in interface SimpleProtocolCodecFactory

Getter for simple encoder.

## 6.10. init()

```
public void init();
```

Initialization. Create and setup of encoder/decoder and serializer/deserializer is handled by Spring.

## 6.11. setMinaDecoder(RTMPMinaProtocolDecoder)

```
public void setMinaDecoder(RTMPMinaProtocolDecoder decoder);
```

Setter for decoder

### Parameters

decoder	Decoder
---------	---------

## 6.12. setMinaEncoder(RTMPMinaProtocolEncoder)

```
public void setMinaEncoder(RTMPMinaProtocolEncoder encoder);
```

Setter for encoder.

### Parameters

encoder	Encoder
---------	---------

# 7. Class RTMPMinaProtocolDecoder

RTMP protocol decoder.

## 7.1. Synopsis

```
public class RTMPMinaProtocolDecoder extends, org.red5.server.net.rtmp.codec.RTMPProtocolDecoder
    implements, org.apache.mina.filter.codec.ProtocolDecoder {
// Public Constructors

    public RTMPMinaProtocolDecoder();
```

```
// Public Methods

    public void decode(org.apache.mina.common.IoSession session,
                      org.apache.mina.common.ByteBuffer in,
                      org.apache.mina.filter.codec.ProtocolDecoderOutput out)
    throws ProtocolCodecException;

    public void dispose(org.apache.mina.common.IoSession ioSession)
    throws Exception;

    public void finishDecode(org.apache.mina.common.IoSession session,
                           org.apache.mina.filter.codec.ProtocolDecoderOutput out)
    throws Exception;

}
```

**Methods inherited from org.red5.server.net.rtmp.codec.RTMPProtocolDecoder:**

decode , decodeAudioData , decodeBuffer , decodeBytesRead , decodeChunkSize ,  
 decodeFlexMessage , decodeFlexSharedObject , decodeFlexStreamSend , decodeHandshake  
 , decodeHeader , decodeInvoke , decodeMessage , decodeNotify , decodeNotifyOrInvoke  
 , decodePacket , decodePing , decodeSharedObject , decodeStreamMetadata ,  
 decodeUnknown , decodeVideoData , doDecodeSharedObject , setDeserializer ,  
 setupClassLoader

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
 , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.codec.RTMPProtocolDecoder:** ioLog ,  
 log

## 7.2. decode(IoSession, ByteBuffer, ProtocolDecoderOutput)

```
public void decode(org.apache.mina.common.IoSession session,
                  org.apache.mina.common.ByteBuffer in,
                  org.apache.mina.filter.codec.ProtocolDecoderOutput out)
throws ProtocolCodecException;
```

**Specified by:** Method decode in interface ProtocolDecoder

## 7.3. dispose(IoSession)

```
public void dispose(org.apache.mina.common.IoSession ioSession)
throws Exception;
```

**Specified by:** Method dispose in interface ProtocolDecoder

## 7.4. finishDecode(IoSession, ProtocolDecoderOutput)

```
public void finishDecode(org.apache.mina.common.IoSession session,
                        org.apache.mina.filter.codec.ProtocolDecoderOutput out)
throws Exception;
```

**Specified by:** Method finishDecode in interface ProtocolDecoder

## 8. Class RTMPMinaProtocolEncoder

Mina protocol encoder for RTMP.

## 8.1. Synopsis

```
public class RTMPMinaProtocolEncoder extends, org.red5.server.net.rtmp.codec.RTMPPProtocolEncoder
    implements, org.apache.mina.filter.codec.ProtocolEncoder {
    // Public Constructors

    public RTMPMinaProtocolEncoder();

    // Public Methods

    public void dispose(org.apache.mina.common.IoSession ioSession)
        throws Exception;

    public void encode(org.apache.mina.common.IoSession session,
                      Object message,
                      org.apache.mina.filter.codec.ProtocolEncoderOutput out)
        throws ProtocolCodecException;

}
```

### Methods inherited from org.red5.server.net.rtmp.codec.RTMPPProtocolEncoder:

doEncodeSharedObject , encode , encodeAudioData , encodeBytesRead , encodeChunkSize , encodeFlexMessage , encodeFlexSharedObject , encodeFlexStreamSend , encodeHeader , encodeInvoke , encodeMessage , encodeNotify , encodeNotifyOrInvoke , encodePacket , encodePing , encodeSharedObject , encodeStreamMetadata , encodeUnknown , encodeVideoData , setSerializer

### Methods inherited from org.red5.server.net.protocol.BaseProtocolEncoder:

generateErrorResult

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.codec.RTMPPProtocolEncoder:** ioLog , log

## 8.2. dispose(IoSession)

```
public void dispose(org.apache.mina.common.IoSession ioSession)
    throws Exception;
```

**Specified by:** Method dispose in interface ProtocolEncoder

## 8.3. encode(IoSession, Object, ProtocolEncoderOutput)

```
public void encode(org.apache.mina.common.IoSession session,
                  Object message,
                  org.apache.mina.filter.codec.ProtocolEncoderOutput out)
    throws ProtocolCodecException;
```

**Specified by:** Method encode in interface ProtocolEncoder

## 9. Class RTMPPProtocolDecoder

RTMP protocol decoder.

## 9.1. Synopsis

```

public class RTMPPProtocolDecoder implements, org.?red5.?server.?net.?rtmp.?message.?Constants, org.?re
// Protected Fields

protected static org.slf4j.Logger ioLog ;

protected static org.slf4j.Logger log ;

// Public Constructors

public RTMPPProtocolDecoder();

// Public Methods

public Object decode(org.red5.server.net.protocol.ProtocolState state,
                     org.apache.mina.common.ByteBuffer in)
throws ProtocolException;

public org.red5.server.net.rtmp.event.AudioData decodeAudioData(org.apache.mina.common.ByteBuffer

public java.util.List decodeBuffer(org.red5.server.net.protocol.ProtocolState state,
                                   org.apache.mina.common.ByteBuffer buffer);

public org.red5.server.net.rtmp.event.BytesRead decodeBytesRead(org.apache.mina.common.ByteBuffer

public org.red5.server.net.rtmp.event.ChunkSize decodeChunkSize(org.apache.mina.common.ByteBuffer

public org.red5.server.net.rtmp.event.FlexMessage decodeFlexMessage(org.apache.mina.common.ByteBuf
RTMP rtmp);

public org.red5.server.so.ISharedObjectMessage decodeFlexSharedObject(org.apache.mina.common.ByteB
RTMP rtmp);

public org.red5.server.net.rtmp.event.FlexStreamSend decodeFlexStreamSend(org.apache.mina.common.B
RTMP rtmp);

public org.apache.mina.common.ByteBuffer decodeHandshake(RTMP rtmp,
org.apache.mina.common.ByteBuffer in);

public org.red5.server.net.rtmp.message.Header decodeHeader(org.apache.mina.common.ByteBuffer in,
org.red5.server.net.rtmp.message.Heade

public org.red5.server.net.rtmp.event.Invoke decodeInvoke(org.apache.mina.common.ByteBuffer in,
RTMP rtmp);

public org.red5.server.net.rtmp.event.IRTMPEvent decodeMessage(RTMP rtmp,
org.red5.server.net.rtmp.message.He
org.apache.mina.common.ByteBuffer i

public org.red5.server.net.rtmp.event.Notify decodeNotify(org.apache.mina.common.ByteBuffer in,
RTMP rtmp);

public org.red5.server.net.rtmp.event.Notify decodeNotify(org.apache.mina.common.ByteBuffer in,
org.red5.server.net.rtmp.message.Header
RTMP rtmp);

public org.red5.server.net.rtmp.message.Packet decodePacket(RTMP rtmp,

```

```

        org.apache.mina.common.ByteBuffer in);

    public org.red5.server.net.rtmp.event.Ping decodePing(org.apache.mina.common.ByteBuffer in);

    public org.red5.server.so.ISharedObjectMessage decodeSharedObject(org.apache.mina.common.ByteBuffer
        RTMP rtmp);

    public org.red5.server.net.rtmp.event.Notify decodeStreamMetadata(org.apache.mina.common.ByteBuffe
        RTMP rtmp);

    public org.red5.server.net.rtmp.event.Unknown decodeUnknown(byte dataType,
        org.apache.mina.common.ByteBuffer in);

    public org.red5.server.net.rtmp.event.VideoData decodeVideoData(org.apache.mina.common.ByteBuffer
        RTMP rtmp);

    public void setDeserializer(org.red5.io.object.Deserializer deserializer);

// Protected Methods

    protected org.red5.server.net.rtmp.event.Notify decodeNotifyOrInvoke(org.red5.server.net.rtmp.event
        org.apache.mina.common.ByteBuffer in,
        org.red5.server.net.rtmp.message
        RTMP rtmp);

    protected void doDecodeSharedObject(org.red5.server.so.SharedObjectMessage so,
        org.apache.mina.common.ByteBuffer in,
        org.red5.io.object.Input input);

    protected void setupClassLoader();

}

}

```

**Direct known subclasses:** org.red5.server.net.rtmp.codec.

RTMPMinaProtocolDecoder , org.red5.server.net.rtmp.codec.RTMPPTProtocolDecoder

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 9.2. RTMPProtocolDecoder()

```
public RTMPProtocolDecoder();
```

Constructs a new RTMPProtocolDecoder.

## 9.3. ioLog

```
protected static org.slf4j.Logger ioLog;
```

I/O logger.

## 9.4. log

```
protected static org.slf4j.Logger log;
```

Logger.

## 9.5. decode(ProtocolState, ByteBuffer)

```
public Object decode(org.red5.server.net.protocol.ProtocolState state,
                    org.apache.mina.common.ByteBuffer in)
                    throws ProtocolException;
```

**Specified by:** Method decode in interface SimpleProtocolDecoder

Decodes byte buffer.

### Parameters

state	Protocol state
in	Input byte buffer
<i>return</i>	Decoded object

ProtocolException

Exception during decoding

## 9.6. decodeAudioData(ByteBuffer)

```
public org.red5.server.net.rtmp.event.AudioData decodeAudioData(org.apache.mina.common.ByteBuffer in)
```

**Specified by:** Method decodeAudioData in interface IEventDecoder

Decodes audio data event.

## 9.7. decodeBuffer(ProtocolState, ByteBuffer)

```
public java.util.List decodeBuffer(org.red5.server.net.protocol.ProtocolState state,
                                  org.apache.mina.common.ByteBuffer buffer);
```

**Specified by:** Method decodeBuffer in interface SimpleProtocolDecoder

Decode all available objects in buffer.

## 9.8. decodeBytesRead(ByteBuffer)

```
public org.red5.server.net.rtmp.event.BytesRead decodeBytesRead(org.apache.mina.common.ByteBuffer in)
```

**Specified by:** Method decodeBytesRead in interface IEventDecoder

Decodes BytesRead event.

## 9.9. decodeChunkSize(ByteBuffer)

```
public org.red5.server.net.rtmp.event.ChunkSize decodeChunkSize(org.apache.mina.common.ByteBuffer in)
```

**Specified by:** Method decodeChunkSize in interface IEventDecoder

Decodes chunk size event.

## 9.10. decodeFlexMessage(ByteBuffer, RTMP)

```
public org.red5.server.net.rtmp.event.FlexMessage decodeFlexMessage(org.apache.mina.common.ByteBuffer in)
```

```
RTMP rtmp);
```

**Specified by:** Method decodeFlexMessage in interface IEventDecoder

Decodes FlexMessage event.

#### Parameters

in	Byte buffer
rtmp	RTMP protocol state
<i>return</i>	FlexMessage event

## 9.11. decodeFlexSharedObject(ByteBuffer, RTMP)

```
public org.red5.server.so.ISharedObjectMessage decodeFlexSharedObject(org.apache.mina.common.ByteBuf  
RTMP rtmp);
```

**Specified by:** Method decodeFlexSharedObject in interface IEventDecoder

Decodes shared object message event from AMF3 encoding.

## 9.12. decodeHandshake(RTMP, ByteBuffer)

```
public org.apache.mina.common.ByteBuffer decodeHandshake(RTMP rtmp,  
org.apache.mina.common.ByteBuffer in);
```

Decodes handshake message.

#### Parameters

rtmp	RTMP protocol state
in	Byte buffer
<i>return</i>	Byte buffer

## 9.13. decodeHeader(ByteBuffer, Header)

```
public org.red5.server.net.rtmp.message.Header decodeHeader(org.apache.mina.common.ByteBuffer in,  
org.red5.server.net.rtmp.message.Header
```

Decodes packet header.

#### Parameters

in	Input byte buffer
lastHeader	Previous header
<i>return</i>	Decoded header

## 9.14. decodeInvoke(ByteBuffer, RTMP)

```
public org.red5.server.net.rtmp.event.Invoke decodeInvoke(org.apache.mina.common.ByteBuffer in,
```

```
RTMP rtmp);
```

**Specified by:** Method decodeInvoke in interface IEventDecoder

Decodes invocation event.

## 9.15. decodeMessage(RTMP, Header, ByteBuffer)

```
public org.red5.server.net.rtmp.event.IRTMPEvent decodeMessage(RTMP rtmp,
    org.red5.server.net.rtmp.message.Header header,
    org.apache.mina.common.ByteBuffer in)
```

Decodes RTMP message event.

### Parameters

rtmp	RTMP protocol state
header	RTMP header
in	Input byte buffer
<i>return</i>	RTMP event

## 9.16. decodeNotify(ByteBuffer, RTMP)

```
public org.red5.server.net.rtmp.event.Notify decodeNotify(org.apache.mina.common.ByteBuffer in,
    RTMP rtmp);
```

**Specified by:** Method decodeNotify in interface IEventDecoder

Decodes notification event.

## 9.17. decodeNotifyOrInvoke(Notify, ByteBuffer, Header, RTMP)

```
protected org.red5.server.net.rtmp.event.Notify decodeNotifyOrInvoke(org.red5.server.net.rtmp.event.Notify notify,
    org.apache.mina.common.ByteBuffer in,
    org.red5.server.net.rtmp.message.Header header,
    RTMP rtmp);
```

Decodes notification event.

### Parameters

notify	Notify event
in	Byte buffer
header	Header
rtmp	RTMP protocol state
<i>return</i>	Notification event

## 9.18. decodePacket(RTMP, ByteBuffer)

```
public org.red5.server.net.rtmp.message.Packet decodePacket(RTMP rtmp,
```

```
org.apache.mina.common.ByteBuffer in);
```

Decodes packet.

Parameters	
rtmp	RTMP protocol state
in	Byte buffer
return	Byte buffer

## 9.19. decodePing(ByteBuffer)

```
public org.red5.server.net.rtmp.event.Ping decodePing(org.apache.mina.common.ByteBuffer in);
```

**Specified by:** Method decodePing in interface IEventDecoder

Decodes ping event.

Parameters	
in	Byte buffer
return	Ping event

## 9.20. decodeSharedObject(ByteBuffer, RTMP)

```
public org.red5.server.so.ISharedObjectMessage decodeSharedObject(org.apache.mina.common.ByteBuffer  
RTMP rtmp);
```

**Specified by:** Method decodeSharedObject in interface IEventDecoder

Decodes shared object message event.

## 9.21. decodeUnknown(byte, ByteBuffer)

```
public org.red5.server.net.rtmp.event.Unknown decodeUnknown(byte dataType,  
org.apache.mina.common.ByteBuffer in);
```

**Specified by:** Method decodeUnknown in interface IEventDecoder

Decodes event of Unknown type.

## 9.22. decodeVideoData(ByteBuffer)

```
public org.red5.server.net.rtmp.event.VideoData decodeVideoData(org.apache.mina.common.ByteBuffer in);
```

**Specified by:** Method decodeVideoData in interface IEventDecoder

Decodes video data event.

## 9.23. doDecodeSharedObject(SharedObjectMessage, ByteBuffer, Input)

```
protected void doDecodeSharedObject(org.red5.server.so.SharedObjectMessage so,  
org.apache.mina.common.ByteBuffer in,
```

```
org.red5.io.object.Input input);
```

Perform the actual decoding of the shared object contents.

#### Parameters

so	
in	
input	

## 9.24. setDeserializer(Deserializer)

```
public void setDeserializer(org.red5.io.object.Deserializer deserializer);
```

Setter for deserializer.

#### Parameters

deserializer	Deserializer
--------------	--------------

## 9.25. setupClassLoader()

```
protected void setupClassLoader();
```

Setup the classloader to use when deserializing custom objects.

# 10. Class RTMPProtocolEncoder

RTMP protocol encoder encodes RTMP messages and packets to byte buffers.

## 10.1. Synopsis

```
public class RTMPProtocolEncoder extends org.?red5.?server.?net.?protocol.?BaseProtocolEncoder
    implements org.?red5.?server.?net.?protocol.?SimpleProtocolEncoder, org.?red5.?server.?net.?rtmp.?

// Protected Fields

protected static org.slf4j.Logger ioLog ;

protected static org.slf4j.Logger log ;

// Public Constructors

public RTMPProtocolEncoder();

// Public Methods

public void doEncodeSharedObject(org.red5.server.so.ISharedObjectMessage so,
    RTMP rtmp,
    org.apache.mina.common.ByteBuffer out);

public org.apache.mina.common.ByteBuffer encode(org.red5.server.net.protocol.ProtocolState state,
    Object message)
    throws Exception;

public org.apache.mina.common.ByteBuffer encodeAudioData(org.red5.server.net.rtmp.event.AudioData
```

## Package org.?red5.?server.?net.?rtmp.?codec

```
public org.apache.mina.common.ByteBuffer encodeBytesRead(org.red5.server.net.rtmp.event.BytesRead

public org.apache.mina.common.ByteBuffer encodeChunkSize(org.red5.server.net.rtmp.event.ChunkSize

public org.apache.mina.common.ByteBuffer encodeFlexMessage(org.red5.server.net.rtmp.event.FlexMessage RTMP rtmp);

public org.apache.mina.common.ByteBuffer encodeFlexSharedObject(org.red5.server.so.ISharedObjectMessage RTMP rtmp);

public org.apache.mina.common.ByteBuffer encodeFlexStreamSend(org.red5.server.net.rtmp.event.FlexStreamSend

public org.apache.mina.common.ByteBuffer encodeHeader(org.red5.server.net.rtmp.message.Header header, org.red5.server.net.rtmp.message.Header lastHeader

public void encodeHeader(org.red5.server.net.rtmp.message.Header header, org.red5.server.net.rtmp.message.Header lastHeader, org.apache.mina.common.ByteBuffer buf);

public org.apache.mina.common.ByteBuffer encodeInvoke(org.red5.server.net.rtmp.event.Invoke invoke, RTMP rtmp);

public org.apache.mina.common.ByteBuffer encodeMessage(RTMP rtmp, org.red5.server.net.rtmp.message.Header header, org.red5.server.net.rtmp.event.IRTMPEvent message);

public org.apache.mina.common.ByteBuffer encodeNotify(org.red5.server.net.rtmp.event.Notify notify, RTMP rtmp);

public org.apache.mina.common.ByteBuffer encodePacket(RTMP rtmp, org.red5.server.net.rtmp.message.Packet packet);

public org.apache.mina.common.ByteBuffer encodePing(org.red5.server.net.rtmp.event.Ping ping);

public org.apache.mina.common.ByteBuffer encodeSharedObject(org.red5.server.so.ISharedObjectMessage RTMP rtmp);

public org.apache.mina.common.ByteBuffer encodeStreamMetadata(org.red5.server.net.rtmp.event.StreamMetadata);

public org.apache.mina.common.ByteBuffer encodeUnknown(org.red5.server.net.rtmp.event.Unknown unknown);

public org.apache.mina.common.ByteBuffer encodeVideoData(org.red5.server.net.rtmp.event.VideoData);

public void setSerializer(org.red5.io.object.Serializer serializer);

// Protected Methods

protected void encodeNotifyOrInvoke(org.apache.mina.common.ByteBuffer out, org.red5.server.net.rtmp.event.Notify invoke, RTMP rtmp);

protected org.apache.mina.common.ByteBuffer encodeNotifyOrInvoke(org.red5.server.net.rtmp.event.Notify invoke, RTMP rtmp);

}
```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?codec.?

RTMPMinaProtocolEncoder , org.?red5.?server.?net.?rtmpt.?codec.?RTMPTProtocolEncoder

**Methods inherited from org.red5.server.net.protocol.BaseProtocolEncoder:**

generateErrorResult

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 10.2. ioLog

```
protected static org.slf4j.Logger ioLog ;
```

I/O operations logger.

## 10.3. log

```
protected static org.slf4j.Logger log ;
```

Logger.

## 10.4. doEncodeSharedObject(ISharedObjectMessage, RTMP, ByteBuffer)

```
public void doEncodeSharedObject(org.red5.server.so.ISharedObjectMessage so,
                                RTMP rtmp,
                                org.apache.mina.common.ByteBuffer out);
```

Perform the actual encoding of the shared object contents.

### Parameters

so	
rtmp	
out	

## 10.5. encode(ProtocolState, Object)

```
public org.apache.mina.common.ByteBuffer encode(org.red5.server.net.protocol.ProtocolState state,
                                                Object message)
                                                throws Exception;
```

**Specified by:** Method encode in interface SimpleProtocolEncoder

Encodes object with given protocol state to byte buffer

## 10.6. encodeAudioData(AudioData)

```
public org.apache.mina.common.ByteBuffer encodeAudioData(org.red5.server.net.rtmp.event.AudioData au
```

**Specified by:** Method encodeAudioData in interface IEventEncoder

Encodes AudioData event to byte buffer.

## 10.7. encodeBytesRead(BytesRead)

```
public org.apache.mina.common.ByteBuffer encodeBytesRead(org.red5.server.net.rtmp.event.BytesRead by
```

**Specified by:** Method encodeBytesRead in interface IEventEncoder

Encodes BytesRead event to byte buffer.

## 10.8. encodeChunkSize(ChunkSize)

```
public org.apache.mina.common.ByteBuffer encodeChunkSize(org.red5.server.net.rtmp.event.ChunkSize ch
```

**Specified by:** Method encodeChunkSize in interface IEventEncoder

Encodes ChunkSize event to byte buffer.

## 10.9. encodeFlexMessage(FlexMessage, RTMP)

```
public org.apache.mina.common.ByteBuffer encodeFlexMessage(org.red5.server.net.rtmp.event.FlexMessage RTMP rtmp);
```

Encodes Flex message event.

### Parameters

msg	Flex message event
return	Encoded data

## 10.10. encodeFlexSharedObject(ISharedObjectMessage, RTMP)

```
public org.apache.mina.common.ByteBuffer encodeFlexSharedObject(org.red5.server.so.ISharedObjectMessage RTMP rtmp);
```

**Specified by:** Method encodeFlexSharedObject in interface IEventEncoder

Encodes SharedObjectMessage event to byte buffer using AMF3 encoding.

## 10.11. encodeHeader(Header, Header)

```
public org.apache.mina.common.ByteBuffer encodeHeader(org.red5.server.net.rtmp.message.Header header, org.red5.server.net.rtmp.message.Header lastHeader);
```

Encode RTMP header.

### Parameters

header	RTMP message header
lastHeader	Previous header
return	Encoded header data

## 10.12. encodeHeader(Header, Header, ByteBuffer)

```
public void encodeHeader(org.red5.server.net.rtmp.message.Header header, org.red5.server.net.rtmp.message.Header lastHeader, org.apache.mina.common.ByteBuffer buf);
```

Encode RTMP header into given ByteBuffer.

**Parameters**

header	RTMP message header
lastHeader	Previous header
buf	Buffer to write encoded header to
return	Encoded header data

**10.13. encodeInvoke(Invoke, RTMP)**

```
public org.apache.mina.common.ByteBuffer encodeInvoke(org.red5.server.net.rtmp.event.Invoke invoke,
                                                    RTMP rtmp);
```

**Specified by:** Method encodeInvoke in interface IEventEncoder

Encodes Invoke event to byte buffer.

**10.14. encodeMessage(RTMP, Header, IRTMPEvent)**

```
public org.apache.mina.common.ByteBuffer encodeMessage(RTMP rtmp,
                                                       org.red5.server.net.rtmp.message.Header header,
                                                       org.red5.server.net.rtmp.event.IRTMPEvent mes
```

Encode message.

**Parameters**

rtmp	RTMP protocol state
header	RTMP message header
message	RTMP message (event)
return	Encoded message data

**10.15. encodeNotify(Notify, RTMP)**

```
public org.apache.mina.common.ByteBuffer encodeNotify(org.red5.server.net.rtmp.event.Notify notify,
                                                    RTMP rtmp);
```

**Specified by:** Method encodeNotify in interface IEventEncoder

Encodes Notify event to byte buffer.

**10.16. encodeNotifyOrInvoke(ByteBuffer, Notify, RTMP)**

```
protected void encodeNotifyOrInvoke(org.apache.mina.common.ByteBuffer out,
                                    org.red5.server.net.rtmp.event.Notify invoke,
                                    RTMP rtmp);
```

Encode notification event and fill given byte buffer.

**Parameters**

out	Byte buffer to fill
-----	---------------------

invoke

Notification event

## 10.17. encodeNotifyOrInvoke(Notify, RTMP)

```
protected org.apache.mina.common.ByteBuffer encodeNotifyOrInvoke(org.red5.server.net.rtmp.event.Notify
RTMP rtmp);
```

Encode notification event.

**Parameters**

invoke	Notification event
<i>return</i>	Encoded event data

## 10.18. encodePacket(RTMP, Packet)

```
public org.apache.mina.common.ByteBuffer encodePacket(RTMP rtmp,
org.red5.server.net.rtmp.message.Packet packet);
```

Encode packet.

**Parameters**

rtmp	RTMP protocol state
packet	RTMP packet
<i>return</i>	Encoded data

## 10.19. encodePing(Ping)

```
public org.apache.mina.common.ByteBuffer encodePing(org.red5.server.net.rtmp.event.Ping ping);
```

**Specified by:** Method encodePing in interface IEventEncoder

Encodes Ping event to byte buffer.

## 10.20. encodeSharedObject(ISharedObjectMessage, RTMP)

```
public org.apache.mina.common.ByteBuffer encodeSharedObject(org.red5.server.so.ISharedObjectMessage
RTMP rtmp);
```

**Specified by:** Method encodeSharedObject in interface IEventEncoder

Encodes SharedObjectMessage event to byte buffer.

## 10.21. encodeUnknown(Unknown)

```
public org.apache.mina.common.ByteBuffer encodeUnknown(org.red5.server.net.rtmp.event.Unknown unknown);
```

**Specified by:** Method encodeUnknown in interface IEventEncoder

Encodes Unknown event to byte buffer.

## 10.22. encodeVideoData(VideoData)

```
public org.apache.mina.common.ByteBuffer encodeVideoData(org.red5.server.net.rtmp.event.VideoData vi
```

**Specified by:** Method encodeVideoData in interface IEventEncoder

Encodes VideoData event to byte buffer.

## 10.23. setSerializer(Serializer)

```
public void setSerializer(org.red5.io.object.Serializer serializer);
```

Setter for serializer.

### Parameters

serializer	Serializer
------------	------------

# 1. Class AllocationDebugger

Simple allocation debugger for Event reference counting.

## 1.1. Synopsis

```
public class AllocationDebugger {  
    // Public Static Methods  
  
    public static AllocationDebugger getInstance();  
  
    // Public Methods  
  
    public synchronized void dump();  
  
    // Protected Methods  
  
    protected synchronized void create(BaseEvent event);  
  
    protected synchronized void release(BaseEvent event);  
  
    protected synchronized void retain(BaseEvent event);  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. create(BaseEvent)

```
protected synchronized void create(BaseEvent event);
```

Add event to map

### Parameters

event	Event
-------	-------

## 1.3. dump()

```
public synchronized void dump();
```

Dumps allocations

## 1.4. getInstance()

```
public static AllocationDebugger getInstance();
```

Getter for instance

### Parameters

return	Allocation debugger instance
--------	------------------------------

## 1.5. release(BaseEvent)

```
protected synchronized void release(BaseEvent event);
```

Release event if there's no more references to it

### Parameters

event	Event
-------	-------

## 1.6. retain(BaseEvent)

```
protected synchronized void retain(BaseEvent event);
```

Retain event

### Parameters

event	Event
-------	-------

## 2. Class AudioData

```
public class AudioData extends, org.red5.server.net.rtmp.event.BaseEvent
    implements, org.red5.server.stream.IStreamData, org.red5.server.api.stream.IStreamPacket
// Protected Fields

    protected org.apache.mina.common.ByteBuffer data ;

// Public Constructors

    public AudioData();

    public AudioData(org.apache.mina.common.ByteBuffer data);

// Public Methods

    public org.apache.mina.common.ByteBuffer getData();

    public byte getDataType();

    public void readExternal(java.io.ObjectInput in)
        throws IOException, ClassNotFoundException;

    public String toString();

    public void writeExternal(java.io.ObjectOutput out)
        throws IOException;

// Protected Methods

    protected void releaseInternal();

}
```

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** `getDataType`, `getHeader`, `getObject`, `getSource`, `getTimestamp`, `getType`, `hasSource`,

```
readExternal , release , releaseInternal , retain , setHeader , setSource ,
setTimestamp , writeExternal
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** header , object ,  
refcount , source , timestamp

## 2.1. AudioData()

```
public AudioData();
```

Constructs a new AudioData.

## 2.2. getData()

```
public org.apache.mina.common.ByteBuffer getData();
```

**Specified by:** Method getData in interface IStreamData

Getter for property 'data'.

## 2.3. getDataType()

```
public byte getDataType();
```

**Specified by:** Method getDataType in interface IStreamPacket

Type of this packet. This is one of the TYPE\_ constants.

## 2.4. releaseInternal()

```
protected void releaseInternal();
```

## 2.5. toString()

```
public String toString();
```

# 3. Class BaseEvent

Base abstract class for all RTMP events

## 3.1. Synopsis

```
public abstract class BaseEvent implements org.?red5.?server.?net.?rtmp.?message.?Constants, org.?red5.?server.?net.?rtmp.?event.?BaseEvent
// Protected Fields
protected org.red5.server.net.rtmp.message.Header header ;
protected Object object ;
```

## Package org.?red5.?server.?net.?rtmp.?event

```
protected int refcount ;  
  
protected org.red5.server.api.event.IEventListener source ;  
  
protected int timestamp ;  
  
// Public Constructors  
  
public BaseEvent();  
  
public BaseEvent(org.red5.server.api.event.IEvent.Type type);  
  
public BaseEvent(org.red5.server.api.event.IEvent.Type type,  
                 org.red5.server.api.event.IEventListener source);  
  
// Public Methods  
  
public abstract byte getDataType();  
  
public org.red5.server.net.rtmp.message.Header getHeader();  
  
public Object getObject();  
  
public org.red5.server.api.event.IEventListener getSource();  
  
public int getTimestamp();  
  
public org.red5.server.api.event.IEvent.Type getType();  
  
public boolean hasSource();  
  
public void readExternal(java.io.ObjectInput in)  
    throws IOException, ClassNotFoundException;  
  
public synchronized void release();  
  
public synchronized void retain();  
  
public void setHeader(org.red5.server.net.rtmp.message.Header header);  
  
public void setSource(org.red5.server.api.event.IEventListener source);  
  
public void setTimestamp(int timestamp);  
  
public void writeExternal(java.io.ObjectOutput out)  
    throws IOException;  
  
// Protected Methods  
  
protected abstract void releaseInternal();  
  
}
```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?event.?AudioData , org.?red5.?server.?net.?rtmp.?event.?BytesRead , org.?red5.?server.?net.?rtmp.?event.?ChunkSize , org.?red5.?server.?net.?rtmp.?event.?ClientBW , org.?red5.?server.?net.?rtmp.?event.?Notify , org.?red5.?server.?net.?rtmp.?event.?Ping , org.?

```
red5.server.net.rtmp.event.ServerBW, org.red5.server.net.rtmp.event.Unknown, org.red5.server.net.rtmp.event.VideoData, org.red5.server.so.SharedObjectMessage
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 3.2. BaseEvent(IEvent.Type)

```
public BaseEvent(org.red5.server.api.event.IEvent.Type type);
```

Create new event of given type

Parameters	
type	Event type

## 3.3. BaseEvent(IEvent.Type, IEventListener)

```
public BaseEvent(org.red5.server.api.event.IEvent.Type type,  
org.red5.server.api.event.IEventListener source);
```

Create new event of given type

Parameters	
type	Event type
source	Event source

## 3.4. header

```
protected org.red5.server.net.rtmp.message.Header header;
```

Event RTMP packet header

## 3.5. object

```
protected Object object;
```

Event target object

## 3.6. refcount

```
protected int refcount;
```

Event references count

## 3.7. source

```
protected org.red5.server.api.event.IEventListener source;
```

Event listener

### 3.8. timestamp

```
protected int timestamp ;
```

Event listener

### 3.9. getDataType()

```
public abstract byte getDataType();
```

**Specified by:** Method getDataType in interface IRTMPEvent

Getter for data type

### 3.10. getHeader()

```
public org.red5.server.net.rtmp.message.Header getHeader();
```

**Specified by:** Method getHeader in interface IRTMPEvent

Getter for header

### 3.11. getObject()

```
public Object getObject();
```

### 3.12. getSource()

```
public org.red5.server.api.event.IEventListener getSource();
```

### 3.13. getTimestamp()

```
public int getTimestamp();
```

**Specified by:** Method getTimestamp in interface IRTMPEvent

Getter for timestamp

### 3.14. getType()

```
public org.red5.server.api.event.IEvent.Type getType();
```

### 3.15. hasSource()

```
public boolean hasSource();
```

### 3.16. release()

```
public synchronized void release();
```

**Specified by:** Method release in interface IRTMPEvent

Hook to free buffers allocated by the event.

### 3.17. releaseInternal()

```
protected abstract void releaseInternal();
```

Rekease event

### 3.18. retain()

```
public synchronized void retain();
```

**Specified by:** Method retain in interface IRTMPEvent

Retain event

### 3.19. setHeader(Header)

```
public void setHeader(org.red5.server.net.rtmp.message.Header header);
```

**Specified by:** Method setHeader in interface IRTMPEvent

Setter for header

### 3.20. setSource(IEventListener)

```
public void setSource(org.red5.server.api.event.IEventListener source);
```

**Specified by:** Method setSource in interface IRTMPEvent

Setter for source

### 3.21. setTimestamp(int)

```
public void setTimestamp(int timestamp);
```

**Specified by:** Method setTimestamp in interface IRTMPEvent

Setter for timestamp

## 4. Class BytesRead

Bytes read event

### 4.1. Synopsis

```
public class BytesRead extends org.red5.server.net.rtmp.event.BaseEvent {
// Public Constructors

    public BytesRead();

    public BytesRead(int bytesRead);

// Public Methods
```

```

public int getBytesRead();

public byte getDataType();

public void readExternal(java.io.ObjectInput in)
    throws IOException, ClassNotFoundException;

public void setBytesRead(int bytesRead);

public String toString();

public void writeExternal(java.io.ObjectOutput out)
    throws IOException;

// Protected Methods

protected void doRelease();

protected void releaseInternal();

}

```

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** `getDataType`, `getHeader`, `getObject`, `getSource`, `getTimestamp`, `getType`, `hasSource`, `readExternal`, `release`, `releaseInternal`, `retain`, `setHeader`, `setSource`, `setTimestamp`, `writeExternal`

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** `header`, `object`, `refcount`, `source`, `timestamp`

## 4.2. BytesRead(int)

```
public BytesRead(int bytesRead);
```

Creates new event with given bytes number

### Parameters

bytesRead	Number of bytes read
-----------	----------------------

## 4.3. doRelease()

```
protected void doRelease();
```

Release event (set bytes read to zero)

## 4.4. getBytesRead()

```
public int getBytesRead();
```

Return number of bytes read

#### Parameters

<i>return</i>	Number of bytes
---------------	-----------------

## 4.5. **getDataType()**

```
public byte getDataType();
```

## 4.6. **releaseInternal()**

```
protected void releaseInternal();
```

## 4.7. **setBytesRead(int)**

```
public void setBytesRead(int bytesRead);
```

Setter for bytes read

#### Parameters

<i>bytesRead</i>	Number of bytes read
------------------	----------------------

## 4.8. **toString()**

```
public String toString();
```

# 5. Class ChunkSize

Chunk size event

## 5.1. Synopsis

```
public class ChunkSize extends, org.?red5.?server.?net.?rtmp.?event.?BaseEvent {
// Public Constructors

    public ChunkSize();

    public ChunkSize(int size);

// Public Methods

    public boolean equals(Object obj);

    public byte getDataType();

    public int getSize();

    public int hashCode();

    public void readExternal(java.io.ObjectInput in)
        throws IOException, ClassNotFoundException;
```

```

    public void setSize(int size);

    public String toString();

    public void writeExternal(java.io.ObjectOutput out)
        throws IOException;

// Protected Methods

    protected void doRelease();

    protected void releaseInternal();

}

```

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** `getDataType`, `getHeader`, `getObject`, `getSource`, `getTimestamp`, `getType`, `hasSource`, `readExternal`, `release`, `releaseInternal`, `retain`, `setHeader`, `setSource`, `setTimestamp`, `writeExternal`

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** `header`, `object`, `refcount`, `source`, `timestamp`

## 5.2. ChunkSize(int)

```
public ChunkSize(int size);
```

Create chunk size event with given size

### Parameters

size	Chunk size
------	------------

## 5.3. doRelease()

```
protected void doRelease();
```

Releases chunk (set size to zero)

## 5.4. equals(Object)

```
public boolean equals(Object obj);
```

## 5.5. getDataType()

```
public byte getDataType();
```

## 5.6. getSize()

```
public int getSize();
```

Getter for size.

#### Parameters

<i>return</i>	Chunk size
---------------	------------

## 5.7. hashCode()

```
public int hashCode();
```

## 5.8. releaseInternal()

```
protected void releaseInternal();
```

## 5.9. setSize(int)

```
public void setSize(int size);
```

Setter for size.

#### Parameters

<i>size</i>	Chunk size
-------------	------------

## 5.10. toString()

```
public String toString();
```

# 6. Class ClientBW

Client bandwidth event

## 6.1. Synopsis

```
public class ClientBW extends org.red5.server.net.rtmp.event.BaseEvent {
// Public Constructors

    public ClientBW();

    public ClientBW(int bandwidth,
                  byte value2);

// Public Methods

    public int getBandwidth();

    public byte getDataType();

    public byte getValue2();

    public void readExternal(java.io.ObjectInput in)
        throws IOException, ClassNotFoundException;

    public void setBandwidth(int bandwidth);
```

```

    public void setValue2(byte value2);

    public String toString();

    public void writeExternal(java.io.ObjectOutput out)
        throws IOException;

// Protected Methods

    protected void releaseInternal();

}

```

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** `getDataType`, `getHeader`, `getObject`, `getSource`, `getTimestamp`, `getType`, `hasSource`, `readExternal`, `release`, `releaseInternal`, `retain`, `setHeader`, `setSource`, `setTimestamp`, `writeExternal`

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** `header`, `object`, `refcount`, `source`, `timestamp`

## 6.2. getBandwidth()

```
public int getBandwidth();
```

Getter for property 'bandwidth'.

### Parameters

<i>return</i>	Value for property 'bandwidth'.
---------------	---------------------------------

## 6.3. getDataType()

```
public byte getDataType();
```

## 6.4. getValue2()

```
public byte getValue2();
```

Getter for value2

### Parameters

<i>return</i>	Value for property 'value2'.
---------------	------------------------------

## 6.5. releaseInternal()

```
protected void releaseInternal();
```

## 6.6. setBandwidth(int)

```
public void setBandwidth(int bandwidth);
```

Setter for bandwidth

### Parameters

bandwidth	New bandwidth
-----------	---------------

## 6.7. setValue2(byte)

```
public void setValue2(byte value2);
```

Setter for property 'value2'.

### Parameters

value2	Value to set for property 'value2'.
--------	-------------------------------------

## 6.8. toString()

```
public String toString();
```

## 7. Class FLVData

```
public class FLVData {
// Public Static Fields

    public static final int AUDIO_ADPCM = 1;

    public static final int AUDIO_MP3 = 2;

    public static final int AUDIO_NELLYMOOSER = 6;

    public static final int AUDIO_NELLYMOOSER_8KHZ = 5;

    public static final int AUDIO_UNCOMPRESSED = 0;

    public static final int FRAMETYPE_DISPOSABLE = 3;

    public static final int FRAMETYPE_INTERFRAME = 2;

    public static final int FRAMETYPE_KEYFRAME = 1;

    public static final int SOUND_RATE_11_KHZ = 2;

    public static final int SOUND_RATE_22_KHZ = 3;

    public static final int SOUND_RATE_44_KHZ = 4;

    public static final int SOUND_RATE_5_5_KHZ = 1;

    public static final int SOUND_SIZE_16_BIT = 2;
```

```

    public static final int SOUND_SIZE_8_BIT = 0;

    public static final int TYPE_AUDIO = 8;

    public static final int TYPE_METADATA = 12;

    public static final int TYPE_VIDEO = 9;

    public static final int VIDEO_ON2_VP6 = 4;

    public static final int VIDEO_SCREEN_VIDEO = 3;

    public static final int VIDEO_SORENSEN_H263 = 2;

// Protected Fields

    protected org.apache.mina.common.ByteBuffer data ;

    protected int timestamp ;

// Public Constructors

    public FLVData();

// Public Methods

    public int getCodec();

    public boolean isDisposable();

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 7.1. AUDIO\_ADPCM

```
public static final int AUDIO_ADPCM = 1;
```

ADPCM data

## 7.2. AUDIO\_MP3

```
public static final int AUDIO_MP3 = 2;
```

MP3 data

## 7.3. AUDIO\_NELLYMOOSER

```
public static final int AUDIO_NELLYMOOSER = 6;
```

Nellymoser encoded data

## 7.4. AUDIO\_NELLYMOOSER\_8KHZ

```
public static final int AUDIO_NELLYMOOSER_8KHZ = 5;
```

Nellymoser 8khz rate data

## 7.5. AUDIO\_UNCOMPRESSED

```
public static final int AUDIO_UNCOMPRESSED = 0;
```

Uncompressed

## 7.6. FRAMETYPE\_DISPOSABLE

```
public static final int FRAMETYPE_DISPOSABLE = 3;
```

Disposable

## 7.7. FRAMETYPE\_INTERFRAME

```
public static final int FRAMETYPE_INTERFRAME = 2;
```

Interframe

## 7.8. FRAMETYPE\_KEYFRAME

```
public static final int FRAMETYPE_KEYFRAME = 1;
```

Keyframe

## 7.9. SOUND\_RATE\_11\_KHZ

```
public static final int SOUND_RATE_11_KHZ = 2;
```

Sound size when 11 khz rate marker

## 7.10. SOUND\_RATE\_22\_KHZ

```
public static final int SOUND_RATE_22_KHZ = 3;
```

Sound size when 22 khz rate marker

## 7.11. SOUND\_RATE\_44\_KHZ

```
public static final int SOUND_RATE_44_KHZ = 4;
```

Sound size when 44 khz rate marker

## 7.12. SOUND\_RATE\_5\_5\_KHZ

```
public static final int SOUND_RATE_5_5_KHZ = 1;
```

Sound size when 5.5 khz rate marker

## 7.13. SOUND\_SIZE\_16\_BIT

```
public static final int SOUND_SIZE_16_BIT = 2;
```

Sound size when 16 khz quality marker

## 7.14. SOUND\_SIZE\_8\_BIT

```
public static final int SOUND_SIZE_8_BIT = 0;
```

Sound size when 8 khz quality marker

## 7.15. TYPE\_AUDIO

```
public static final int TYPE_AUDIO = 8;
```

Audio data

## 7.16. TYPE\_METADATA

```
public static final int TYPE_METADATA = 12;
```

Metadata

## 7.17. TYPE\_VIDEO

```
public static final int TYPE_VIDEO = 9;
```

Video data

## 7.18. VIDEO\_ON2\_VP6

```
public static final int VIDEO_ON2_VP6 = 4;
```

ON2 VP6 codec marker

## 7.19. VIDEO\_SCREEN\_VIDEO

```
public static final int VIDEO_SCREEN_VIDEO = 3;
```

Screen video

## 7.20. VIDEO\_SORENSEN\_H263

```
public static final int VIDEO_SORENSEN_H263 = 2;
```

Sorenson H263 codec marker

## 7.21. getCodec()

```
public int getCodec();
```

Getter for codec. Returns 0 by now.

Parameters

**return****Codec**

## 7.22. **isDisposable()**

```
public boolean isDisposable();
```

Getter for disposable state

### Parameters

<b>return</b>	<code>true</code> if FVL data is disposable, <code>false</code> otherwise
---------------	---

# 8. Class FlexMessage

Flex method invocation. To be implemented.

## 8.1. Synopsis

```
public class FlexMessage extends org.red5.server.net.rtmp.event.Invoke {
    // Public Constructors

    public FlexMessage();

    // Public Methods

    public byte getDataType();

}
```

**Methods inherited from org.red5.server.net.rtmp.event.Invoke:** equals , getCall ,  
getDataType , toString

**Methods inherited from org.red5.server.net.rtmp.event.Notify:** doRelease ,  
getConnectionParams , getData , getInvokeId , readExternal , releaseInternal ,  
setCall , setConnectionParams , setData , setInvokeId , writeExternal

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** getHeader ,  
getObject , getSource , getTimestamp , getType , hasSource , release , retain ,  
setHeader , setSource , setTimestamp

**Methods inherited from java.lang.Object:** clone , finalize , getClass , hashCode , notify  
, notifyAll , wait

**Fields inherited from org.red5.server.net.rtmp.event.Notify:** call , data

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** header , object ,  
refcount , source , timestamp

## 8.2. **getDataType()**

```
public byte getDataType();
```

## 9. Class FlexStreamSend

AMF3 stream send message.

### 9.1. Synopsis

```
public class FlexStreamSend extends org.red5.server.net.rtmp.event.Notify {
// Public Constructors

    public FlexStreamSend();

    public FlexStreamSend(org.apache.mina.common.ByteBuffer data);

// Public Methods

    public byte getDataType();

}
```

**Methods inherited from org.red5.server.net.rtmp.event.Notify:** doRelease , equals , getCall , getConnectionParams , getData , getDataType , getInvokeId , readExternal , releaseInternal , setCall , setConnectionParams , setData , setInvokeId , toString , writeExternal

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** getHeader , getObject , getSource , getTimestamp , getType , hasSource , release , retain , setHeader , setSource , setTimestamp

**Methods inherited from java.lang.Object:** clone , finalize , getClass , hashCode , notify , notifyAll , wait

**Fields inherited from org.red5.server.net.rtmp.event.Notify:** call , data

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** header , object , refcount , source , timestamp

### 9.2. FlexStreamSend(ByteBuffer)

```
public FlexStreamSend(org.apache.mina.common.ByteBuffer data);
```

Create new stream send object.

#### Parameters

data
------

### 9.3. getDataType()

```
public byte getDataType();
```

## 10. Interface IRTMPEvent

```
public interface IRTMPEvent extends org.red5.server.api.event.IEvent {
```

```
// Public Methods

    public byte getDataType();

    public org.red5.server.net.rtmp.message.Header getHeader();

    public int getTimestamp();

    public void release();

    public void retain();

    public void setHeader(org.red5.server.net.rtmp.message.Header header);

    public void setSource(org.red5.server.api.event.IEventListener source);

    public void setTimestamp(int timestamp);

}
```

## 10.1. **getDataType()**

```
public byte getDataType();
```

Getter for data type

**Parameters**

<i>return</i>	Data type
---------------	-----------

## 10.2. **getHeader()**

```
public org.red5.server.net.rtmp.message.Header getHeader();
```

Getter for header

**Parameters**

<i>return</i>	RTMP packet header
---------------	--------------------

## 10.3. **getTimestamp()**

```
public int getTimestamp();
```

Getter for timestamp

**Parameters**

<i>return</i>	Event timestamp
---------------	-----------------

## 10.4. **release()**

```
public void release();
```

Hook to free buffers allocated by the event.

## 10.5. retain()

```
public void retain();
```

Retain event

## 10.6. setHeader(Header)

```
public void setHeader(org.red5.server.net.rtmp.message.Header header);
```

Setter for header

### Parameters

header	RTMP packet header
--------	--------------------

## 10.7. setSource(IEventListener)

```
public void setSource(org.red5.server.api.event.IEventListener source);
```

Setter for source

### Parameters

source	Source
--------	--------

## 10.8. setTimestamp(int)

```
public void setTimestamp(int timestamp);
```

Setter for timestamp

### Parameters

timestamp	New event timestamp
-----------	---------------------

# 11. Class Invoke

Remote invocation event

## 11.1. Synopsis

```
public class Invoke extends, org.?red5.?server.?net.?rtmp.?event.?Notify {
// Public Constructors

    public Invoke();

    public Invoke(org.apache.mina.common.ByteBuffer data);

    public Invoke(org.red5.server.api.service.IPendingServiceCall call);
```

```
// Public Methods

    public boolean equals(Object obj);

    public org.red5.server.api.service.IPendingServiceCall getCall();

    public byte getDataType();

    public String toString();

}
```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?event.?FlexMessage

**Methods inherited from org.red5.server.net.rtmp.event.Notify:** doRelease , equals , getCall , getConnectionParams , getData , getDataType , getInvokeId , readExternal , releaseInternal , setCall , setConnectionParams , setData , setInvokeId , toString , writeExternal

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** getHeader , getObject , getSource , getTimestamp , getType , hasSource , release , retain , setHeader , setSource , setTimestamp

**Methods inherited from java.lang.Object:** clone , finalize , getClass , hashCode , notify , notifyAll , wait

**Fields inherited from org.red5.server.net.rtmp.event.Notify:** call , data

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** header , object , refcount , source , timestamp

## 11.2. Invoke()

```
public Invoke();
```

Constructs a new Invoke.

## 11.3. Invoke(ByteBuffer)

```
public Invoke(org.apache.mina.common.ByteBuffer data);
```

Create new invocation event with given data

### Parameters

data	Event data
------	------------

## 11.4. Invoke(IPendingServiceCall)

```
public Invoke(org.red5.server.api.service.IPendingServiceCall call);
```

Create new invocation event with given pending service call

**Parameters**

call	Pending call
------	--------------

**11.5. equals(Object)**

```
public boolean equals(Object obj);
```

**11.6. getCall()**

```
public org.red5.server.api.service.IPendingServiceCall getCall();
```

**11.7. getDataType()**

```
public byte getDataType();
```

**11.8. toString()**

```
public String toString();
```

**12. Class Notify**

Stream notification event

**12.1. Synopsis**

```
public class Notify extends org.red5.server.net.rtmp.event.BaseEvent
    implements org.red5.server.stream.IStreamData, org.red5.server.api.stream.IStreamPacket
// Protected Fields

    protected org.red5.server.api.service.IServiceCall call;

    protected org.apache.mina.common.ByteBuffer data;

// Public Constructors

    public Notify();

    public Notify(org.apache.mina.common.ByteBuffer data);

    public Notify(org.red5.server.api.service.IServiceCall call);

// Public Methods

    public boolean equals(Object obj);

    public org.red5.server.api.service.IServiceCall getCall();

    public java.util.Map getConnectionParams();

    public org.apache.mina.common.ByteBuffer getData();

    public byte getDataType();

    public int getInvokeId();
```

```

public void readExternal(java.io.ObjectInput in)
    throws IOException, ClassNotFoundException;

public void setCall(org.red5.server.api.service.IServiceCall call);

public void setConnectionParams(java.util.Map connectionParams);

public void setData(org.apache.mina.common.ByteBuffer data);

public void setInvokeId(int invokeId);

public String toString();

public void writeExternal(java.io.ObjectOutput out)
    throws IOException;

// Protected Methods

protected void doRelease();

protected void releaseInternal();

}

```

**Direct known subclasses:** org.red5.server.net.rtmp.event.FlexStreamSend , org.red5.server.net.rtmp.event.Invoke

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** getDataType , getHeader , getObject , getSource , getTimestamp , getType , hasSource , readExternal , release , releaseInternal , retain , setHeader , setSource , setTimestamp , writeExternal

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** header , object , refcount , source , timestamp

## 12.2. Notify()

```
public Notify();
```

Constructs a new Notify.

## 12.3. Notify(ByteBuffer)

```
public Notify(org.apache.mina.common.ByteBuffer data);
```

Create new notification event with given byte buffer

### Parameters

data	Byte buffer
------	-------------

## 12.4. Notify(IServiceCall)

```
public Notify(org.red5.server.api.service.IServiceCall call);
```

Create new notification event with given service call

### Parameters

call	Service call
------	--------------

## 12.5. call

```
protected org.red5.server.api.service.IServiceCall call ;
```

Service call

## 12.6. data

```
protected org.apache.mina.common.ByteBuffer data ;
```

Event data

## 12.7. doRelease()

```
protected void doRelease();
```

Release event (nullify call object)

## 12.8. equals(Object)

```
public boolean equals(Object obj);
```

## 12.9. getCall()

```
public org.red5.server.api.service.IServiceCall getCall();
```

Getter for service call

### Parameters

return	Service call
--------	--------------

## 12.10. getConnectionParams()

```
public java.util.Map getConnectionParams();
```

Getter for connection parameters

### Parameters

return	Connection parameters
--------	-----------------------

## 12.11. getData()

```
public org.apache.mina.common.ByteBuffer getData();
```

**Specified by:** Method getData in interface IStreamData

Getter for property 'data'.

## 12.12. **getDataType()**

```
public byte getDataType();
```

**Specified by:** Method getDataType in interface IStreamPacketType of this packet. This is one of the `TYPE_` constants.

## 12.13. **getInvokeId()**

```
public int getInvokeId();
```

Getter for invoke id

**Parameters**

<i>return</i>	Invoke id
---------------	-----------

## 12.14. **releaseInternal()**

```
protected void releaseInternal();
```

## 12.15. **setCall(IServiceCall)**

```
public void setCall(org.red5.server.api.service(IServiceCall call);
```

Setter for call

**Parameters**

<i>call</i>	Service call
-------------	--------------

## 12.16. **setConnectionParams(Map)**

```
public void setConnectionParams(java.util.Map connectionParams);
```

Setter for connection parameters

**Parameters**

<i>connectionParams</i>	Connection parameters
-------------------------	-----------------------

## 12.17. **setData(ByteBuffer)**

```
public void setData(org.apache.mina.common.ByteBuffer data);
```

Setter for data

**Parameters**

data	Data
------	------

## 12.18. setInvokeld(int)

```
public void setInvokeId(int invokeId);
```

Setter for invoke id

### Parameters

invokeld	Invoke id
----------	-----------

## 12.19. toString()

```
public String toString();
```

# 13. Class Ping

Ping event, actually combination of different events

## 13.1. Synopsis

```
public class Ping extends org.?red5.?server.?net.?rtmp.?event.?BaseEvent {  
    // Public Static Fields
```

```
        public static final short CLIENT_BUFFER = 3;
```

```
        public static final short PING_CLIENT = 6;
```

```
        public static final short PONG_SERVER = 7;
```

```
        public static final short STREAM_CLEAR = 0;
```

```
        public static final short STREAM_PLAYBUFFER_CLEAR = 1;
```

```
        public static final short STREAM_RESET = 4;
```

```
        public static final int UNDEFINED = -1;
```

```
        public static final short UNKNOWN_2 = 2;
```

```
        public static final short UNKNOWN_5 = 5;
```

```
        public static final short UNKNOWN_8 = 8;
```

```
    // Public Constructors
```

```
        public Ping();
```

```
        public Ping(short value1,  
                  int value2);
```

```
        public Ping(short value1,  
                  int value2,  
                  int value3);
```

```

public Ping(short value1,
           int value2,
           int value3,
           int value4);

// Public Methods

public byte getDataType();

public String getDebug();

public short getValue1();

public int getValue2();

public int getValue3();

public int getValue4();

public void readExternal(java.io.ObjectInput in)
    throws IOException, ClassNotFoundException;

public void setDebug(String debug);

public void setValue1(short value1);

public void setValue2(int value2);

public void setValue3(int value3);

public void setValue4(int value4);

public String toString();

public void writeExternal(java.io.ObjectOutput out)
    throws IOException;

// Protected Methods

protected void doRelease();

protected void releaseInternal();

}

```

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** `getDataType`, `getHeader`, `getObject`, `getSource`, `getTimestamp`, `getType`, `hasSource`, `readExternal`, `release`, `releaseInternal`, `retain`, `setHeader`, `setSource`, `setTimestamp`, `writeExternal`

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** `header`, `object`, `refcount`, `source`, `timestamp`

## 13.2. Ping()

```
public Ping();
```

Constructs a new Ping.

## 13.3. CLIENT\_BUFFER

```
public static final short CLIENT_BUFFER = 3;
```

Client buffer

## 13.4. PING\_CLIENT

```
public static final short PING_CLIENT = 6;
```

Client ping event

## 13.5. PONG\_SERVER

```
public static final short PONG_SERVER = 7;
```

Server response event

## 13.6. STREAM\_CLEAR

```
public static final short STREAM_CLEAR = 0;
```

Stream clear event

## 13.7. STREAM\_PLAYBUFFER\_CLEAR

```
public static final short STREAM_PLAYBUFFER_CLEAR = 1;
```

Stream play

## 13.8. STREAM\_RESET

```
public static final short STREAM_RESET = 4;
```

Stream reset

## 13.9. UNDEFINED

```
public static final int UNDEFINED = -1;
```

Event type is undefined

## 13.10. UNKNOWN\_2

```
public static final short UNKNOWN_2 = 2;
```

Unknown event

## 13.11. UNKNOWN\_5

```
public static final short UNKNOWN_5 = 5;
```

One more unknown event

## 13.12. UNKNOWN\_8

```
public static final short UNKNOWN_8 = 8;
```

One more unknown event

## 13.13. getDataType()

```
public byte getDataType();
```

## 13.14. getDebug()

```
public String getDebug();
```

Getter for property 'debug'.

### Parameters

<i>return</i>	Value for property 'debug'.
---------------	-----------------------------

## 13.15. getValue1()

```
public short getValue1();
```

Getter for property 'value1'.

### Parameters

<i>return</i>	Value for property 'value1'.
---------------	------------------------------

## 13.16. getValue2()

```
public int getValue2();
```

Getter for property 'value2'.

### Parameters

<i>return</i>	Value for property 'value2'.
---------------	------------------------------

## 13.17. getValue3()

```
public int getValue3();
```

Getter for property 'value3'.

### Parameters

<i>return</i>	Value for property 'value3'.
---------------	------------------------------

### 13.18. getValue4()

```
public int getValue4();
```

Getter for property 'value4'.

#### Parameters

<i>return</i>	Value for property 'value4'.
---------------	------------------------------

### 13.19. releaseInternal()

```
protected void releaseInternal();
```

### 13.20. setDebug(String)

```
public void setDebug(String debug);
```

Setter for property 'debug'.

#### Parameters

debug	Value to set for property 'debug'.
-------	------------------------------------

### 13.21. setValue1(short)

```
public void setValue1(short value1);
```

Setter for property 'value1'.

#### Parameters

value1	Value to set for property 'value1'.
--------	-------------------------------------

### 13.22. setValue2(int)

```
public void setValue2(int value2);
```

Setter for property 'value2'.

#### Parameters

value2	Value to set for property 'value2'.
--------	-------------------------------------

### 13.23. setValue3(int)

```
public void setValue3(int value3);
```

Setter for property 'value3'.

#### Parameters

value3	Value to set for property 'value3'.
--------	-------------------------------------

## 13.24. setValue4(int)

```
public void setValue4(int value4);
```

Setter for property 'value4'.

### Parameters

value4	Value to set for property 'value4'.
--------	-------------------------------------

## 13.25. toString()

```
public String toString();
```

# 14. Class SerializeUtils

The utility class provides conversion methods to ease the use of byte arrays, mina bytebuffers, and nio bytebuffers.

## 14.1. Synopsis

```
public class SerializeUtils {
// Public Constructors

    public SerializeUtils();

// Public Static Methods

    public static void ByteArrayToByteBuffer(byte[] byteBuf,
                                             org.apache.mina.common.ByteBuffer buf);

    public static void ByteArrayToNioByteBuffer(byte[] byteBuf,
                                                java.nio.ByteBuffer buf);

    public static byte[] ByteBufferToByteArray(org.apache.mina.common.ByteBuffer buf);

    public static byte[] NioByteBufferToByteArray(java.nio.ByteBuffer buf);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

# 15. Class ServerBW

Server bandwidth event

## 15.1. Synopsis

```
public class ServerBW extends, org.?red5.?server.?net.?rtmp.?event.?BaseEvent {
// Public Constructors
```

```

public ServerBW();

public ServerBW(int bandwidth);

// Public Methods

public int getBandwidth();

public byte getDataType();

public void readExternal(java.io.ObjectInput in)
    throws IOException, ClassNotFoundException;

public void setBandwidth(int bandwidth);

public String toString();

public void writeExternal(java.io.ObjectOutput out)
    throws IOException;

// Protected Methods

protected void releaseInternal();

}

```

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** `getDataType`, `getHeader`, `getObject`, `getSource`, `getTimestamp`, `getType`, `hasSource`, `readExternal`, `release`, `releaseInternal`, `retain`, `setHeader`, `setSource`, `setTimestamp`, `writeExternal`

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** `header`, `object`, `refcount`, `source`, `timestamp`

## 15.2. ServerBW(int)

```
public ServerBW(int bandwidth);
```

Server bandwidth event

### Parameters

bandwidth	Bandwidth
-----------	-----------

## 15.3. getBandwidth()

```
public int getBandwidth();
```

Getter for bandwidth

### Parameters

<i>return</i>	Bandwidth
---------------	-----------

## 15.4. **getDataType()**

```
public byte getDataType();
```

## 15.5. **releaseInternal()**

```
protected void releaseInternal();
```

## 15.6. **setBandwidth(int)**

```
public void setBandwidth(int bandwidth);
```

Setter for bandwidth

### Parameters

bandwidth	New bandwidth.
-----------	----------------

## 15.7. **toString()**

```
public String toString();
```

# 16. Class Unknown

Unknown event

## 16.1. Synopsis

```
public class Unknown extends, org.?red5.?server.?net.?rtmp.?event.?BaseEvent {  
    // Protected Fields
```

```
        protected org.apache.mina.common.ByteBuffer data ;
```

```
        protected byte dataType ;
```

```
    // Public Constructors
```

```
        public Unknown();
```

```
        public Unknown(byte dataType,  
                      org.apache.mina.common.ByteBuffer data);
```

```
    // Public Methods
```

```
        public org.apache.mina.common.ByteBuffer getData();
```

```
        public byte getDataType();
```

```
        public void readExternal(java.io.ObjectInput in)  
            throws IOException, ClassNotFoundException;
```

```
        public String toString();
```

```

public void writeExternal(java.io.ObjectOutput out)
    throws IOException;

// Protected Methods

protected void releaseInternal();

}

```

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** `getDataType`, `getHeader`, `getObject`, `getSource`, `getTimestamp`, `getType`, `hasSource`, `readExternal`, `release`, `releaseInternal`, `retain`, `setHeader`, `setSource`, `setTimestamp`, `writeExternal`

**Methods inherited from java.lang.Object:** `clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** `header`, `object`, `refcount`, `source`, `timestamp`

## 16.2. Unknown(byte, ByteBuffer)

```

public Unknown(byte dataType,
               org.apache.mina.common.ByteBuffer data);

```

Create new unknown event with given data and data type

### Parameters

dataType	Data type
data	Event data

## 16.3. data

```

protected org.apache.mina.common.ByteBuffer data ;

```

Event data

## 16.4. dataType

```

protected byte dataType ;

```

Type of data

## 16.5. getData()

```

public org.apache.mina.common.ByteBuffer getData();

```

Getter for data

### Parameters

`return``Data`

## 16.6. `getDataType()`

```
public byte getDataType();
```

## 16.7. `releaseInternal()`

```
protected void releaseInternal();
```

## 16.8. `toString()`

```
public String toString();
```

# 17. Class VideoData

Video data event

## 17.1. Synopsis

```
public class VideoData extends, org.?red5.?server.?net.?rtmp.?event.?BaseEvent
    implements, org.?red5.?io.?IoConstants, org.?red5.?server.?stream.?IStreamData, org.?red5.?server.?

// Protected Fields

    protected org.apache.mina.common.ByteBuffer data ;

// Public Constructors

    public VideoData();
    public VideoData(org.apache.mina.common.ByteBuffer data);

// Public Methods

    public org.apache.mina.common.ByteBuffer getData();
    public byte getDataType();
    public org.red5.server.net.rtmp.event.VideoData.FrameType getFrameType();
    public void readExternal(java.io.ObjectInput in)
        throws IOException, ClassNotFoundException;
    public String toString();
    public void writeExternal(java.io.ObjectOutput out)
        throws IOException;
// Protected Methods
    protected void releaseInternal();
}
```

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** `getDataType`, `getHeader`, `getObject`, `getSource`, `getTimestamp`, `getType`, `hasSource`,

```
readExternal , release , releaseInternal , retain , setHeader , setSource ,
setTimestamp , writeExternal
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** header , object , refcount , source , timestamp

## 17.2. VideoData()

```
public VideoData();
```

Constructs a new VideoData.

## 17.3. VideoData(ByteBuffer)

```
public VideoData(org.apache.mina.common.ByteBuffer data);
```

Create video data event with given data buffer

### Parameters

data	Video data
------	------------

## 17.4. data

```
protected org.apache.mina.common.ByteBuffer data ;
```

Video data

## 17.5. getData()

```
public org.apache.mina.common.ByteBuffer getData();
```

**Specified by:** Method getData in interface IStreamData

Getter for property 'data'.

## 17.6. getDataType()

```
public byte getDataType();
```

**Specified by:** Method getDataType in interface IStreamPacket

Type of this packet. This is one of the TYPE\_ constants.

## 17.7. getFrameType()

```
public org.red5.server.net.rtmp.event.VideoData.FrameType getFrameType();
```

Getter for frame type

**Parameters**

<i>return</i>	Type of video frame
---------------	---------------------

**17.8. releaseInternal()**

```
protected void releaseInternal();
```

**17.9. toString()**

```
public String toString();
```

**18. Class VideoData.FrameType**

Videoframe type

**18.1. Synopsis**

```
public static final class VideoData.FrameType extends java.lang.Enum {
// Public Static Fields

    public static final org.red5.server.net.rtmp.event.VideoData.FrameType DISPOSABLE_INTERFRAME ;

    public static final org.red5.server.net.rtmp.event.VideoData.FrameType INTERFRAME ;

    public static final org.red5.server.net.rtmp.event.VideoData.FrameType KEYFRAME ;

    public static final org.red5.server.net.rtmp.event.VideoData.FrameType UNKNOWN ;

// Public Static Methods

    public static org.red5.server.net.rtmp.event.VideoData.FrameType valueOf(String name);

    public static org.red5.server.net.rtmp.event.VideoData.FrameType[] values();

}
```

**Methods inherited from java.lang.Enum:** clone, compareTo, equals, finalize, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

**Methods inherited from java.lang.Object:** getClass, notify, notifyAll, wait

# 1. Interface Constants

Class for AMF and RTMP marker values constants

## 1.1. Synopsis

```
public interface Constants {  
    // Public Static Fields  
  
    public static final String ACTION_CLOSE_STREAM = "closeStream";  
  
    public static final String ACTION_CONNECT = "connect";  
  
    public static final String ACTION_CREATE_STREAM = "createStream";  
  
    public static final String ACTION_DELETE_STREAM = "deleteStream";  
  
    public static final String ACTION_DISCONNECT = "disconnect";  
  
    public static final String ACTION_PAUSE = "pause";  
  
    public static final String ACTION_PLAY = "play";  
  
    public static final String ACTION_PUBLISH = "publish";  
  
    public static final String ACTION_RECEIVE_AUDIO = "receiveAudio";  
  
    public static final String ACTION_RECEIVE_VIDEO = "receiveVideo";  
  
    public static final String ACTION_RELEASE_STREAM = "releaseStream";  
  
    public static final String ACTION_SEEK = "seek";  
  
    public static final String ACTION_STOP = "stop";  
  
    public static final int HANDSHAKE_SIZE = 1536;  
  
    public static final byte HEADER_CONTINUE = 3;  
  
    public static final byte HEADER_NEW = 0;  
  
    public static final byte HEADER_SAME_SOURCE = 1;  
  
    public static final byte HEADER_TIMER_CHANGE = 2;  
  
    public static final int MEDIUM_INT_MAX = 16777215;  
  
    public static final byte SO_CLIENT_CLEAR_DATA = 8;  
  
    public static final byte SO_CLIENT_DELETE_DATA = 9;  
  
    public static final byte SO_CLIENT_INITIAL_DATA = 11;  
  
    public static final byte SO_CLIENT_SEND_MESSAGE = 6;
```

```

public static final byte SO_CLIENT_STATUS = 7;

public static final byte SO_CLIENT_UPDATE_ATTRIBUTE = 5;

public static final byte SO_CLIENT_UPDATE_DATA = 4;

public static final byte SO_CONNECT = 1;

public static final byte SO_DELETE_ATTRIBUTE = 10;

public static final byte SO_DISCONNECT = 2;

public static final byte SO_SEND_MESSAGE = 6;

public static final byte SO_SET_ATTRIBUTE = 3;

public static final byte TYPE_AUDIO_DATA = 8;

public static final byte TYPE_BYTES_READ = 3;

public static final byte TYPE_CHUNK_SIZE = 1;

public static final byte TYPE_CLIENT_BANDWIDTH = 6;

public static final byte TYPE_FLEX_MESSAGE = 17;

public static final byte TYPE_FLEX_SHARED_OBJECT = 16;

public static final byte TYPE_FLEX_STREAM_SEND = 15;

public static final byte TYPE_INVOKE = 20;

public static final byte TYPE_NOTIFY = 18;

public static final byte TYPE_PING = 4;

public static final byte TYPE_SERVER_BANDWIDTH = 5;

public static final byte TYPE_SHARED_OBJECT = 19;

public static final byte TYPE_STREAM_METADATA = 18;

public static final byte TYPE_VIDEO_DATA = 9;
}

```

## 1.2. HANDSHAKE\_SIZE

```
public static final int HANDSHAKE_SIZE = 1536;
```

Size of initial handshake between client and server

## 1.3. HEADER\_CONTINUE

```
public static final byte HEADER_CONTINUE = 3;
```

There's more to encode

## 1.4. HEADER\_NEW

```
public static final byte HEADER_NEW = 0;
```

New header marker

## 1.5. HEADER\_SAME\_SOURCE

```
public static final byte HEADER_SAME_SOURCE = 1;
```

Same source marker

## 1.6. HEADER\_TIMER\_CHANGE

```
public static final byte HEADER_TIMER_CHANGE = 2;
```

Timer change marker

## 1.7. MEDIUM\_INT\_MAX

```
public static final int MEDIUM_INT_MAX = 16777215;
```

Medium integer max value

## 1.8. SO\_CLIENT\_CLEAR\_DATA

```
public static final byte SO_CLIENT_CLEAR_DATA = 8;
```

Clear event for Shared Object

## 1.9. SO\_CLIENT\_DELETE\_DATA

```
public static final byte SO_CLIENT_DELETE_DATA = 9;
```

Delete data for Shared Object

## 1.10. SO\_CLIENT\_INITIAL\_DATA

```
public static final byte SO_CLIENT_INITIAL_DATA = 11;
```

Initial SO data flag

## 1.11. SO\_CLIENT\_SEND\_MESSAGE

```
public static final byte SO_CLIENT_SEND_MESSAGE = 6;
```

Send SO message flag

## 1.12. SO\_CLIENT\_STATUS

```
public static final byte SO_CLIENT_STATUS = 7;
```

Shared Object status marker

## 1.13. SO\_CLIENT\_UPDATE\_ATTRIBUTE

```
public static final byte SO_CLIENT_UPDATE_ATTRIBUTE = 5;
```

Client Shared Object attribute update

## 1.14. SO\_CLIENT\_UPDATE\_DATA

```
public static final byte SO_CLIENT_UPDATE_DATA = 4;
```

Client Shared Object data update

## 1.15. SO\_CONNECT

```
public static final byte SO_CONNECT = 1;
```

Shared Object connection

## 1.16. SO\_DELETE\_ATTRIBUTE

```
public static final byte SO_DELETE_ATTRIBUTE = 10;
```

Shared Object attribute deletion flag

## 1.17. SO\_DISCONNECT

```
public static final byte SO_DISCONNECT = 2;
```

Shared Object disconnection

## 1.18. SO\_SEND\_MESSAGE

```
public static final byte SO_SEND_MESSAGE = 6;
```

Send message flag

## 1.19. SO\_SET\_ATTRIBUTE

```
public static final byte SO_SET_ATTRIBUTE = 3;
```

Set Shared Object attribute flag

## 1.20. TYPE\_AUDIO\_DATA

```
public static final byte TYPE_AUDIO_DATA = 8;
```

Audio data marker

## 1.21. TYPE\_BYTES\_READ

```
public static final byte TYPE_BYTES_READ = 3;
```

Send every x bytes read by both sides

## 1.22. TYPE\_CHUNK\_SIZE

```
public static final byte TYPE_CHUNK_SIZE = 1;
```

RTMP chunk size constant

## 1.23. TYPE\_CLIENT\_BANDWIDTH

```
public static final byte TYPE_CLIENT_BANDWIDTH = 6;
```

Client (upstream) bandwidth marker

## 1.24. TYPE\_FLEX\_MESSAGE

```
public static final byte TYPE_FLEX_MESSAGE = 17;
```

AMF3 message

## 1.25. TYPE\_FLEX\_SHARED\_OBJECT

```
public static final byte TYPE_FLEX_SHARED_OBJECT = 16;
```

AMF3 shared object

## 1.26. TYPE\_FLEX\_STREAM\_SEND

```
public static final byte TYPE_FLEX_STREAM_SEND = 15;
```

AMF3 stream send

## 1.27. TYPE\_INVOKE

```
public static final byte TYPE_INVOKE = 20;
```

Invoke operation (remoting call but also used for streaming) marker

## 1.28. TYPE\_NOTIFY

```
public static final byte TYPE_NOTIFY = 18;
```

Notification is invocation without response

## 1.29. TYPE\_PING

```
public static final byte TYPE_PING = 4;
```

Ping is a stream control message, has subtypes

## 1.30. TYPE\_SERVER\_BANDWIDTH

```
public static final byte TYPE_SERVER_BANDWIDTH = 5;
```

Server (downstream) bandwidth marker

## 1.31. TYPE\_SHARED\_OBJECT

```
public static final byte TYPE_SHARED_OBJECT = 19;
```

Shared Object marker

## 1.32. TYPE\_STREAM\_METADATA

```
public static final byte TYPE_STREAM_METADATA = 18;
```

Stream metadata

## 1.33. TYPE\_VIDEO\_DATA

```
public static final byte TYPE_VIDEO_DATA = 9;
```

Video data marker

## 2. Class Header

RTMP packet header

### 2.1. Synopsis

```
public class Header implements, org.red5.server.net.rtmp.message.Constants, java.io.Externalizable
// Public Constructors

    public Header();

// Public Methods

    public Object clone();

    public boolean equals(Object other);

    public int getChannelId();

    public byte getDataType();

    public int getSize();

    public int getStreamId();

    public int getTimer();

    public boolean isTimerRelative();

    public void readExternal(java.io.ObjectInput in)
        throws IOException, ClassNotFoundException;

    public void setChannelId(int channelId);
```

```

    public void setDataType(byte dataType);

    public void setSize(int size);

    public void setStreamId(int streamId);

    public void setTimer(int timer);

    public void setTimerRelative(boolean timerRelative);

    public String toString();

    public void writeExternal(java.io.ObjectOutput out)
        throws IOException;
}

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. clone()

```
public Object clone();
```

## 2.3. equals(Object)

```
public boolean equals(Object other);
```

## 2.4. getChannelId()

```
public int getChannelId();
```

Getter for channel id

### Parameters

return	Channel id
--------	------------

## 2.5. getDataType()

```
public byte getDataType();
```

Getter for data type

### Parameters

return	Data type
--------	-----------

## 2.6. getSize()

```
public int getSize();
```

Getter for size.

Parameters

<i>return</i>	Header size
---------------	-------------

## 2.7. getStreamId()

```
public int getStreamId();
```

Getter for stream id

Parameters

<i>return</i>	Stream id
---------------	-----------

## 2.8. getTimer()

```
public int getTimer();
```

Getter for timer

Parameters

<i>return</i>	Timer
---------------	-------

## 2.9. isTimerRelative()

```
public boolean isTimerRelative();
```

Getter for timer relative flag

Parameters

<i>return</i>	true if timer value is relative, false otherwise
---------------	--

## 2.10. setChannelId(int)

```
public void setChannelId(int channelId);
```

Setter for channel id

Parameters

channelId	Header channel id
-----------	-------------------

## 2.11. setDataType(byte)

```
public void setDataType(byte dataType);
```

Setter for data type

Parameters

dataType	Data type
----------	-----------

## 2.12. setSize(int)

```
public void setSize(int size);
```

Setter for size

Parameters

size	Header size
------	-------------

## 2.13. setStreamId(int)

```
public void setStreamId(int streamId);
```

Setter for stream id

Parameters

streamId	Stream id
----------	-----------

## 2.14. setTimer(int)

```
public void setTimer(int timer);
```

Setter for timer

Parameters

timer	Timer
-------	-------

## 2.15. setTimerRelative(boolean)

```
public void setTimerRelative(boolean timerRelative);
```

Setter for timer relative flag

Parameters

timerRelative	true if timer values are relative, false otherwise
---------------	--

## 2.16. toString()

```
public String toString();
```

# 3. Class Packet

RTMP packet. Consists of packet header, data and event context.

## 3.1. Synopsis

```
public class Packet implements, java.io.Externalizable {
// Protected Fields
```

```

protected org.apache.mina.common.ByteBuffer data ;

protected Header header ;

protected org.red5.server.net.rtmp.event.IRTMPEvent message ;

// Public Constructors

public Packet();

public Packet(Header header);

public Packet(Header header,
             org.red5.server.net.rtmp.event.IRTMPEvent event);

// Public Methods

public org.apache.mina.common.ByteBuffer getData();

public Header getHeader();

public org.red5.server.net.rtmp.event.IRTMPEvent getMessage();

public void readExternal(java.io.ObjectInput in)
                         throws IOException, ClassNotFoundException;

public void setData(org.apache.mina.common.ByteBuffer data);

public void setMessage(org.red5.server.net.rtmp.event.IRTMPEvent message);

public void writeExternal(java.io.ObjectOutput out)
                         throws IOException;

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 3.2. Packet(Header)

```
public Packet(Header header);
```

Create packet with given header

### Parameters

header	Packet header
--------	---------------

## 3.3. Packet(Header, IRTMPEvent)

```
public Packet(Header header,
             org.red5.server.net.rtmp.event.IRTMPEvent event);
```

Create packet with given header and event context

**Parameters**

header	RTMP header
event	RTMP message

**3.4. data**

```
protected org.apache.mina.common.ByteBuffer data ;
```

**Packet data****3.5. header**

```
protected Header header ;
```

**Header****3.6. message**

```
protected org.red5.server.net.rtmp.event.IRTMPEvent message ;
```

**RTMP event****3.7. getData()**

```
public org.apache.mina.common.ByteBuffer getData();
```

**Getter for data****Parameters**

return	Packet data
--------	-------------

**3.8. getHeader()**

```
public Header getHeader();
```

**Getter for header****Parameters**

return	Packet header
--------	---------------

**3.9. getMessage()**

```
public org.red5.server.net.rtmp.event.IRTMPEvent getMessage();
```

**Getter for event context****Parameters**

return	RTMP event context
--------	--------------------

### 3.10. setData(ByteBuffer)

```
public void setData(org.apache.mina.common.ByteBuffer data);
```

Setter for data

#### Parameters

data	Packet data
------	-------------

### 3.11. setMessage(IRTMPEvent)

```
public void setMessage(org.red5.server.net.rtmp.event.IRTMPEvent message);
```

Setter for event context

#### Parameters

message	RTMP event context
---------	--------------------

## 4. Class SharedObjectTypeMapping

SO event types mapping

### 4.1. Synopsis

```
public class SharedObjectTypeMapping {
// Public Static Fields

    public static final org.red5.server.so.ISharedObjectEvent.Type[] typeMap;

// Protected Fields

    protected static org.slf4j.Logger log;

// Public Constructors

    public SharedObjectTypeMapping();

// Public Static Methods

    public static byte toByte(org.red5.server.so.ISharedObjectEvent.Type type);

    public static String toString(org.red5.server.so.ISharedObjectEvent.Type type);

    public static org.red5.server.so.ISharedObjectEvent.Type toType(byte rtmpType);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 4.2. log

```
protected static org.slf4j.Logger log ;
```

Logger

### 4.3. typeMap

```
public static final org.red5.server.so.ISharedObjectEvent.Type[] typeMap ;
```

Types map

### 4.4. toByte(ISharedObjectEvent.Type)

```
public static byte toByte(org.red5.server.so.ISharedObjectEvent.Type type);
```

Convert SO event type to byte representation that RTMP uses

#### Parameters

<i>type</i>	Event type
<i>return</i>	Byte representation of given event type

### 4.5. toString(ISharedObjectEvent.Type)

```
public static String toString(org.red5.server.so.ISharedObjectEvent.Type type);
```

String representation of type

#### Parameters

<i>type</i>	Type
<i>return</i>	String representation of type

### 4.6. toType(byte)

```
public static org.red5.server.so.ISharedObjectEvent.Type toType(byte rtmpType);
```

Convert byte value of RTMP marker to event type

#### Parameters

<i>rtmpType</i>	RTMP marker value
<i>return</i>	Corresponding Shared Object event type

# 1. Class RuntimeStatusObject

Runtime status object

## 1.1. Synopsis

```
public class RuntimeStatusObject extends org.red5.server.net.rtmp.StatusObject {  
    // Protected Fields  
  
    protected int clientid ;  
  
    protected String details ;  
  
    // Public Constructors  
  
    public RuntimeStatusObject();  
  
    public RuntimeStatusObject(String code,  
                             String level,  
                             String description);  
  
    public RuntimeStatusObject(String code,  
                             String level,  
                             String description,  
                             String details,  
                             int clientid);  
  
    // Public Methods  
  
    public int getClientid();  
  
    public String getDetails();  
  
    public void readExternal(java.io.ObjectInput in)  
        throws IOException, ClassNotFoundException;  
  
    public void setClientid(int clientid);  
  
    public void setDetails(String details);  
  
    public void writeExternal(java.io.ObjectOutput out)  
        throws IOException;  
}
```

**Methods inherited from org.red5.server.net.rtmp.StatusObject:** asStatus ,  
getApplication , getCode , getDescription , getLevel , readExternal , serialize ,  
setAdditional , setApplication , setCode , setDescription , setLevel , toString ,  
writeExternal

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , wait

**Fields inherited from org.red5.server.net.rtmp.StatusObject:** additional ,  
application , code , description , ERROR , level , STATUS , WARNING

## 1.2. RuntimeStatusObject()

```
public RuntimeStatusObject();
```

Constructs a new RuntimeStatusObject.

## 1.3. RuntimeStatusObject(String, String, String)

```
public RuntimeStatusObject(String code,
                           String level,
                           String description);
```

Create runtime status object with given code, level and description

### Parameters

code	Status code
level	Level
description	Status event description

## 1.4. RuntimeStatusObject(String, String, String, String, int)

```
public RuntimeStatusObject(String code,
                           String level,
                           String description,
                           String details,
                           int clientid);
```

Create runtime status object with given code, level, description, details and client id

### Parameters

code	Status code
level	Level
description	Status event description
details	Status event details
clientid	Client id

## 1.5. clientid

```
protected int clientid ;
```

Client id

## 1.6. details

```
protected String details ;
```

Status event details

## 1.7. getClientid()

```
public int getClientid();
```

Getter for client id

### Parameters

<i>return</i>	Client id
---------------	-----------

## 1.8. getDetails()

```
public String getDetails();
```

Getter for details

### Parameters

<i>return</i>	Status event details
---------------	----------------------

## 1.9. setClientid(int)

```
public void setClientid(int clientid);
```

Setter for client id

### Parameters

clientid	Client id
----------	-----------

## 1.10. setDetails(String)

```
public void setDetails(String details);
```

Setter for details

### Parameters

details	Status event details
---------	----------------------

## 2. Class Status

Represents status object that are transferred between server and client

### 2.1. Synopsis

```
public class Status implements, org.?red5.?server.?net.?rtmp.?status.?StatusCodes, org.?red5.?io.?obje
// Public Static Fields

    public static final String ERROR = "error";

    public static final String STATUS = "status";

    public static final String WARNING = "warning";
```

## Package org.?red5.?server.?net.?rtmp.?status

```
// Protected Fields

protected int clientid ;

protected String code ;

protected String description ;

protected String details ;

protected String level ;

// Public Constructors

public Status();

public Status(String code);

public Status(String code,
             String level,
             String description);

// Public Methods

public int getClientid();

public String getCode();

public String getDescription();

public String getDetails();

public String getLevel();

public void readExternal(java.io.ObjectInput in)
    throws IOException, ClassNotFoundException;

public void serialize(org.red5.io.object.Output output,
                     org.red5.io.object.Serializer serializer);

public void setClientid(int clientid);

public void setCode(String code);

public void setDescription(String description);

public void setDescription(String description);

public void setDetails(String details);

public void setLevel(String level);

public String toString();

public void writeExternal(java.io.ObjectOutput out)
    throws IOException;

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. Status()

```
public Status();
```

Constructs a new Status.

## 2.3. Status(String)

```
public Status(String code);
```

Creates status object with given status code

Parameters	
code	Status code

## 2.4. Status(String, String, String)

```
public Status(String code,
              String level,
              String description);
```

Creates status object with given level, description and status code

Parameters	
code	Status code
level	Level
description	Description

## 2.5. clientid

```
protected int clientid;
```

Id of client

## 2.6. code

```
protected String code;
```

Status code

## 2.7. description

```
protected String description;
```

Status event description

## 2.8. details

```
protected String details ;
```

Status event details

## 2.9. ERROR

```
public static final String ERROR = "error";
```

Error constant

## 2.10. level

```
protected String level ;
```

Status level

## 2.11. STATUS

```
public static final String STATUS = "status";
```

Status constant

## 2.12. WARNING

```
public static final String WARNING = "warning";
```

Warning constant

## 2.13. getClientid()

```
public int getClientid();
```

Getter for client id

### Parameters

<i>return</i>	Client id
---------------	-----------

## 2.14. getCode()

```
public String getCode();
```

Getter for status code.

### Parameters

<i>return</i>	Status code
---------------	-------------

## 2.15. getDescription()

```
public String getDescription();
```

Getter for description.

### Parameters

<i>return</i>	Status event description.
---------------	---------------------------

## 2.16. getDetails()

```
public String getDetails();
```

Getter for details

### Parameters

<i>return</i>	Status event details
---------------	----------------------

## 2.17. getLevel()

```
public String getLevel();
```

Getter for level.

### Parameters

<i>return</i>	Level
---------------	-------

## 2.18. serialize(Output, Serializer)

```
public void serialize(org.red5.io.object.Output output,
                     org.red5.io.object.Serializer serializer);
```

**Specified by:** Method `serialize` in interface `ICustomSerializable`

Serialize this object to the given output stream.

### Parameters

<i>output</i>	
---------------	--

<i>serializer</i>	
-------------------	--

**Description copied from interface: `serialize`**

## 2.19. setClientid(int)

```
public void setClientid(int clientid);
```

Setter for client id

Parameters

clientid	Client id
----------	-----------

## 2.20. setCode(String)

```
public void setCode(String code);
```

Setter for code

Parameters

code	Status code
------	-------------

## 2.21. setDescription(String)

```
public void setDescription(String description);
```

Setter for description.

Parameters

description	Status event description.
-------------	---------------------------

## 2.22. setDescription(String)

```
public void setDescription(String description);
```

Setter for description.

Parameters

description	Status event description
-------------	--------------------------

## 2.23. setDetails(String)

```
public void setDetails(String details);
```

Setter for details.

Parameters

details	Status event details
---------	----------------------

## 2.24. setLevel(String)

```
public void setLevel(String level);
```

Setter for level

## Parameters

level	Level
-------	-------

**2.25. `toString()`**

```
public String toString();
```

## 3. Interface StatusCodes

Collection of commonly used constants with status codes. Descriptions provided as in FMS 2.0.1 documentation available at adobe.com with some minor additions and comments.

### 3.1. Synopsis

```
public interface StatusCodes {
    // Public Static Fields

    public static final String APP_GC = "Application.GC";

    public static final String APP_RESOURCE_LOWMEMORY = "Application.Resource.LowMemory";

    public static final String APP_SCRIPT_ERROR = "Application.Script.Error";

    public static final String APP_SCRIPT_WARNING = "Application.Script.Warning";

    public static final String APP_SHUTDOWN = "Application.Shutdown";

    public static final String NC_CALL_BADVERSION = "NetConnection.Call.BadVersion";

    public static final String NC_CALL_FAILED = "NetConnection.CallFailed";

    public static final String NC_CONNECT_APPSHUTDOWN = "NetConnection.Connect.AppShutdown";

    public static final String NC_CONNECT_CLOSED = "NetConnection.Connect.Closed";

    public static final String NC_CONNECT_FAILED = "NetConnection.ConnectFailed";

    public static final String NC_CONNECT_INVALID_APPLICATION = "NetConnection.Connect.InvalidApp";

    public static final String NC_CONNECT_REJECTED = "NetConnection.Connect.Rejected";

    public static final String NC_CONNECT_SUCCESS = "NetConnection.Connect.Success";

    public static final String NS_CLEAR_FAILED = "NetStream.ClearFailed";

    public static final String NS_CLEAR_SUCCESS = "NetStream.ClearSuccess";

    public static final String NS_DATA_START = "NetStream.DataStart";

    public static final String NS_FAILED = "NetStreamFailed";

    public static final String NS_INVALID_ARGUMENT = "NetStream.InvalidArg";
```

## Package org.?red5.?server.?net.?rtmp.?status

```
public static final String NS_PAUSE_NOTIFY = "NetStream.Pause.Notify";  
  
public static final String NS_PLAY_COMPLETE = "NetStream.Play.Complete";  
  
public static final String NS_PLAY_FAILED = "NetStream.PlayFailed";  
  
public static final String NS_PLAY_FILE_STRUCTURE_INVALID = "NetStream.Play.FileStructureInvalid";  
  
public static final String NS_PLAY_INSUFFICIENT_BW = "NetStream.Play.InsufficientBW";  
  
public static final String NS_PLAY_NO_SUPPORTED_TRACK_FOUND = "NetStream.Play.NoSupportedTrackFound";  
  
public static final String NS_PLAY_PUBLISHNOTIFY = "NetStream.Play.PublishNotify";  
  
public static final String NS_PLAY_RESET = "NetStream.Play.Reset";  
  
public static final String NS_PLAY_START = "NetStream.Play.Start";  
  
public static final String NS_PLAY_STOP = "NetStream.Play.Stop";  
  
public static final String NS_PLAY_STREAMNOTFOUND = "NetStream.Play.StreamNotFound";  
  
public static final String NS_PLAY_SWITCH = "NetStream.Play.Switch";  
  
public static final String NS_PLAY_UNPUBLISHNOTIFY = "NetStream.Play.UnpublishNotify";  
  
public static final String NS_PUBLISH_BADNAME = "NetStream.Publish.BadName";  
  
public static final String NS_PUBLISH_START = "NetStream.Publish.Start";  
  
public static final String NS_RECORD_FAILED = "NetStream.RecordFailed";  
  
public static final String NS_RECORD_NOACCESS = "NetStream.Record.NoAccess";  
  
public static final String NS_RECORD_START = "NetStream.Record.Start";  
  
public static final String NS_RECORD_STOP = "NetStream.Record.Stop";  
  
public static final String NS_SEEK_FAILED = "NetStream.SeekFailed";  
  
public static final String NS_SEEK_NOTIFY = "NetStream.SeekNotify";  
  
public static final String NS_UNPAUSE_NOTIFY = "NetStream.UnpauseNotify";  
  
public static final String NS_UNPUBLISHED_SUCCESS = "NetStream.UnpublishSuccess";  
  
public static final String SO_CREATION_FAILED = "SharedObject.ObjectCreationFailed";  
  
public static final String SO_NO_READ_ACCESS = "SharedObject.NoReadAccess";  
  
public static final String SO_NO_WRITE_ACCESS = "SharedObject.NoWriteAccess";  
  
public static final String SO_PERSISTENCE_MISMATCH = "SharedObject.BadPersistence";  
  
}
```

## 3.2. APP\_GC

```
public static final String APP_GC = "Application.GC";
```

This information object is passed to the onAppStop event handler when the application instance is about to be destroyed by the server.

## 3.3. APP\_RESOURCE\_LOWMEMORY

```
public static final String APP_RESOURCE_LOWMEMORY = "Application.Resource.LowMemory";
```

The ActionScript engine is low on runtime memory. This provides an opportunity for the application instance to free some resources or take suitable action. If the application instance runs out of memory, it is unloaded and all users are disconnected. In this state, the server will not invoke the Application.onDisconnect event handler or the Application.onAppStop event handler

## 3.4. APP\_SCRIPT\_ERROR

```
public static final String APP_SCRIPT_ERROR = "Application.Script.Error";
```

The ActionScript engine has encountered a runtime error. In addition to the standard infoObject properties, the following properties are set: filename: name of the offending ASC file. lineno: line number where the error occurred. linebuf: source code of the offending line.

## 3.5. APP\_SCRIPT\_WARNING

```
public static final String APP_SCRIPT_WARNING = "Application.Script.Warning";
```

The ActionScript engine has encountered a runtime warning. In addition to the standard infoObject properties, the following properties are set: filename: name of the offending ASC file. lineno: line number where the error occurred. linebuf: source code of the offending line

## 3.6. APP\_SHUTDOWN

```
public static final String APP_SHUTDOWN = "Application.Shutdown";
```

This information object is passed to the onAppStop handler when the application is being shut down

## 3.7. NC\_CALL\_BADVERSION

```
public static final String NC_CALL_BADVERSION = "NetConnection.Call.BadVersion";
```

The URI specified in the NetConnection.connect method did not specify 'rtmp' as the protocol. 'rtmp' must be specified when connecting to FMS and Red5. Either not supported version of AMF was used (3 when only 0 is supported)

## 3.8. NC\_CALL\_FAILED

```
public static final String NC_CALL_FAILED = "NetConnection.Call.Failed";
```

The NetConnection.call method was not able to invoke the server-side method or command.

### **3.9. NC\_CONNECT\_APPSHUTDOWN**

```
public static final String NC_CONNECT_APPSHUTDOWN = "NetConnection.Connect.AppShutdown";
```

The application has been shut down (for example, if the application is out of memory resources and must shut down to prevent the server from crashing) or the server has shut down.

### **3.10. NC\_CONNECT\_CLOSED**

```
public static final String NC_CONNECT_CLOSED = "NetConnection.Connect.Closed";
```

The connection was closed successfully

### **3.11. NC\_CONNECT\_FAILED**

```
public static final String NC_CONNECT_FAILED = "NetConnection.Connect.Failed";
```

The connection attempt failed.

### **3.12. NC\_CONNECT\_INVALID\_APPLICATION**

```
public static final String NC_CONNECT_INVALID_APPLICATION = "NetConnection.Connect.InvalidApp";
```

The application name specified during connect is invalid.

### **3.13. NC\_CONNECT\_REJECTED**

```
public static final String NC_CONNECT_REJECTED = "NetConnection.Connect.Rejected";
```

The client does not have permission to connect to the application, the application expected different parameters from those that were passed, or the application name specified during the connection attempt was not found on the server.

### **3.14. NC\_CONNECT\_SUCCESS**

```
public static final String NC_CONNECT_SUCCESS = "NetConnection.Connect.Success";
```

The connection attempt succeeded.

### **3.15. NS\_CLEAR\_FAILED**

```
public static final String NS_CLEAR_FAILED = "NetStream.Clear.Failed";
```

A recorded stream failed to delete.

### **3.16. NS\_CLEAR\_SUCCESS**

```
public static final String NS_CLEAR_SUCCESS = "NetStream.Clear.Success";
```

A recorded stream was deleted successfully.

### 3.17. NS\_DATA\_START

```
public static final String NS_DATA_START = "NetStream.Data.Start";
```

### 3.18. NS\_FAILED

```
public static final String NS_FAILED = "NetStream.Failed";
```

An attempt to use a Stream method (at client-side) failed

### 3.19. NS\_INVALID\_ARGUMENT

```
public static final String NS_INVALID_ARGUMENT = "NetStream.InvalidArg";
```

Invalid arguments were passed to a NetStream method.

### 3.20. NS\_PAUSE\_NOTIFY

```
public static final String NS_PAUSE_NOTIFY = "NetStream.Pause.Notify";
```

The subscriber has used the seek command to move to a particular location in the recorded stream.

### 3.21. NS\_PLAY\_COMPLETE

```
public static final String NS_PLAY_COMPLETE = "NetStream.Play.Complete";
```

Playlist playback is complete.

### 3.22. NS\_PLAY\_FAILED

```
public static final String NS_PLAY_FAILED = "NetStream.PlayFailed";
```

An attempt to play back a stream failed

### 3.23. NS\_PLAY\_FILE\_STRUCTURE\_INVALID

```
public static final String NS_PLAY_FILE_STRUCTURE_INVALID = "NetStream.Play.FileStructureInvalid";
```

This event is sent if the player detects an MP4 with an invalid file structure. Flash Player cannot play files that have invalid file structures. New for FMS3

### 3.24. NS\_PLAY\_INSUFFICIENT\_BW

```
public static final String NS_PLAY_INSUFFICIENT_BW = "NetStream.Play.InsufficientBW";
```

Data is playing behind the normal speed

### 3.25. NS\_PLAY\_NO\_SUPPORTED\_TRACK\_FOUND

```
public static final String NS_PLAY_NO_SUPPORTED_TRACK_FOUND = "NetStream.Play.NoSupportedTrackFound"
```

This event is sent if the player does not detect any supported tracks. If there aren't any supported video, audio or data tracks found, Flash Player does not play the file. New for FMS3

### **3.26. NS\_PLAY\_PUBLISHNOTIFY**

```
public static final String NS_PLAY_PUBLISHNOTIFY = "NetStream.Play.PublishNotify";
```

The initial publish to a stream was successful. This message is sent to all subscribers

### **3.27. NS\_PLAY\_RESET**

```
public static final String NS_PLAY_RESET = "NetStream.Play.Reset";
```

A playlist was reset

### **3.28. NS\_PLAY\_START**

```
public static final String NS_PLAY_START = "NetStream.Play.Start";
```

Play was started

### **3.29. NS\_PLAY\_STOP**

```
public static final String NS_PLAY_STOP = "NetStream.Play.Stop";
```

Play was stopped

### **3.30. NS\_PLAY\_STREAMNOTFOUND**

```
public static final String NS_PLAY_STREAMNOTFOUND = "NetStream.Play.StreamNotFound";
```

An attempt was made to play a stream that does not exist

### **3.31. NS\_PLAY\_SWITCH**

```
public static final String NS_PLAY_SWITCH = "NetStream.Play.Switch";
```

Playlist playback switched from one stream to another.

### **3.32. NS\_PLAY\_UNPUBLISHNOTIFY**

```
public static final String NS_PLAY_UNPUBLISHNOTIFY = "NetStream.Play.UnpublishNotify";
```

An unpublish from a stream was successful. This message is sent to all subscribers

### **3.33. NS\_PUBLISH\_BADNAME**

```
public static final String NS_PUBLISH_BADNAME = "NetStream.Publish.BadName";
```

An attempt was made to publish a stream that is already being published by someone else.

### 3.34. NS\_PUBLISH\_START

```
public static final String NS_PUBLISH_START = "NetStream.Publish.Start";
```

An attempt to publish was successful.

### 3.35. NS\_RECORD\_FAILED

```
public static final String NS_RECORD_FAILED = "NetStream.Record.Failed";
```

An attempt to record a stream failed

### 3.36. NS\_RECORD\_NOACCESS

```
public static final String NS_RECORD_NOACCESS = "NetStream.Record.NoAccess";
```

An attempt was made to record a read-only stream

### 3.37. NS\_RECORD\_START

```
public static final String NS_RECORD_START = "NetStream.Record.Start";
```

Recording was started

### 3.38. NS\_RECORD\_STOP

```
public static final String NS_RECORD_STOP = "NetStream.Record.Stop";
```

Recording was stopped

### 3.39. NS\_SEEK\_FAILED

```
public static final String NS_SEEK_FAILED = "NetStream.SeekFailed";
```

The stream doesn't support seeking.

### 3.40. NS\_SEEK\_NOTIFY

```
public static final String NS_SEEK_NOTIFY = "NetStream.Seek.Notify";
```

The subscriber has used the seek command to move to a particular location in the recorded stream.

### 3.41. NS\_UNPAUSE\_NOTIFY

```
public static final String NS_UNPAUSE_NOTIFY = "NetStream.Unpause.Notify";
```

Publishing has stopped

### 3.42. NS\_UNPUBLISHED\_SUCCESS

```
public static final String NS_UNPUBLISHED_SUCCESS = "NetStream.Unpublish.Success";
```

An attempt to unpublish was successful

### 3.43. SO\_CREATION\_FAILED

```
public static final String SO_CREATION_FAILED = "SharedObject.ObjectCreationFailed";
```

The creation of a shared object was denied.

### 3.44. SO\_NO\_READ\_ACCESS

```
public static final String SO_NO_READ_ACCESS = "SharedObject.NoReadAccess";
```

Read access to a shared object was denied.

### 3.45. SO\_NO\_WRITE\_ACCESS

```
public static final String SO_NO_WRITE_ACCESS = "SharedObject.NoWriteAccess";
```

Write access to a shared object was denied.

### 3.46. SO\_PERSISTENCE\_MISMATCH

```
public static final String SO_PERSISTENCE_MISMATCH = "SharedObject.BadPersistence";
```

The persistence parameter passed to SharedObject.getRemote() is different from the one used when the shared object was created.

## 4. Class StatusObject

Status object that is sent to client with every status event

### 4.1. Synopsis

```
public class StatusObject implements, java.?io.?Serializable, org.?red5.?io.?object.?ICustomSerializah
// Public Static Fields

    public static final String ERROR = "error";

    public static final String STATUS = "status";

    public static final String WARNING = "warning";

// Protected Fields

    protected java.util.Map<java.lang.String, java.lang.Object> additional ;

    protected Object application ;

    protected String code ;

    protected String description ;

    protected String level ;

// Public Constructors

    public StatusObject();
```

```

public StatusObject(String code,
                    String level,
                    String description);

// Public Methods

public Status asStatus();

public Object getApplication();

public String getCode();

public String getDescription();

public String getLevel();

public void readExternal(java.io.ObjectInput in)
    throws IOException, ClassNotFoundException;

public void serialize(org.red5.io.object.Output output,
                     org.red5.io.object.Serializer serializer);

public void setAdditional(String name,
                         Object value);

public void setApplication(Object application);

public void setCode(String code);

public void setDescription(String description);

public void setLevel(String level);

public String toString();

public void writeExternal(java.io.ObjectOutput out)
    throws IOException;

}

```

**Direct known subclasses:** org.?red5.?server.?net.?rtmp.?status.?RuntimeStatusObject

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 4.2. StatusObject()

```
public StatusObject();
```

Constructs a new StatusObject.

## 4.3. asStatus()

```
public Status asStatus();
```

Generate Status object that can be returned through a RTMP channel.

#### Parameters

<i>return</i>	
---------------	--

### 4.4. getApplication()

<code>public Object getApplication();</code>
--

Getter for property 'application'.

#### Parameters

<i>return</i>	Value for property 'application'.
---------------	-----------------------------------

### 4.5. getCode()

<code>public String getCode();</code>
---------------------------------------

Getter for property 'code'.

#### Parameters

<i>return</i>	Value for property 'code'.
---------------	----------------------------

### 4.6. getDescription()

<code>public String getDescription();</code>
--

Getter for property 'description'.

#### Parameters

<i>return</i>	Value for property 'description'.
---------------	-----------------------------------

### 4.7. getLevel()

<code>public String getLevel();</code>
--

Getter for property 'level'.

#### Parameters

<i>return</i>	Value for property 'level'.
---------------	-----------------------------

### 4.8. serialize(Output, Serializer)

<code>public void serialize(org.red5.io.object.Output output, org.red5.io.object.Serializer serializer);</code>
---

**Specified by:** Method `serialize` in interface `ICustomSerializable`

Serialize this object to the given output stream.

**Parameters**

output

serializer

**Description copied from interface: serialize****4.9. setApplication(Object)**

```
public void setApplication(Object application);
```

Setter for property 'application'.

**Parameters**

application Value to set for property 'application'.

**4.10. setCode(String)**

```
public void setCode(String code);
```

Setter for property 'code'.

**Parameters**

code Value to set for property 'code'.

**4.11. setDescription(String)**

```
public void setDescription(String description);
```

Setter for property 'description'.

**Parameters**

description Value to set for property 'description'.

**4.12. setLevel(String)**

```
public void setLevel(String level);
```

Setter for property 'level'.

**Parameters**

level Value to set for property 'level'.

**4.13. toString()**

```
public String toString();
```

**5. Class StatusObjectService**

Service that works with status objects. Note all status object should aim to be under 128 bytes.

## 5.1. Synopsis

```
public class StatusObjectService implements, org.?red5.?server.?net.?rtmp.?status.?StatusCodes {
    // Protected Fields

    protected java.util.Map<java.lang.String, byte[]> cachedStatusObjects ;

    protected static org.slf4j.Logger log ;

    protected org.red5.io.object.Serializer serializer ;

    protected java.util.Map<java.lang.String, org.red5.server.net.rtmp.status.StatusObject> statusObje

    // Public Constructors

    public StatusObjectService();

    // Public Methods

    public void cacheStatusObjects();

    public byte[] getCachedStatusObjectAsByteArray(String statusCode);

    public StatusObject getStatusObject(String statusCode);

    public void initialize();

    public void loadStatusObjects();

    public void serializeStatusObject(org.apache.mina.common.ByteBuffer out,
                                     StatusObject statusObject);

    public void setSerializer(org.red5.io.object.Serializer serializer);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 5.2. cachedStatusObjects

```
protected java.util.Map<java.lang.String, byte[]> cachedStatusObjects ;
```

Cached status objects map

## 5.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 5.4. serializer

```
protected org.red5.io.object.Serializer serializer ;
```

Serializer

## 5.5. statusObjects

```
protected java.util.Map<java.lang.String, org.red5.server.net.rtmp.status.StatusObject> statusObjects
```

Status objects map

## 5.6. cacheStatusObjects()

```
public void cacheStatusObjects();
```

Cache status objects

## 5.7. getCachedStatusObjectAsByteArray(String)

```
public byte[] getCachedStatusObjectAsByteArray(String statusCode);
```

Return status object by code as byte array

### Parameters

statusCode	Status object code
<i>return</i>	Status object with given code as byte array

## 5.8. getStatusObject(String)

```
public StatusObject getStatusObject(String statusCode);
```

Return status object by code

### Parameters

statusCode	Status object code
<i>return</i>	Status object with given code

## 5.9. initialize()

```
public void initialize();
```

Initialization

## 5.10. loadStatusObjects()

```
public void loadStatusObjects();
```

Creates all status objects and adds them to status objects map

## 5.11. serializeStatusObject(ByteBuffer, StatusObject)

```
public void serializeStatusObject(org.apache.mina.common.ByteBuffer out,  
                                StatusObject statusObject);
```

Serializes status object

### Parameters

out	Byte buffer for output object
statusObject	Status object to serialize

## 5.12. setSerializer(Serializer)

```
public void setSerializer(org.red5.io.object.Serializer serializer);
```

Setter for serializer

### Parameters

serializer	Serializer object
------------	-------------------

# 1. Class TomcatRTMPSLoader

Loader for the RTMPS server which uses Tomcat.

## 1.1. Synopsis

```
public class TomcatRTMPSLoader extends org.red5.server.net.rtmpt.TomcatRTMPTLoader {  
    // Protected Fields  
  
    protected org.apache.catalina.Engine rtmpsEngine;  
  
    // Public Constructors  
  
    public TomcatRTMPSLoader();  
  
    // Public Methods  
  
    public void init();  
  
    public void setServer(org.red5.server.api.IServer server);  
}
```

**Methods inherited from org.red5.server.net.rtmpt.TomcatRTMPTLoader:** init , setMappings , setServer

**Methods inherited from org.red5.server.tomcat.TomcatLoader:** addContext , formatPath , getBaseHost , getConnector , getEmbedded , getEngine , getHost , getRealm , registerJMX , removeContext , setBaseHost , setConnectionProperties , setConnector , setConnectors , setContexts , setEmbedded , setHost , setHosts , setRealm , setValves , shutdown

**Methods inherited from org.red5.server.LoaderBase:** getApplicationContext , getApplicationLoader , getRed5ApplicationContext , removeRed5ApplicationContext , setApplicationContext , setApplicationLoader , setRed5ApplicationContext , setWebappFolder

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmpt.TomcatRTMPTLoader:** context , rtmpsEngine , server , servletMappings

**Fields inherited from org.red5.server.tomcat.TomcatLoader:** connectionProperties , connector , defaultParentContextKey , defaultSpringConfigLocation , embedded , engine , host , realm , valves

**Fields inherited from org.red5.server.LoaderBase:** applicationContext , loader , red5AppCtx , webappFolder

## 1.2. rtmpsEngine

```
protected org.apache.catalina.Engine rtmpsEngine ;
```

RTMPS Tomcat engine.

### 1.3. init()

```
public void init();
```

### 1.4. setServer(I Server)

```
public void setServer(org.red5.server.api.I Server server);
```

Setter for server

#### Parameters

server	Value to set for property 'server'.
--------	-------------------------------------

# 1. Class BaseRTMPTConnection

```
public abstract class BaseRTMPTConnection extends org.red5.server.net.RTMPConnection {  
    // Protected Fields  
  
    protected org.apache.mina.common.ByteBuffer buffer ;  
  
    protected volatile boolean closing ;  
  
    protected org.red5.server.net.protocol.SimpleProtocolDecoder decoder ;  
  
    protected org.red5.server.net.protocol.SimpleProtocolEncoder encoder ;  
  
    protected org.red5.server.net.rtmp.IRTMPHandler handler ;  
  
    protected java.util.List<java.lang.Object> notifyMessages ;  
  
    protected final java.util.concurrent.locks.Lock notifyRead ;  
  
    protected final java.util.concurrent.locks.Lock notifyWrite ;  
  
    protected java.util.List<org.apache.mina.common.ByteBuffer> pendingMessages ;  
  
    protected final java.util.concurrent.locks.Lock pendingRead ;  
  
    protected final java.util.concurrent.locks.Lock pendingWrite ;  
  
    protected long readBytes ;  
  
    protected long writtenBytes ;  
  
    // Public Constructors  
  
    public BaseRTMPTConnection(String type);  
  
    // Public Methods  
  
    public void close();  
  
    public java.util.List decode(org.apache.mina.common.ByteBuffer data);  
  
    public long getPendingMessages();  
  
    public abstract org.apache.mina.common.ByteBuffer getPendingMessages(int targetSize);  
  
    public long getReadBytes();  
  
    public long getWrittenBytes();  
  
    public boolean isClosing();  
  
    public void rawWrite(org.apache.mina.common.ByteBuffer packet);  
  
    public void realClose();
```

```

    public void write(org.red5.server.net.rtmp.message.Packet packet);

    // Protected Methods

    protected org.apache.mina.common.ByteBuffer foldPendingMessages(int targetSize);

}

```

**Direct known subclasses:** org.?red5.?server.?net.?rtmpt.?RTMPTClientConnection , org.?red5.?server.?net.?rtmpt.?RTMPTConnection

**Methods inherited from org.red5.server.net.rtmp.RTMPConnection:** addClientStream , close , closeChannel , connect , createOutputStream , createStreamName , deleteStreamById , getBandwidthConfigure , getChannel , getClientBytesRead , getEncoding , getId , getInvokeId , getLastPingTime , getNextAvailableChannelId , getParentBWControllable , getPendingCall , getPendingVideoMessages , getReadBytes , getState , getStreamByChannelId , getStreamById , getStreamIdForChannel , getStreams , getUsedStreamCount , getVideoCodecFactory , getWrittenBytes , invoke , isChannelUsed , messageDropped , messageReceived , messageSent , newBroadcastStream , newPlaylistSubscriberStream , newSingleItemSubscriberStream , notify , onInactive , ping , pingReceived , rawWrite , receivedBytesRead , registerDeferredResult , registerPendingCall , registerStream , rememberStreamBufferDuration , removeClientStream , reserveStreamId , retrievePendingCall , setBandwidthConfigure , setId , setMaxHandshakeTimeout , setMaxInactivity , setPingInterval , setSchedulingService , setState , setup , startRoundTripMeasurement , startWaitForHandshake , toString , unregisterDeferredResult , unregisterStream , unreserveStreamId , updateBytesRead , write , writingMessage

**Methods inherited from org.red5.server.BaseConnection:** dispatchEvent , getBasicScopes , getClient , getConnectParams , getDroppedMessages , getHost , getPath , getPendingMessages , getReadMessages , getRemoteAddress , getRemoteAddresses , getRemotePort , getScope , getSessionId , getType , getWrittenMessages , handleEvent , initialize , isConnected , notifyEvent , registerBasicScope , unregisterBasicScope

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttribute , getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute , getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute , getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute , hasAttribute , removeAttribute , removeAttributes , setAttribute , setAttributes

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notifyAll , wait

**Fields inherited from org.red5.server.net.rtmp.RTMPConnection:** clientId , deferredResults , encoding , invokeId , keepAliveJobName , lastPingSent , lastPingTime , lastPongReceived , log , maxInactivity , oName , pendingCalls , pingInterval , state , streamBuffers

**Fields inherited from org.red5.server.BaseConnection:** basicScopes , client , closed , droppedMessages , host , params , path , readMessages , remoteAddress , remoteAddresses , remotePort , scope , sessionId , type , writtenMessages

Fields inherited from **org.red5.server.AttributeStore**: attributes

## 1.1. buffer

```
protected org.apache.mina.common.ByteBuffer buffer ;
```

Byte buffer

## 1.2. closing

```
protected volatile boolean closing ;
```

Closing flag

## 1.3. decoder

```
protected org.red5.server.net.protocol.SimpleProtocolDecoder decoder ;
```

Protocol decoder

## 1.4. encoder

```
protected org.red5.server.net.protocol.SimpleProtocolEncoder encoder ;
```

Protocol encoder

## 1.5. handler

```
protected org.red5.server.net.rtmp.IRTMPHandler handler ;
```

RTMP events handler

## 1.6. notifyMessages

```
protected java.util.List<java.lang.Object> notifyMessages ;
```

List of notification messages

## 1.7. pendingMessages

```
protected java.util.List<org.apache.mina.common.ByteBuffer> pendingMessages ;
```

List of pending messages

## 1.8. readBytes

```
protected long readBytes ;
```

Number of read bytes

## 1.9. writtenBytes

```
protected long writtenBytes ;
```

Number of written bytes

## 1.10. close()

```
public void close();
```

## 1.11. decode(ByteBuffer)

```
public java.util.List decode(org.apache.mina.common.ByteBuffer data);
```

Decode data sent by the client.

### Parameters

<i>data</i>	the data to decode
<i>return</i>	a list of decoded objects

## 1.12. getPendingMessages()

```
public long getPendingMessages();
```

## 1.13. getPendingMessages(int)

```
public abstract org.apache.mina.common.ByteBuffer getPendingMessages(int targetSize);
```

Return any pending messages up to a given size.

### Parameters

<i>targetSize</i>	the size the resulting buffer should have
<i>return</i>	a buffer containing the data to send or null if no messages are pending

## 1.14. getReadBytes()

```
public long getReadBytes();
```

## 1.15. getWrittenBytes()

```
public long getWrittenBytes();
```

## 1.16. isClosing()

```
public boolean isClosing();
```

Getter for property 'closing'.

### Parameters

<i>return</i>	Value for property 'closing'.
---------------	-------------------------------

## 1.17. rawWrite(ByteBuffer)

```
public void rawWrite(org.apache.mina.common.ByteBuffer packet);
```

Send raw data down the connection.

### Parameters

packet	the buffer containing the raw data
--------	------------------------------------

## 1.18. realClose()

```
public void realClose();
```

Real close

## 1.19. write(Packet)

```
public void write(org.red5.server.net.rtmp.message.Packet packet);
```

Send RTMP packet down the connection.

### Parameters

packet	the packet to send
--------	--------------------

## 2. Class RTMPTClient

RTMPT client object

### 2.1. Synopsis

```
public class RTMPTClient extends org.?red5.?server.?net.?rtmp.?BaseRTMPClientHandler {
// Public Constructors

    public RTMPTClient();

// Public Methods

    public java.util.Map<java.lang.String, java.lang.Object> makeDefaultConnectionParams(String server,
                                                                                           int port,
                                                                                           String applic

    public void messageReceived(org.red5.server.net.rtmp.RTMPConnection conn,
                               org.red5.server.net.protocol.ProtocolState state,
                               Object in)
        throws Exception;

// Protected Methods

    protected void startConnector(String server,
                                 int port);

}
```

#### Methods inherited from org.red5.server.net.rtmp.BaseRTMPClientHandler:

connect , connectionClosed , connectionOpened , createStream , disconnect ,

```
getCodecFactory , getConnManager , getSharedObject , handleException , invoke ,
makeDefaultConnectionParams , onChunkSize , onInvoke , onPing , onSharedObject ,
play , publish , publishStreamData , setCodecFactory , setConnectionClosedHandler ,
setExceptionHandler , setServiceProvider , setStreamEventDispatcher , startConnector
```

**Methods inherited from org.red5.server.net.rtmp.BaseRTMPHandler:** getHostname ,  
getStreamId , handlePendingCallResult , messageReceived , messageSent ,  
onStreamBytesRead , setApplicationContext

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.BaseRTMPClientHandler:**

```
codecFactory , connectArguments , connectCallback , connectionParams , connManager ,
serviceInvoker , serviceProvider , sharedObjects , streamEventDispatcher
```

**Fields inherited from org.red5.server.net.rtmp.BaseRTMPHandler:** appCtx , log

## 2.2. messageReceived(RTMPConnection, ProtocolState, Object)

```
public void messageReceived(org.red5.server.net.rtmp.RTMPConnection conn,
                           org.red5.server.net.protocol.ProtocolState state,
                           Object in)
                           throws Exception;
```

## 3. Class RTMPTClientConnection

```
public class RTMPTClientConnection extends org.?red5.?server.?net.?rtmpt.?BaseRTMPTConnection {
// Public Constructors

    public RTMPTClientConnection();

// Public Methods

    public org.apache.mina.common.ByteBuffer getPendingMessages(int targetSize);

    public void setClient(RTMPTClient handler);

// Protected Methods

    protected void onInactive();

}
```

**Methods inherited from org.red5.server.net.rtmpt.BaseRTMPTConnection:** close ,  
decode , foldPendingMessages , getPendingMessages , getReadBytes , getWrittenBytes ,  
isClosing , rawWrite , realClose , write

**Methods inherited from org.red5.server.net.rtmp.RTMPConnection:** addClientStream  
, closeChannel , connect , createOutputStream , createStreamName , deleteStreamById  
, getBandwidthConfigure , getChannel , getClientBytesRead , getEncoding  
, getId , getInvokeId , getLastPingTime , getNextAvailableChannelId ,  
getParentBWControllable , getPendingCall , getPendingVideoMessages , getState

```
, getStreamByChannelId , getStreamById , getStreamIdForChannel , getStreams ,
getUsedStreamCount , getVideoCodecFactory , invoke , isChannelUsed , messageDropped ,
messageReceived , messageSent , newBroadcastStream , newPlaylistSubscriberStream ,
newSingleItemSubscriberStream , notify , onInactive , ping , pingReceived ,
receivedBytesRead , registerDeferredResult , registerPendingCall , registerStream ,
rememberStreamBufferDuration , removeClientStream , reserveStreamId ,
retrievePendingCall , setBandwidthConfigure , setId , setMaxHandshakeTimeout ,
setMaxInactivity , setPingInterval , setSchedulingService , setState ,
setup , startRoundTripMeasurement , startWaitForHandshake , toString ,
unregisterDeferredResult , unregisterStream , unreserveStreamId , updateBytesRead ,
writingMessage
```

**Methods inherited from org.red5.server.BaseConnection:** dispatchEvent ,  
 getBasicScopes , getClient , getConnectParams , getDroppedMessages , getHost ,  
 getPath , getReadMessages , getRemoteAddress , getRemoteAddresses , getRemotePort ,  
 getScope , getSessionId , getType , getWrittenMessages , handleEvent , initialize ,  
 isConnected , notifyEvent , registerBasicScope , unregisterBasicScope

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttribute ,  
 getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute ,  
 getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute ,  
 getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute ,  
 hasAttribute , removeAttribute , removeAttributes , setAttribute , setAttributes

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
 , notifyAll , wait

**Fields inherited from org.red5.server.net.rtmpt.BaseRTMPTConnection:** buffer ,  
 closing , decoder , encoder , handler , notifyMessages , notifyRead , notifyWrite ,  
 pendingMessages , pendingRead , pendingWrite , readBytes , writtenBytes

**Fields inherited from org.red5.server.net.rtmp.RTMPConnection:** clientId ,  
 deferredResults , encoding , invokeId , keepAliveJobName , lastPingSent ,  
 lastPingTime , lastPongReceived , log , maxInactivity , oName , pendingCalls ,  
 pingInterval , state , streamBuffers

**Fields inherited from org.red5.server.BaseConnection:** basicScopes , client ,  
 closed , droppedMessages , host , params , path , readMessages , remoteAddress ,  
 remoteAddresses , remotePort , scope , sessionId , type , writtenMessages

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 4. Class RTMPTConnection

A RTMPT client / session.

### 4.1. Synopsis

```
public class RTMPTConnection extends org.?red5.?server.?net.?rtmpt.?BaseRTMPTConnection {  

// Protected Fields
```

```

protected static final long INCREASE_POLLING_DELAY_COUNT = 10L;

protected static final byte INITIAL_POLLING_DELAY = 0;

protected static final byte MAX_POLLING_DELAY = 32;

protected volatile long noPendingMessages ;

protected volatile byte pollingDelay ;

protected RTMPTServlet servlet ;

// Public Methods

public org.apache.mina.common.ByteBuffer getPendingMessages(int targetSize);

public byte getPollingDelay();

public void realClose();

public void setServletRequest(javax.servlet.http.HttpServletRequest request);

// Protected Methods

protected void onInactive();

protected void setServlet(RTMPTServlet servlet);

}

```

**Methods inherited from org.red5.server.net.rtmp.BaseRTMPTConnection:** close , decode , foldPendingMessages , getPendingMessages , getReadBytes , getWrittenBytes , isClosing , rawWrite , realClose , write

**Methods inherited from org.red5.server.net.rtmp.RTMPConnection:** addClientStream , closeChannel , connect , createOutputStream , createStreamName , deleteStreamById , getBandwidthConfigure , getChannel , getClientBytesRead , getEncoding , getId , getInvokeId , getLastPingTime , getNextAvailableChannelId , getParentBWControllable , getPendingCall , getPendingVideoMessages , getState , getStreamByChannelId , getStreamById , getStreamIdForChannel , getStreams , getUsedStreamCount , getVideoCodecFactory , invoke , isChannelUsed , messageDropped , messageReceived , messageSent , newBroadcastStream , newPlaylistSubscriberStream , newSingleItemSubscriberStream , notify , onInactive , ping , pingReceived , receivedBytesRead , registerDeferredResult , registerPendingCall , registerStream , rememberStreamBufferDuration , removeClientStream , reserveStreamId , retrievePendingCall , setBandwidthConfigure , setId , setMaxHandshakeTimeout , setMaxInactivity , setPingInterval , setSchedulingService , setState , setup , startRoundTripMeasurement , startWaitForHandshake , toString , unregisterDeferredResult , unregisterStream , unreserveStreamId , updateBytesRead , writingMessage

**Methods inherited from org.red5.server.BaseConnection:** dispatchEvent , getBasicScopes , getClient , getConnectParams , getDroppedMessages , getHost , getPath , getReadMessages , getRemoteAddress , getRemoteAddresses , getRemotePort ,

```
getScope , getSessionId , getType , getWrittenMessages , handleEvent , initialize ,
isConnected , notifyEvent , registerBasicScope , unregisterBasicScope
```

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttribute ,  
 getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute ,  
 getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute ,  
 getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute ,  
 hasAttribute , removeAttribute , removeAttributes , setAttribute , setAttributes

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
 , notifyAll , wait

**Fields inherited from org.red5.server.net.rtmpt.BaseRTMPTConnection:** buffer ,  
 closing , decoder , encoder , handler , notifyMessages , notifyRead , notifyWrite ,  
 pendingMessages , pendingRead , pendingWrite , readBytes , writtenBytes

**Fields inherited from org.red5.server.net.rtmp.RTMPConnection:** clientId  
 , deferredResults , encoding , invokeId , keepAliveJobName , lastPingSent ,  
 lastPingTime , lastPongReceived , log , maxInactivity , oName , pendingCalls ,  
 pingInterval , state , streamBuffers

**Fields inherited from org.red5.server.BaseConnection:** basicScopes , client ,  
 closed , droppedMessages , host , params , path , readMessages , remoteAddress ,  
 remoteAddresses , remotePort , scope , sessionId , type , writtenMessages

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 4.2. INCREASE\_POLLING\_DELAY\_COUNT

```
protected static final long INCREASE_POLLING_DELAY_COUNT = 10L;
```

Start to increase the polling delay after this many empty results

## 4.3. INITIAL\_POLLING\_DELAY

```
protected static final byte INITIAL_POLLING_DELAY = 0;
```

Polling delay to start with.

## 4.4. MAX\_POLLING\_DELAY

```
protected static final byte MAX_POLLING_DELAY = 32;
```

Maximum polling delay.

## 4.5. noPendingMessages

```
protected volatile long noPendingMessages ;
```

Empty result counter, after reaching INCREASE\_POLLING\_DELAY\_COUNT polling delay  
 will increase

## 4.6. pollingDelay

```
protected volatile byte pollingDelay ;
```

Polling delay value

## 4.7. servlet

```
protected RTMPTServlet servlet ;
```

Servlet that created this connection.

## 4.8. getPendingMessages(int)

```
public org.apache.mina.common.ByteBuffer getPendingMessages(int targetSize);
```

## 4.9. getPollingDelay()

```
public byte getPollingDelay();
```

Return the polling delay to use.

### Parameters

return	the polling delay
--------	-------------------

## 4.10. onInactive()

```
protected void onInactive();
```

## 4.11. realClose()

```
public void realClose();
```

## 4.12. setServlet(RTMPTServlet)

```
protected void setServlet(RTMPTServlet servlet);
```

Set the servlet that created the connection.

### Parameters

servlet
---------

## 4.13. setServletRequest(HttpServletRequest)

```
public void setServletRequest(javax.servlet.http.HttpServletRequest request);
```

Setter for servlet request.

### Parameters

request	Servlet request
---------	-----------------

## 5. Class RTMPTHandler

Handler for RTMPT messages.

### 5.1. Synopsis

```
public class RTMPTHandler extends org.?red5.?server.?net.?rtmpt.?RTMPTHandler {
    // Public Static Fields

    public static final String HANDLER_ATTRIBUTE = "red5.RMPTHandler";

    // Protected Fields

    protected org.red5.server.net.protocol.SimpleProtocolCodecFactory codecFactory;

    // Public Constructors

    public RTMPTHandler();

    // Public Methods

    public org.red5.server.net.protocol.SimpleProtocolCodecFactory getCodecFactory();

    public void messageReceived(org.red5.server.net.rtmp.RTMPConnection conn,
                               org.red5.server.net.protocol.ProtocolState state,
                               Object in)
            throws Exception;

    public void setCodecFactory(org.red5.server.net.protocol.SimpleProtocolCodecFactory factory);

}
```

**Methods inherited from org.red5.server.net.rtmp.RTMPTHandler:** getStatus , invokeCall , onChunkSize , onInvoke , onPing , onSharedObject , setServer , setStatusObjectService

**Methods inherited from org.red5.server.net.rtmp.BaseRTMPTHandler:**

connectionClosed , connectionOpened , getHostname , getStreamId , handlePendingCallResult , messageReceived , messageSent , onStreamBytesRead , setApplicationContext

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.RTMPTHandler:** log , server , statusObjectService

**Fields inherited from org.red5.server.net.rtmp.BaseRTMPTHandler:** appCtx

### 5.2. codecFactory

```
protected org.red5.server.net.protocol.SimpleProtocolCodecFactory codecFactory;
```

Protocol codec factory

## 5.3. HANDLER\_ATTRIBUTE

```
public static final String HANDLER_ATTRIBUTE = "red5.RMPTHandler";
```

Handler constant

## 5.4. getCodecFactory()

```
public org.red5.server.net.protocol.SimpleProtocolCodecFactory getCodecFactory();
```

Getter for codec factory

Parameters

<i>return</i>	Codec factory
---------------	---------------

## 5.5. messageReceived(RTMPConnection, ProtocolState, Object)

```
public void messageReceived(org.red5.server.net.rtmp.RTMPConnection conn,
                           org.red5.server.net.protocol.ProtocolState state,
                           Object in)
                           throws Exception;
```

## 5.6. setCodecFactory(SimpleProtocolCodecFactory)

```
public void setCodecFactory(org.red5.server.net.protocol.SimpleProtocolCodecFactory factory);
```

Setter for codec factory

Parameters

<i>factory</i>	Codec factory to use
----------------	----------------------

# 6. Class RTMPTLoader

Loader for the RTMPT server.

## 6.1. Synopsis

```
public class RTMPTLoader implements org.springframework.context.ApplicationContextAware {
// Protected Fields

    protected org.springframework.context.ApplicationContext applicationContext;

    protected static org.slf4j.Logger log;

    protected String rtmpConfig;

    protected org.mortbay.jetty.Server rtmpServer;

    protected org.red5.server.api.IServer server;

// Public Constructors

    public RTMPTLoader();
```

```
// Public Methods

    public void init()
        throws Exception;

    public void setApplicationContext(org.springframework.context.ApplicationContext context)
        throws BeansException;

    public void setServer(org.red5.server.api.IServer server);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 6.2. applicationContext

```
protected org.springframework.context.ApplicationContext applicationContext ;
```

Application context

## 6.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 6.4. rtmptConfig

```
protected String rtmptConfig ;
```

RTMP config path

## 6.5. rtmptServer

```
protected org.mortbay.jetty.Server rtmptServer ;
```

RTMP server

## 6.6. server

```
protected org.red5.server.api.IServer server ;
```

Red5 server instance

## 6.7. init()

```
public void init()
    throws Exception;
```

Initialization

Exception

Exception

## 6.8. setApplicationContext(ApplicationContext)

```
public void setApplicationContext(org.springframework.context.ApplicationContext context)
    throws BeansException;
```

**Specified by:** Method `setApplicationContext` in interface `ApplicationContextAware`

## 6.9. setServer(I Server)

```
public void setServer(org.red5.server.api.I Server server);
```

Setter for server

### Parameters

server	Server instance
--------	-----------------

## 7. Class RTMPTServlet

Servlet that handles all RTMPT requests.

### 7.1. Synopsis

```
public class RTMPTServlet extends javax.servlet.http.HttpServlet {
// Protected Fields

    protected org.springframework.web.context.WebApplicationContext appCtx ;
    protected static org.slf4j.Logger log ;

// Public Constructors

    public RTMPTServlet();

// Public Methods

    public void destroy();

    public void init()
        throws ServletException;

    public void setHandler(RTMPTHandler handler);

    public void setRtmpConnManager(org.red5.server.net.rtmp.IRTMPConnManager rtmpConnManager);

// Protected Methods

    protected RTMPTConnection createConnection();

    protected RTMPTConnection getClient(javax.servlet.http.HttpServletRequest req);

    protected Integer getClientId(javax.servlet.http.HttpServletRequest req);

    protected RTMPTConnection getConnection(int clientId);

    protected void handleBadRequest(String message,
                                    javax.servlet.http.HttpServletResponse resp)
```

```

    throws IOException;

    protected void handleClose(javax.servlet.http.HttpServletRequest req,
                             javax.servlet.http.HttpServletResponse resp)
    throws ServletException, IOException;

    protected void handleIdle(javax.servlet.http.HttpServletRequest req,
                             javax.servlet.http.HttpServletResponse resp)
    throws ServletException, IOException;

    protected void handleOpen(javax.servlet.http.HttpServletRequest req,
                            javax.servlet.http.HttpServletResponse resp)
    throws ServletException, IOException;

    protected void handleSend(javax.servlet.http.HttpServletRequest req,
                            javax.servlet.http.HttpServletResponse resp)
    throws ServletException, IOException;

    protected void notifyClosed(RTMPTConnection conn);

    protected void removeConnection(int clientId);

    protected void returnMessage(byte message,
                                javax.servlet.http.HttpServletResponse resp)
    throws IOException;

    protected void returnMessage(String message,
                                javax.servlet.http.HttpServletResponse resp)
    throws IOException;

    protected void returnMessage(RTMPTConnection client,
                                org.apache.mina.common.ByteBuffer buffer,
                                javax.servlet.http.HttpServletResponse resp)
    throws IOException;

    protected void returnPendingMessages(RTMPTConnection client,
                                        javax.servlet.http.HttpServletResponse resp)
    throws IOException;

    protected void service(javax.servlet.http.HttpServletRequest req,
                          javax.servlet.http.HttpServletResponse resp)
    throws ServletException, IOException;

    protected void skipData(javax.servlet.http.HttpServletRequest req)
    throws IOException;
}

}

```

**Methods inherited from javax.servlet.http.HttpServlet:** doDelete , doGet , doHead , doOptions , doPost , doPut , doTrace , getLastModified , service

**Methods inherited from javax.servlet.GenericServlet:** destroy , getInitParameter , getInitParameterNames , getServletConfig , getServletContext , getServletInfo , getServletName , init , log

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 7.2. appCtx

```
protected org.springframework.web.context.WebApplicationContext appCtx ;
```

Web app context

## 7.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 7.4. destroy()

```
public void destroy();
```

## 7.5. getClient(HttpServletRequest)

```
protected RTMPTConnection getClient(javax.servlet.http.HttpServletRequest req);
```

Get the RTMPT client for a session.

### Parameters

req	Servlet request
<i>return</i>	RTMP client connection

## 7.6. getClientId(HttpServletRequest)

```
protected Integer getClientId(javax.servlet.http.HttpServletRequest req);
```

Return the client id from a url like /send/123456/12 -> 123456

### Parameters

req	Servlet request
<i>return</i>	Client id

## 7.7. handleBadRequest(String, HttpServletResponse)

```
protected void handleBadRequest(String message,
                               javax.servlet.http.HttpServletResponse resp)
                               throws IOException;
```

Return an error message to the client.

### Parameters

message	Message
<i>resp</i>	Servlet response

IOException  
I/O exception

## 7.8. handleClose(HttpServletRequest, HttpServletResponse)

```
protected void handleClose(javax.servlet.http.HttpServletRequest req,
                         javax.servlet.http.HttpServletResponse resp)
    throws ServletException, IOException;
```

Close a RTMPT session.

### Parameters

req	Servlet request
resp	Servlet response

ServletException  
Servlet exception  
  
IOException  
I/O exception

## 7.9. handleIdle(HttpServletRequest, HttpServletResponse)

```
protected void handleIdle(javax.servlet.http.HttpServletRequest req,
                         javax.servlet.http.HttpServletResponse resp)
    throws ServletException, IOException;
```

Poll RTMPT session for updates.

### Parameters

req	Servlet request
resp	Servlet response

ServletException  
Servlet exception  
  
IOException  
I/O exception

## 7.10. handleOpen(HttpServletRequest, HttpServletResponse)

```
protected void handleOpen(javax.servlet.http.HttpServletRequest req,
                         javax.servlet.http.HttpServletResponse resp)
    throws ServletException, IOException;
```

Start a new RTMPT session.

### Parameters

req	Servlet request
resp	Servlet response

ServletException  
Servlet exception

IOException  
I/O exception

## 7.11. handleSend(HttpServletRequest, HttpServletResponse)

```
protected void handleSend(javax.servlet.http.HttpServletRequest req,
                         javax.servlet.http.HttpServletResponse resp)
                         throws ServletException, IOException;
```

Add data for an established session.

### Parameters

req	Servlet request
resp	Servlet response

ServletException  
Servlet exception

IOException  
I/O exception

## 7.12. init()

```
public void init()
                  throws ServletException;
```

## 7.13. notifyClosed(RTMPTConnection)

```
protected void notifyClosed(RTMPTConnection conn);
```

A connection has been closed that was created by this servlet.

### Parameters

conn
------

## 7.14. returnMessage(byte, HttpServletResponse)

```
protected void returnMessage(byte message,
                            javax.servlet.http.HttpServletResponse resp)
                            throws IOException;
```

Return a single byte to the client.

### Parameters

message	Message
resp	Servlet response

IOException  
I/O exception

## 7.15. returnMessage(RTMPTConnection, ByteBuffer, HttpServletResponse)

```
protected void returnMessage(RTMPTConnection client,
                            org.apache.mina.common.ByteBuffer buffer,
                            javax.servlet.http.HttpServletResponse resp)
throws IOException;
```

Return raw data to the client.

### Parameters

client	RTMP connection
buffer	Raw data as byte buffer
resp	Servlet response

IOException  
I/O exception

## 7.16. returnMessage(String, HttpServletResponse)

```
protected void returnMessage(String message,
                            javax.servlet.http.HttpServletResponse resp)
throws IOException;
```

Return a message to the client.

### Parameters

message	Message
resp	Servlet response

IOException  
I/O exception

## 7.17. returnPendingMessages(RTMPTConnection, HttpServletResponse)

```
protected void returnPendingMessages(RTMPTConnection client,
                                    javax.servlet.http.HttpServletResponse resp)
throws IOException;
```

Send pending messages to client.

### Parameters

client	RTMP connection
resp	Servlet response

IOException  
I/O exception

## 7.18. service(HttpServletRequest, HttpServletResponse)

```
protected void service(javax.servlet.http.HttpServletRequest req,
                      javax.servlet.http.HttpServletResponse resp)
                    throws ServletException, IOException;
```

Main entry point for the servlet.

### Parameters

req	Request object
resp	Response object

## 7.19. setHandler(RTMPTHandler)

```
public void setHandler(RTMPTHandler handler);
```

Set the RTMPTHandler to use in this servlet.

### Parameters

handler	
---------	--

## 7.20. skipData(HttpServletRequest)

```
protected void skipData(javax.servlet.http.HttpServletRequest req)
                        throws IOException;
```

Skip data sent by the client.

### Parameters

req	Servlet request
-----	-----------------

IOException

I/O exception

# 8. Class TomcatRTMPTLoader

Loader for the RTMPT server which uses Tomcat.

## 8.1. Synopsis

```
public class TomcatRTMPTLoader extends org.?red5.?server.?tomcat.?TomcatLoader {
// Protected Fields

    protected org.apache.catalina.Context context ;

    protected org.apache.catalina.Engine rtmptEngine ;

    protected org.red5.server.api.IServer server ;

    protected java.util.Map<java.lang.String, java.lang.String> servletMappings ;
```

```
// Public Constructors

    public TomcatRTMPTLoader();

// Public Methods

    public void init();

    public void setMappings(java.util.Map<java.lang.String, java.lang.String> mappings);

    public void setServer(org.red5.server.api.IServer server);

}
```

**Direct known subclasses:** org.?red5.?server.?net.?rtmps.?TomcatRTMPSTLoader

**Methods inherited from org.red5.server.tomcat.TomcatLoader:** addContext , formatPath , getBaseHost , getConnector , getEmbedded , getEngine , getHost , getRealm , init , registerJMX , removeContext , setBaseHost , setConnectionProperties , setConnector , setConnectors , setContexts , setEmbedded , setHost , setHosts , setRealm , setValves , shutdown

**Methods inherited from org.red5.server.LoaderBase:** getApplicationContext , getApplicationLoader , getRed5ApplicationContext , removeRed5ApplicationContext , setApplicationContext , setApplicationLoader , setRed5ApplicationContext , setWebappFolder

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.tomcat.TomcatLoader:** connectionProperties , connector , defaultParentContextKey , defaultSpringConfigLocation , embedded , engine , host , realm , valves

**Fields inherited from org.red5.server.LoaderBase:** applicationContext , loader , red5AppCtx , webappFolder

## 8.2. context

```
protected org.apache.catalina.Context context ;
```

Context, in terms of JEE context is web application in a servlet container

## 8.3. rtmptEngine

```
protected org.apache.catalina.Engine rtmptEngine ;
```

RTMPT Tomcat engine.

## 8.4. server

```
protected org.red5.server.api.IServer server ;
```

Server instance

## 8.5. servletMappings

```
protected java.util.Map<java.lang.String, java.lang.String> servletMappings ;
```

Extra servlet mappings to add

## 8.6. init()

```
public void init();
```

## 8.7. setMappings(Map<String, String>)

```
public void setMappings(java.util.Map<java.lang.String, java.lang.String> mappings);
```

Set servlet mappings

### Parameters

mappings
----------

## 8.8. setServer(I Server)

```
public void setServer(org.red5.server.api.I Server server);
```

Setter for server

### Parameters

server	Value to set for property 'server'.
--------	-------------------------------------

# 1. Class RTMPTCodecFactory

RTMP codec factory creates RTMP codec objects

## 1.1. Synopsis

```
public class RTMPTCodecFactory implements org.red5.server.net.protocol.SimpleProtocolCodecFactory
// Protected Fields

protected RTMPTProtocolDecoder decoder ;

protected org.red5.io.object.Deserializer deserializer ;

protected RTMPTProtocolEncoder encoder ;

protected org.red5.io.object.Serializer serializer ;

// Public Constructors

public RTMPTCodecFactory();

// Public Methods

public org.red5.server.net.protocol.SimpleProtocolDecoder getSimpleDecoder();

public org.red5.server.net.protocol.SimpleProtocolEncoder getSimpleEncoder();

public void init();

public void setDeserializer(org.red5.io.object.Deserializer deserializer);

public void setSerializer(org.red5.io.object.Serializer serializer);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. decoder

```
protected RTMPTProtocolDecoder decoder ;
```

RTMP decoder

## 1.3. deserializer

```
protected org.red5.io.object.Deserializer deserializer ;
```

Deserializer

## 1.4. encoder

```
protected RTMPTProtocolEncoder encoder ;
```

RTMP encoder

## 1.5. serializer

```
protected org.red5.io.object.Serializer serializer ;
```

Serializer

## 1.6. getSimpleDecoder()

```
public org.red5.server.net.protocol.SimpleProtocolDecoder getSimpleDecoder();
```

**Specified by:** Method getSimpleDecoder in interface SimpleProtocolCodecFactory

Getter for simple decoder.

## 1.7. getSimpleEncoder()

```
public org.red5.server.net.protocol.SimpleProtocolEncoder getSimpleEncoder();
```

**Specified by:** Method getSimpleEncoder in interface SimpleProtocolCodecFactory

Getter for simple encoder.

## 1.8. init()

```
public void init();
```

Initialization

## 1.9. setDeserializer(Deserializer)

```
public void setDeserializer(org.red5.io.object.Deserializer deserializer);
```

Setter for deserializer.

**Parameters**

deserializer	Deserializer used by this codec factory.
--------------	--

## 1.10. setSerializer(Serializer)

```
public void setSerializer(org.red5.io.object.Serializer serializer);
```

Setter for serializer

**Parameters**

serializer	Value to set for property 'serializer'.
------------	---

## 2. Class RTMPTProtocolDecoder

RTMPT protocol decoder. To be implemented.

## 2.1. Synopsis

```
public class RTMPTProtocolDecoder extends org.red5.server.net.rtmp.codec.RTMPPProtocolDecoder {
    // Public Constructors

    public RTMPTProtocolDecoder();
}

}
```

**Methods inherited from org.red5.server.net.rtmp.codec.RTMPPProtocolDecoder:**

decode , decodeAudioData , decodeBuffer , decodeBytesRead , decodeChunkSize ,  
decodeFlexMessage , decodeFlexSharedObject , decodeFlexStreamSend , decodeHandshake  
, decodeHeader , decodeInvoke , decodeMessage , decodeNotify , decodeNotifyOrInvoke  
, decodePacket , decodePing , decodeSharedObject , decodeStreamMetadata ,  
decodeUnknown , decodeVideoData , doDecodeSharedObject , setDeserializer ,  
setupClassLoader

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.codec.RTMPPProtocolDecoder:** ioLog ,  
log

## 3. Class RTMPTProtocolEncoder

RTMPT protocol encoder. To be implemented.

### 3.1. Synopsis

```
public class RTMPTProtocolEncoder extends org.red5.server.net.rtmp.codec.RTMPPProtocolEncoder {
    // Public Constructors

    public RTMPTProtocolEncoder();
}

}
```

**Methods inherited from org.red5.server.net.rtmp.codec.RTMPPProtocolEncoder:**

doEncodeSharedObject , encode , encodeAudioData , encodeBytesRead , encodeChunkSize  
, encodeFlexMessage , encodeFlexSharedObject , encodeFlexStreamSend , encodeHeader  
, encodeInvoke , encodeMessage , encodeNotify , encodeNotifyOrInvoke , encodePacket  
, encodePing , encodeSharedObject , encodeStreamMetadata , encodeUnknown ,  
encodeVideoData , setSerializer

**Methods inherited from org.red5.server.net.protocol.BaseProtocolEncoder:**

generateErrorResult

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.codec.RTMPPProtocolEncoder:** ioLog ,  
log

# 1. Class AMFGatewayServlet

Servlet that handles remoting requests.

## 1.1. Synopsis

```
public class AMFGatewayServlet extends javax.servlet.http.HttpServlet {  
    // Public Static Fields  
  
    public static final String APPLICATION_AMF = "application/x-amf";  
  
    // Protected Fields  
  
    protected org.red5.server.net.remoting.codec.RemotingCodecFactory codecFactory;  
  
    protected static org.slf4j.Logger log;  
  
    protected org.red5.server.api.IServer server;  
  
    protected org.springframework.web.context.WebApplicationContext webAppCtx;  
  
    // Public Constructors  
  
    public AMFGatewayServlet();  
  
    // Public Methods  
  
    public void init()  
        throws ServletException;  
  
    public void service(javax.servlet.http.HttpServletRequest req,  
                      javax.servlet.http.HttpServletResponse resp)  
        throws ServletException, IOException;  
  
    // Protected Methods  
  
    protected org.red5.server.net.remoting.message.RemotingPacket decodeRequest(javax.servlet.http.Http  
        throws Exception;  
  
    protected org.red5.server.api.IGlobalScope getGlobalScope(javax.servlet.http.HttpServletRequest re  
  
    protected boolean handleRemotingPacket(javax.servlet.http.HttpServletRequest req,  
                                           org.red5.server.api.IContext context,  
                                           org.red5.server.api.IScope scope,  
                                           org.red5.server.net.remoting.message.RemotingPacket message);  
  
    protected void sendResponse(javax.servlet.http.HttpServletResponse resp,  
                               org.red5.server.net.remoting.message.RemotingPacket packet)  
        throws Exception;  
  
    protected void serviceAMF(javax.servlet.http.HttpServletRequest req,  
                            javax.servlet.http.HttpServletResponse resp)  
        throws ServletException, IOException;  
}
```

**Methods inherited from javax.servlet.http.HttpServlet:** doDelete , doGet , doHead ,  
doOptions , doPost , doPut , doTrace , getLastModified , service

**Methods inherited from javax.servlet.GenericServlet:** destroy , getInitParameter , getInitParameterNames , getServletConfig , getServletContext , getServletInfo , getServletName , init , log

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 1.2. APPLICATION\_AMF

```
public static final String APPLICATION_AMF = "application/x-amf";
```

AMF MIME type

## 1.3. codecFactory

```
protected org.red5.server.net.remoting.codec.RemotingCodecFactory codecFactory ;
```

Remoting codec factory

## 1.4. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 1.5. server

```
protected org.red5.server.api.IServer server ;
```

Red5 server instance

## 1.6. webAppCtx

```
protected org.springframework.web.context.WebApplicationContext webAppCtx ;
```

Web app context

## 1.7. decodeRequest(HttpServletRequest)

```
protected org.red5.server.net.remoting.message.RemotingPacket decodeRequest(javax.servlet.http.HttpServletRequest
throws Exception;
```

Decode request

### Parameters

req	Request
return	Remoting packet

Exception

General exception

## 1.8. getGlobalScope(HttpServletRequest)

```
protected org.red5.server.api.IGlobalScope getGlobalScope(javax.servlet.http.HttpServletRequest req)
```

Return the global scope to use for the given request.

### Parameters

req	
<i>return</i>	

## 1.9. handleRemotingPacket(HttpServletRequest, IContext, IScope, RemotingPacket)

```
protected boolean handleRemotingPacket(javax.servlet.http.HttpServletRequest req,
                                      org.red5.server.api.IContext context,
                                      org.red5.server.api.IScope scope,
                                      org.red5.server.net.remoting.message.RemotingPacket message);
```

Handles AMF request by making calls

### Parameters

req	Request
message	Remoting packet
<i>return</i>	true on success

## 1.10. init()

```
public void init()
    throws ServletException;
```

## 1.11. sendResponse(HttpServletRequest, RemotingPacket)

```
protected void sendResponse(javax.servlet.http.HttpServletResponse resp,
                           org.red5.server.net.remoting.message.RemotingPacket packet)
    throws Exception;
```

Sends response to client

### Parameters

resp	Response
packet	Remoting packet

packet	Remoting packet
--------	-----------------

Exception

General exception

## 1.12. service(HttpServletRequest, HttpServletResponse)

```
public void service(javax.servlet.http.HttpServletRequest req,
```

```
javax.servlet.http.HttpServletResponse resp)
throws ServletException, IOException;
```

## 1.13. serviceAMF(HttpServletRequest, HttpServletResponse)

```
protected void serviceAMF(javax.servlet.http.HttpServletRequest req,
                          javax.servlet.http.HttpServletResponse resp)
throws ServletException, IOException;
```

Works out AMF request

### Parameters

req	Request
resp	Response

ServletException

Servlet exception

IOException

I/O exception

## 2. Class AMFTunnelServlet

Servlet to tunnel to the AMF gateway servlet.

### 2.1. Synopsis

```
public class AMFTunnelServlet extends javax.servlet.http.HttpServlet {
// Protected Fields

    protected static org.slf4j.Logger logger ;

// Public Constructors

    public AMFTunnelServlet();

// Public Methods

    public void init(javax.servlet.ServletConfig config)
        throws ServletException;

// Protected Methods

    protected void service(javax.servlet.http.HttpServletRequest req,
                          javax.servlet.http.HttpServletResponse resp)
        throws ServletException, IOException;

}
```

**Methods inherited from javax.servlet.http.HttpServlet:** doDelete , doGet , doHead , doOptions , doPost , doPut , doTrace , getLastModified , service

**Methods inherited from javax.servlet.GenericServlet:** destroy , getInitParameter , getInitParameterNames , getServletConfig , getServletContext , getServletInfo , getServletName , init , log

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. service(HttpServletRequest, HttpServletResponse)

```
protected void service(javax.servlet.http.HttpServletRequest req,
                      javax.servlet.http.HttpServletResponse resp)
throws ServletException, IOException;
```

Redirect to HTTP port.

## 3. Class CaptureViewerServlet

```
public class CaptureViewerServlet extends javax.servlet.http.HttpServlet {
// Public Constructors

    public CaptureViewerServlet();

// Public Methods

    public String formatHTML(org.red5.server.net.rtmp.message.Packet packet,
                           int id,
                           long time);

    public void init()
    throws ServletException;

// Protected Methods

    protected void service(javax.servlet.http.HttpServletRequest req,
                          javax.servlet.http.HttpServletResponse resp)
    throws ServletException;

}
```

**Methods inherited from javax.servlet.http.HttpServlet:** doDelete, doGet, doHead, doOptions, doPost, doPut, doTrace, getLastModified, service

**Methods inherited from javax.servlet.GenericServlet:** destroy, getInitParameter, getInitParameterNames, getServletConfig, getServletContext, getServletInfo, getServletName, init, log

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 3.1. formatHTML(Packet, int, long)

```
public String formatHTML(org.red5.server.net.rtmp.message.Packet packet,
                        int id,
                        long time);
```

Formats HTML

Parameters

packet	RTMP packet
id	id
time	Time
return	Formatted packet representation in HTML

### 3.2. init()

```
public void init()
    throws ServletException;
```

### 3.3. service(HttpServletRequest, HttpServletResponse)

```
protected void service(javax.servlet.http.HttpServletRequest req,
                      javax.servlet.http.HttpServletResponse resp)
    throws ServletException;
```

Writes HTML out of dump

## 4. Class RedirectHTTPServlet

Servlet to redirect to HTTP port of Red5.

### 4.1. Synopsis

```
public class RedirectHTTPServlet extends javax.servlet.http.HttpServlet {
// Public Constructors

    public RedirectHTTPServlet();

// Protected Methods

    protected void service(javax.servlet.http.HttpServletRequest req,
                          javax.servlet.http.HttpServletResponse resp)
        throws ServletException, IOException;

}
```

**Methods inherited from javax.servlet.http.HttpServlet:** doDelete , doGet , doHead , doOptions , doPost , doPut , doTrace , getLastModified , service

**Methods inherited from javax.servlet.GenericServlet:** destroy , getInitParameter , getInitParameterNames , getServletConfig , getServletContext , getServletInfo , getServletName , init , log

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

### 4.2. service(HttpServletRequest, HttpServletResponse)

```
protected void service(javax.servlet.http.HttpServletRequest req,
                      javax.servlet.http.HttpServletResponse resp)
    throws ServletException, IOException;
```

Redirect to HTTP port.

## 5. Class RequestDumpServlet

Servlet that dumps request data

### 5.1. Synopsis

```
public class RequestDumpServlet extends javax.?servlet.?http.?HttpServlet {
// Public Static Fields

    public static final String APPLICATION_AMF = "application/x-amf";

// Protected Fields

    protected static org.slf4j.Logger log;

// Public Constructors

    public RequestDumpServlet();

// Protected Methods

    protected void service(javax.servlet.http.HttpServletRequest req,
                          javax.servlet.http.HttpServletResponse resp)
        throws ServletException, IOException;

}
```

**Methods inherited from javax.servlet.http.HttpServlet:** doDelete , doGet , doHead , doOptions , doPost , doPut , doTrace , getLastModified , service

**Methods inherited from javax.servlet.GenericServlet:** destroy , getInitParameter , getInitParameterNames , getServletConfig , getServletContext , getServletInfo , getServletName , init , log

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 5.2. APPLICATION\_AMF

```
public static final String APPLICATION_AMF = "application/x-amf";
```

AMF MIME type

### 5.3. log

```
protected static org.slf4j.Logger log;
```

Logger

## 5.4. service(HttpServletRequest, HttpServletResponse)

```
protected void service(javax.servlet.http.HttpServletRequest req,
```

```
    javax.servlet.http.HttpServletResponse resp)
throws ServletException, IOException;
```

## 6. Class ServletUtils

```
public class ServletUtils {
// Public Static Fields

    public static final int DEFAULT_BUFFER_SIZE = 2048;

// Public Constructors

    public ServletUtils();

// Public Static Methods

    public static void copy(java.io.InputStream input,
                          java.io.OutputStream output)
    throws IOException;

    public static void copy(java.io.InputStream input,
                          java.io.OutputStream output,
                          int bufferSize)
    throws IOException;

    public static void copyThenClose(java.io.InputStream input,
                                    java.io.OutputStream output)
    throws IOException;

    public static byte[] getBytes(java.io.InputStream input)
    throws IOException;

    public static java.util.List<java.lang.String> getRemoteAddresses(javax.servlet.http.HttpServletResponse resp)
    throws IOException;
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 6.1. DEFAULT\_BUFFER\_SIZE

```
public static final int DEFAULT_BUFFER_SIZE = 2048;
```

Default value is 2048.

### 6.2. copy(InputStream, OutputStream)

```
public static void copy(java.io.InputStream input,
                      java.io.OutputStream output)
    throws IOException;
```

Copies information from the input stream to the output stream using a default buffer size of 2048 bytes.

Parameters
------------

input	
output	

java.io.IOException  
java.io.IOException

### 6.3. copy(InputStream, OutputStream, int)

```
public static void copy(java.io.InputStream input,
                      java.io.OutputStream output,
                      int bufferSize)
throws IOException;
```

Copies information from the input stream to the output stream using the specified buffer size

Parameters	
input	
bufferSize	
output	

java.io.IOException  
java.io.IOException

### 6.4. copyThenClose(InputStream, OutputStream)

```
public static void copyThenClose(java.io.InputStream input,
                                 java.io.OutputStream output)
throws IOException;
```

Copies information between specified streams and then closes both of the streams.

Parameters	
output	
input	

java.io.IOException  
java.io.IOException

### 6.5. getBytes(InputStream)

```
public static byte[] getBytes(java.io.InputStream input)
throws IOException;
```

Parameters	
input	
return	

```
java.io.IOException
    java.io.IOException
```

**Returns**

a byte[] containing the information contained in the specified InputStream.

## 6.6. getRemoteAddresses(HttpServletRequest)

```
public static java.util.List<java.lang.String> getRemoteAddresses(javax.servlet.http.HttpServletRequest)
```

Return all remote addresses that were involved in the passed request.

### Parameters

request	
<i>return</i>	

## 7. Class StatisticsServlet

Servlet that processes the statistics XML-RPC requests.

### 7.1. Synopsis

```
public class StatisticsServlet extends javax.servlet.http.HttpServlet {
    // Protected Fields

    protected org.springframework.web.context.WebApplicationContext webAppCtx;

    protected org.red5.server.api.IContext webContext;

    // Public Constructors

    public StatisticsServlet();

    // Public Methods

    public void doPost(javax.servlet.http.HttpServletRequest request,
                      javax.servlet.http.HttpServletResponse response)
        throws ServletException, IOException;

    public void init()
        throws ServletException;

}
```

**Methods inherited from javax.servlet.http.HttpServlet:** doDelete , doGet , doHead , doOptions , doPost , doPut , doTrace , getLastModified , service

**Methods inherited from javax.servlet.GenericServlet:** destroy , getInitParameter , getInitParameterNames , getServletConfig , getServletContext , getServletInfo , getServletName , init , log

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 7.2. doPost(HttpServletRequest, HttpServletResponse)

```
public void doPost(javax.servlet.http.HttpServletRequest request,
                   javax.servlet.http.HttpServletResponse response)
                     throws ServletException, IOException;
```

## 7.3. init()

```
public void init()
  throws ServletException;
```

# 8. Class ZAMFGatewayServlet

```
public class ZAMFGatewayServlet extends javax.servlet.http.HttpServlet {
// Public Static Fields

  public static final String APPLICATION_AMF = "application/x-amf";

// Protected Fields

  protected static org.slf4j.Logger log;

// Public Constructors

  public ZAMFGatewayServlet();

// Protected Methods

  protected void service(javax.servlet.http.HttpServletRequest req,
                        javax.servlet.http.HttpServletResponse resp)
                        throws ServletException, IOException;

}
```

**Methods inherited from javax.servlet.http.HttpServlet:** doDelete , doGet , doHead , doOptions , doPost , doPut , doTrace , getLastModified , service

**Methods inherited from javax.servlet.GenericServlet:** destroy , getInitParameter , getInitParameterNames , getServletConfig , getServletContext , getServletInfo , getServletName , init , log

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 8.1. service(HttpServletRequest, HttpServletResponse)

```
protected void service(javax.servlet.http.HttpServletRequest req,
                      javax.servlet.http.HttpServletResponse resp)
                      throws ServletException, IOException;
```

# 9. Class ZAMFGatewayServlet.Handler

```
protected class ZAMFGatewayServlet.Handler extends org.apache.mina.common.IoHandlerAdapter {
```

```
// Protected Fields

protected javax.servlet.http.HttpServletRequest req;

protected javax.servlet.http.HttpServletResponse resp;

// Public Constructors

public ZAMFGatewayServlet.Handler(javax.servlet.http.HttpServletRequest req,
                                     javax.servlet.http.HttpServletResponse resp);

// Public Methods

public void messageReceived(org.apache.mina.common.IoSession session,
                           Object message)
throws Exception;

}
```

**Methods inherited from org.apache.mina.common.IoHandlerAdapter:**

exceptionCaught , messageReceived , messageSent , sessionClosed , sessionCreated ,  
sessionId , sessionOpened

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 9.1. messageReceived(IoSession, Object)

```
public void messageReceived(org.apache.mina.common.IoSession session,
                           Object message)
throws Exception;
```

# 1. Class BasicHandler

=> client send to server <= server send to client << server broadcast Connecting to the server => byte(join) << byte(join) int(id) => byte(list) <= byte(list) int(count) int(id) int(id) ... Sending a message to all => byte(send) [..anything..] << byte(send) [..anything..] Server ping client to keep alive, every second <= byte(noop) => byte(noop) Timeouts (after 10s no reply) << byte(exit) int(id)

## 1.1. Synopsis

```
public class BasicHandler extends org.apache.mina.common.IoHandlerAdapter {  
    // Protected Fields  
  
    protected static org.slf4j.Logger log ;  
  
    protected java.util.Set<org.apache.mina.common.IoSession> sessions ;  
  
    protected boolean showInfo ;  
  
    protected java.util.Timer timer ;  
  
    // Public Constructors  
  
    public BasicHandler();  
  
    // Public Methods  
  
    public void exceptionCaught(org.apache.mina.common.IoSession session,  
                                Throwable ex)  
        throws Exception;  
  
    public void messageReceived(org.apache.mina.common.IoSession session,  
                               Object message)  
        throws Exception;  
  
    public void sessionCreated(org.apache.mina.common.IoSession session)  
        throws Exception;  
  
    // Protected Methods  
  
    protected void broadcast(org.apache.mina.common.IoSession exclude,  
                           org.apache.mina.common.ByteBuffer data);  
  
    protected void echo(org.apache.mina.common.IoSession session,  
                      org.apache.mina.common.ByteBuffer data);  
  
    protected void join(org.apache.mina.common.IoSession session);  
  
    protected void leave(org.apache.mina.common.IoSession session);  
  
    protected void list(org.apache.mina.common.IoSession to);  
}
```

### Methods inherited from org.apache.mina.common.IoHandlerAdapter:

exceptionCaught , messageReceived , messageSent , sessionClosed , sessionCreated ,  
sessionIdle , sessionOpened

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 2. Class BasicHandler.TimeoutTask

```
protected class BasicHandler.TimeoutTask extends, java.util.TimerTask {
// Protected Constructors

    protected BasicHandler.TimeoutTask();

// Public Methods

    public void run();

}
```

**Methods inherited from java.util.TimerTask:** cancel , run , scheduledExecutionTime

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 3. Class Standalone

```
public class Standalone {
// Public Static Fields

    public static final int PORT = 5150;

// Public Constructors

    public Standalone();

// Public Static Methods

    public static void main(String[] args)
        throws Exception;

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

# 1. Class FilePersistence

Simple file-based persistence for objects. Lowers memory usage if used instead of RAM memory storage.

## 1.1. Synopsis

```
public class FilePersistence extends org.red5.server.persistence.RamPersistence {  
    // Public Constructors  
  
    public FilePersistence(org.red5.server.api.IScope scope);  
  
    public FilePersistence(org.springframework.core.io.support.ResourcePatternResolver resolver);  
  
    // Public Methods  
  
    public org.red5.server.api.persistence.IPersistable load(String name);  
  
    public boolean load(org.red5.server.api.persistence.IPersistable object);  
  
    public void notifyClose();  
  
    public boolean remove(String name);  
  
    public boolean remove(org.red5.server.api.persistence.IPersistable object);  
  
    public boolean save(org.red5.server.api.persistence.IPersistable object);  
  
    public void setExtension(String extension);  
  
    public void setPath(String path);  
  
    // Protected Methods  
  
    protected void checkRemoveEmptyDirectories(String base);  
  
    protected String getObjectPath(String id,  
                                  String name);  
  
    protected boolean saveObject(org.red5.server.api.persistence.IPersistable object);  
}
```

**Methods inherited from org.red5.server.persistence.RamPersistence:** getId ,  
getObjectName , getNames , getPath , getObjects , load , notifyClose ,  
remove , save

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.persistence.RamPersistence:** objects ,  
PERSISTENCE\_NO\_NAME , resources

## 1.2. FilePersistence(IScope)

```
public FilePersistence(org.red5.server.api.IScope scope);
```

Create file persistence object for given scope

### Parameters

scope	Scope
-------	-------

## 1.3. FilePersistence(ResourcePatternResolver)

```
public FilePersistence(org.springframework.core.io.support.ResourcePatternResolver resolver);
```

Create file persistence object from given resource pattern resolver

### Parameters

resolver	Resource pattern resolver and loader
----------	--------------------------------------

## 1.4. checkRemoveEmptyDirectories(String)

```
protected void checkRemoveEmptyDirectories(String base);
```

Remove empty dirs

### Parameters

base	Base directory
------	----------------

## 1.5. getObjectPath(String, String)

```
protected String getObjectPath(String id,  
                             String name);
```

## 1.6. load(IPersistable)

```
public boolean load(org.red5.server.api.persistence.IPersistable object);
```

## 1.7. load(String)

```
public org.red5.server.api.persistence.IPersistable load(String name);
```

## 1.8. notifyClose()

```
public void notifyClose();
```

## 1.9. remove(IPersistable)

```
public boolean remove(org.red5.server.api.persistence.IPersistable object);
```

## 1.10. remove(String)

```
public boolean remove(String name);
```

## 1.11. save(IPersistable)

```
public boolean save(org.red5.server.api.persistence.IPersistable object);
```

## 1.12. saveObject(IPersistable)

```
protected boolean saveObject(org.red5.server.api.persistence.IPersistable object);
```

Save persistable object

### Parameters

object	Persistable object
<i>return</i>	true on success, false otherwise

## 1.13. setExtension(String)

```
public void setExtension(String extension);
```

Setter for extension.

### Parameters

extension	New extension.
-----------	----------------

## 1.14. setPath(String)

```
public void setPath(String path);
```

Setter for file path.

### Parameters

path	New path
------	----------

## 2. Class FilePersistenceThread

Thread that writes modified persistent objects to the file system periodically.

### 2.1. Synopsis

```
public class FilePersistenceThread implements java.lang.Runnable {
// Public Static Methods

    public static FilePersistenceThread getInstance();

// Public Methods

    public void run();

    public void shutdown();

// Protected Methods

    protected void modified(org.red5.server.api.persistence.IPersistable object,
```

```

        FilePersistence store);

protected void notifyClose(FilePersistence store);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. getInstance()

```
public static FilePersistenceThread getInstance();
```

Return singleton instance of the thread.

### Parameters

return
--------

## 2.3. modified(IPersistable, FilePersistence)

```

protected void modified(org.red5.server.api.persistence.IPersistable object,
                      FilePersistence store);

```

Notify thread that an object was modified in a persistence store.

### Parameters

object
--------

store
-------

## 2.4. notifyClose(FilePersistence)

```
protected void notifyClose(FilePersistence store);
```

Write any pending objects for the given store to disk.

### Parameters

store
-------

## 2.5. run()

```
public void run();
```

**Specified by:** Method run in interface Runnable

Write modified objects to the file system periodically.

## 2.6. shutdown()

```
public void shutdown();
```

Cleanly shutdown the tasks

## 3. Class RamPersistence

Persistence implementation that stores the objects in memory. This serves as default persistence if nothing has been configured.

### 3.1. Synopsis

```
public class RamPersistence implements org.?red5.?server.?api.?persistence.?IPersistenceStore {
    // Protected Fields

        protected static final String PERSISTENCE_NO_NAME = "__null__";

        protected java.util.Map<java.lang.String, org.red5.server.api.persistence.IPersistable> objects;

        protected org.springframework.core.io.support.ResourcePatternResolver resources;

    // Public Constructors

        public RamPersistence(org.red5.server.api.IScope scope);

        public RamPersistence(org.springframework.core.io.support.ResourcePatternResolver resources);

    // Public Methods

        public java.util.Set<java.lang.String> getObjectNames();

        public java.util.Collection<org.red5.server.api.persistence.IPersistable> getObjects();

        public org.red5.server.api.persistence.IPersistable load(String name);

        public boolean load(org.red5.server.api.persistence.IPersistable obj);

        public void notifyClose();

        public boolean remove(String name);

        public boolean remove(org.red5.server.api.persistence.IPersistable object);

        public boolean save(org.red5.server.api.persistence.IPersistable object);

    // Protected Methods

        protected String getObjectId(org.red5.server.api.persistence.IPersistable object);

        protected String getObjectName(String id);

        protected String getObjectPath(String id,
                                      String name);

}
```

**Direct known subclasses:** org.?red5.?server.?persistence.?FilePersistence

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

### 3.2. RamPersistence(IScope)

```
public RamPersistence(org.red5.server.api.IScope scope);
```

Creates RAM persistence object from scope

#### Parameters

scope	Scope
-------	-------

### 3.3. RamPersistence(ResourcePatternResolver)

```
public RamPersistence(org.springframework.core.io.support.ResourcePatternResolver resources);
```

Creates RAM persistence object from resource pattern resolvers

#### Parameters

resources	Resource pattern resolver and loader
-----------	--------------------------------------

### 3.4. objects

```
protected java.util.Map<java.lang.String, org.red5.server.api.persistence.IPersistable> objects ;
```

Map for persistable objects

### 3.5. PERSISTENCE\_NO\_NAME

```
protected static final String PERSISTENCE_NO_NAME = "__null__";
```

This is used in the id for objects that have a name of null

### 3.6. resources

```
protected org.springframework.core.io.support.ResourcePatternResolver resources ;
```

Resource pattern resolver. Resolves resources from patterns, loads resources.

### 3.7. getObjectId(IPersistable)

```
protected String getObjectId(org.red5.server.api.persistence.IPersistable object);
```

Get object id

#### Parameters

object	Persistable object whose id is asked for
return	Given persistable object id

### 3.8. getObjectName(String)

```
protected String getObjectName(String id);
```

Get resource name from path

Parameters	
id	Object ID. The format of the object id is //.
return	Resource name

### 3.9. getObjectNames()

```
public java.util.Set<java.lang.String> getObjectNames();
```

**Specified by:** Method getObjectNames in interface IPersistenceStore

Return iterator over the names of all already loaded objects in the storage.

### 3.10. getObjectPath(String, String)

```
protected String getObjectPath(String id,
                               String name);
```

Get object path for given id and name

Parameters	
id	Object ID. The format of the object id is //
name	Object name
return	Resource path

### 3.11. getObjects()

```
public java.util.Collection<org.red5.server.api.persistence.IPersistable> getObjects();
```

**Specified by:** Method getObjects in interface IPersistenceStore

Return iterator over the already loaded objects in the storage.

### 3.12. load(IPersistable)

```
public boolean load(org.red5.server.api.persistence.IPersistable obj);
```

**Specified by:** Method load in interface IPersistenceStore

Load state of an already instantiated persistent object.

### 3.13. load(String)

```
public org.red5.server.api.persistence.IPersistable load(String name);
```

**Specified by:** Method load in interface IPersistenceStore

Load a persistent object with the given name. The object must provide either a constructor that takes an input stream as only parameter or an empty constructor so it can be loaded from the persistence store.

### 3.14. notifyClose()

```
public void notifyClose();
```

**Specified by:** Method notifyClose in interface IPersistenceStore

Notify store that it's being closed. This allows the store to write any pending objects to disk.

### 3.15. remove(IPersistable)

```
public boolean remove(org.red5.server.api.persistence.IPersistable object);
```

**Specified by:** Method remove in interface IPersistenceStore

Delete the passed persistent object.

### 3.16. remove(String)

```
public boolean remove(String name);
```

**Specified by:** Method remove in interface IPersistenceStore

Delete the persistent object with the given name.

### 3.17. save(IPersistable)

```
public boolean save(org.red5.server.api.persistence.IPersistable object);
```

**Specified by:** Method save in interface IPersistenceStore

Persist given object.

# 1. Class ThreadPool

ThreadPool

## 1.1. Synopsis

```
public class ThreadPool implements, org.jboss.red5.server.pooling.ThreadPoolMBean {  
    // Public Constructors  
  
    public ThreadPool();  
  
    // Public Methods  
  
    public void execute(Runnable command);  
  
    public int getPoolSize();  
  
    public void setPoolSize(int poolSize);  
  
    public void shutdown();  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

# 2. Interface ThreadPoolMBean

JMX interface for Thread Pool instrumentation.

## 2.1. Synopsis

```
public interface ThreadPoolMBean {  
    // Public Methods  
  
    public int getPoolSize();  
  
    public void setPoolSize(int poolSize);  
}
```

# 3. Class Worker

```
public class Worker implements, java.lang.Runnable {  
    // Public Constructors  
  
    public Worker();  
  
    // Public Methods  
  
    public synchronized void execute(String className,
```

```

        String methName,
        Object[] params,
        Class[] paramTypes,
        Object synObj);

    public String getClassName();

    public String getMethodName();

    public Object[] getMethodParams();

    public Class[] getParamTypes();

    public Object getResult();

    public void reset();

    public void run();

    public void setClassName(String className);

    public void setMethodName(String methodName);

    public void setMethodParams(Object[] methodParams);

    public void setParamTypes(Class[] paramTypes);

    public void setResult(Object result);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

### 3.1. execute(String, String, Object[], Class<?>[], Object)

```

public synchronized void execute(String clsName,
                               String methName,
                               Object[] params,
                               Class[] paramTypes,
                               Object synObj);

```

execute

#### Parameters

clsName
methName
params
paramTypes
paramTypes

### 3.2. getClassName()

```
public String getClassName();
```

#### Parameters

<i>return</i>	Returns the className.
---------------	------------------------

### 3.3. getMethodName()

```
public String getMethodName();
```

#### Parameters

<i>return</i>	Returns the methodName.
---------------	-------------------------

### 3.4. getMethodParams()

```
public Object[] getMethodParams();
```

#### Parameters

<i>return</i>	Returns the methodParams.
---------------	---------------------------

### 3.5. getParamTypes()

```
public Class[] getParamTypes();
```

#### Parameters

<i>return</i>	Returns the paramTypes.
---------------	-------------------------

### 3.6. getResult()

```
public Object getResult();
```

#### Parameters

<i>return</i>	Returns the result.
---------------	---------------------

### 3.7. reset()

```
public void reset();
```

reset the members to service next request.

### 3.8. run()

```
public void run();
```

**Specified by:** Method `run` in interface `Runnable`

### 3.9. setClassName(String)

```
public void setClassName(String className);
```

#### Parameters

className	The className to set.
-----------	-----------------------

### 3.10. setMethodName(String)

```
public void setMethodName(String methodName);
```

#### Parameters

methodName	The methodName to set.
------------	------------------------

### 3.11. setMethodParams(Object[])

```
public void setMethodParams(Object[] methodParams);
```

#### Parameters

methodParams	The methodParams to set.
--------------	--------------------------

### 3.12. setParamTypes(Class<?>[])

```
public void setParamTypes(Class[] paramTypes);
```

#### Parameters

paramTypes	The paramTypes to set.
------------	------------------------

### 3.13. setResult(Object)

```
public void setResult(Object result);
```

#### Parameters

result	The result to set.
--------	--------------------

# 1. Class QuartzSchedulingService

Scheduling service that uses Quartz as backend.

## 1.1. Synopsis

```
public class QuartzSchedulingService implements org.red5.server.api.scheduling.ISchedulingService
// Public Constructors

    public QuartzSchedulingService();

// Public Methods

    public String addScheduledJob(int interval,
                                  org.red5.server.api.scheduling.IScheduledJob job);

    public String addScheduledJobAfterDelay(int interval,
                                            org.red5.server.api.scheduling.IScheduledJob job,
                                            int delay);

    public String addScheduledOnceJob(java.util.Date date,
                                      org.red5.server.api.scheduling.IScheduledJob job);

    public String addScheduledOnceJob(long timeDelta,
                                      org.red5.server.api.scheduling.IScheduledJob job);

    public String getJobName();

    public java.util.List<java.lang.String> getScheduledJobNames();

    public void removeScheduledJob(String name);

    public void shutdown()
        throws SchedulerException;

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. QuartzSchedulingService()

```
public QuartzSchedulingService();
```

Constructs a new QuartzSchedulingService.

## 1.3. addScheduledJob(int, IScheduledJob)

```
public String addScheduledJob(int interval,
                            org.red5.server.api.scheduling.IScheduledJob job);
```

**Specified by:** Method addScheduledJob in interface ISchedulingService

Schedule a job for periodic execution.

## 1.4. addScheduledJobAfterDelay(int, IScheduledJob, int)

```
public String addScheduledJobAfterDelay(int interval,
                                         org.red5.server.api.scheduling.IScheduledJob job,
                                         int delay);
```

**Specified by:** Method addScheduledJobAfterDelay in interface ISchedulingService

Schedule a job for periodic execution which will start after the specified delay.

## 1.5. addScheduledOnceJob(Date, IScheduledJob)

```
public String addScheduledOnceJob(java.util.Date date,
                                   org.red5.server.api.scheduling.IScheduledJob job);
```

**Specified by:** Method addScheduledOnceJob in interface ISchedulingService

Schedule a job for single execution at a given date. Please note that the jobs are not saved if Red5 is restarted in the meantime.

## 1.6. addScheduledOnceJob(long, IScheduledJob)

```
public String addScheduledOnceJob(long timeDelta,
                                   org.red5.server.api.scheduling.IScheduledJob job);
```

**Specified by:** Method addScheduledOnceJob in interface ISchedulingService

Schedule a job for single execution in the future. Please note that the jobs are not saved if Red5 is restarted in the meantime.

## 1.7. getJobName()

```
public String getJobName();
```

**Specified by:** Method getJobName in interface QuartzSchedulingServiceMBean

Getter for job name.

### Parameters

<i>return</i>	Job name
---------------	----------

## 1.8. getScheduledJobNames()

```
public java.util.List<java.lang.String> getScheduledJobNames();
```

**Specified by:** Method getScheduledJobNames in interface ISchedulingService

Return names of scheduled jobs.

## 1.9. removeScheduledJob(String)

```
public void removeScheduledJob(String name);
```

**Specified by:** Method removeScheduledJob in interface ISchedulingService

Stop executing a previously scheduled job.

## 2. Class QuartzSchedulingServiceJob

Scheduled job that is registered in the Quartz scheduler.

### 2.1. Synopsis

```
public class QuartzSchedulingServiceJob implements org.quartz.Job {
    // Protected Fields

    protected static final String SCHEDULED_JOB = "scheduled_job";

    protected static final String SCHEDULING_SERVICE = "scheduling_service";

    // Public Constructors

    public QuartzSchedulingServiceJob();

    // Public Methods

    public void execute(org.quartz.JobExecutionContext arg0)
        throws JobExecutionException;

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 2.2. SCHEDULED\_JOB

```
protected static final String SCHEDULED_JOB = "scheduled_job";
```

Scheduled job constant

### 2.3. SCHEDULING\_SERVICE

```
protected static final String SCHEDULING_SERVICE = "scheduling_service";
```

Scheduling service constant

### 2.4. execute(JobExecutionContext)

```
public void execute(org.quartz.JobExecutionContext arg0)
    throws JobExecutionException;
```

**Specified by:** Method execute in interface Job

## 3. Interface QuartzSchedulingServiceMBean

Scheduling service that uses Quartz as backend.

### 3.1. Synopsis

```
public interface QuartzSchedulingServiceMBean {  
    // Public Methods  
  
    public String getJobName();  
  
    public java.util.List<java.lang.String> getScheduledJobNames();  
  
    public void removeScheduledJob(String name);  
}
```

### 3.2. getJobName()

```
public String getJobName();
```

Getter for job name.

#### Parameters

<i>return</i>	Job name
---------------	----------

# 1. Class GroovyScriptFactory

org.springframework.scripting.ScriptFactory implementation for a Groovy script.

Typically used in combination with a

org.springframework.scripting.support.ScriptFactoryPostProcessor; see the latter's Javadoc for a configuration example.

## 1.1. Synopsis

```
public class GroovyScriptFactory implements org.springframework.scripting.ScriptFactory, org.springfr...
```

// Public Constructors

```
public GroovyScriptFactory(String scriptSourceLocator);
```

```
public GroovyScriptFactory(String scriptSourceLocator, Class[] scriptInterfaces);
```

```
public GroovyScriptFactory(String scriptSourceLocator, org.springframework.scripting.groovy.GroovyObjectCustomizer groovyObjec...
```

// Public Methods

```
public Class[] getScriptInterfaces();
```

```
public String getScriptSourceLocator();
```

```
public Object getScriptedObject(org.springframework.scripting.ScriptSource scriptSource, Class[] actualInterfaces) throws IOException, ScriptCompilationException;
```

```
public Class getScriptedObjectType(org.springframework.scripting.ScriptSource scriptSource) throws IOException, ScriptCompilationException;
```

```
public boolean requiresConfigInterface();
```

```
public boolean requiresScriptedObjectRefresh(org.springframework.scripting.ScriptSource src);
```

```
public void setBeanClassLoader(ClassLoader classLoader);
```

```
public String toString();
```

// Protected Methods

```
protected Object executeScript(Class scriptClass) throws ScriptCompilationException;
```

```
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

*Since*

2.0

**See Also**

[org.springframework.scripting.support.ScriptFactoryPostProcessor](#),  
[groovy.lang.GroovyClassLoader](#)

## 1.2. GroovyScriptFactory(String)

```
public GroovyScriptFactory(String scriptSourceLocator);
```

Create a new GroovyScriptFactory for the given script source.

We don't need to specify script interfaces here, since a Groovy script defines its Java interfaces itself.

**Parameters**

scriptSourceLocator	a locator that points to the source of the script. Interpreted by the post-processor that actually creates the script.
---------------------	--

## 1.3. GroovyScriptFactory(String, GroovyObjectCustomizer)

```
public GroovyScriptFactory(String scriptSourceLocator,  
                           org.springframework.scripting.groovy.GroovyObjectCustomizer groovyObjectC
```

Create a new GroovyScriptFactory for the given script source, specifying a strategy interface that can create a custom MetaClass to supply missing methods and otherwise change the behavior of the object.

We don't need to specify script interfaces here, since a Groovy script defines its Java interfaces itself.

**Parameters**

scriptSourceLocator	a locator that points to the source of the script. Interpreted by the post-processor that actually creates the script.
groovyObjectCustomizer	customizer that can set a custom metaclass or make other changes to the GroovyObject created by this factory (may be null)

## 1.4. executeScript(Class)

```
protected Object executeScript(Class scriptClass)  
throws ScriptCompilationException;
```

Instantiate the given Groovy script class and run it if necessary.

**Parameters**

scriptClass	the Groovy script class
return	the result object (either an instance of the script class or the result of running the script instance)

ScriptCompilationException  
in case of instantiation failure

## 1.5. getScriptedObject( ScriptSource, Class[] )

```
public Object getScriptedObject(org.springframework.scripting.ScriptSource scriptSource,
                               Class[] actualInterfaces)
                           throws IOException, ScriptCompilationException;
```

**Specified by:** Method `getScriptedObject` in interface `ScriptFactory`

Loads and parses the Groovy script via the GroovyClassLoader.

**See Also**

`groovy.lang.GroovyClassLoader`

## 1.6. getScriptInterfaces()

```
public Class[] getScriptInterfaces();
```

**Specified by:** Method `getScriptInterfaces` in interface `ScriptFactory`

Groovy scripts determine their interfaces themselves, hence we don't need to explicitly expose interfaces here.

Parameters	
<i>return</i>	null always

## 1.7. requiresConfigInterface()

```
public boolean requiresConfigInterface();
```

**Specified by:** Method `requiresConfigInterface` in interface `ScriptFactory`

Groovy scripts do not need a config interface, since they expose their setters as public methods.

# 1. Class JythonScriptFactory

org.springframework.scripting.ScriptFactory implementation for a Python script.

## 1.1. Synopsis

```
public class JythonScriptFactory implements org.springframework.scripting.ScriptFactory {  
    // Public Constructors  
  
    public JythonScriptFactory(String scriptSourceLocator);  
  
    public JythonScriptFactory(String scriptSourceLocator,  
        Class[] scriptInterfaces);  
  
    public JythonScriptFactory(String scriptSourceLocator,  
        Class[] scriptInterfaces,  
        Object[] arguments);  
  
    // Public Methods  
  
    public Class[] getScriptInterfaces();  
  
    public String getScriptSourceLocator();  
  
    public Object getScriptedObject(org.springframework.scripting.ScriptSource scriptSourceLocator,  
        Class[] scriptInterfaces)  
        throws IOException, ScriptCompilationException;  
  
    public Class getScriptedObjectType(org.springframework.scripting.ScriptSource src)  
        throws IOException, ScriptCompilationException;  
  
    public boolean requiresConfigInterface();  
  
    public boolean requiresScriptedObjectRefresh(org.springframework.scripting.ScriptSource src);  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### See Also

A prototype of Jython ScriptFactory for Spring Framework [<http://yanweng.blogspot.com/2006/02/prototype-of-jython-scriptfactory-for.html>]

## 1.2. getScriptedObject(ScriptSource, Class[])

```
public Object getScriptedObject(org.springframework.scripting.ScriptSource scriptSourceLocator,  
    Class[] scriptInterfaces)  
    throws IOException, ScriptCompilationException;
```

**Specified by:** Method `getScriptedObject` in interface `ScriptFactory`

## 1.3. getScriptInterfaces()

```
public Class[] getScriptInterfaces();
```

**Specified by:** Method `getScriptInterfaces` in interface `ScriptFactory`

## 1.4. `getScriptSourceLocator()`

```
public String getScriptSourceLocator();
```

**Specified by:** Method `getScriptSourceLocator` in interface `ScriptFactory`

## 1.5. `requiresConfigInterface()`

```
public boolean requiresConfigInterface();
```

**Specified by:** Method `requiresConfigInterface` in interface `ScriptFactory`

# 1. Class RhinoScriptFactory

org.springframework.scripting.ScriptFactory implementation for a Rhino / Javascript script.

Typically used in combination with a

org.springframework.scripting.support.ScriptFactoryPostProcessor; see the latter's Javadoc for a configuration example.

## 1.1. Synopsis

```
public class RhinoScriptFactory implements org.springframework.scripting.ScriptFactory {  
    // Public Constructors  
  
    public RhinoScriptFactory(String scriptSourceLocator);  
  
    public RhinoScriptFactory(String scriptSourceLocator,  
                             Class scriptInterface);  
  
    public RhinoScriptFactory(String scriptSourceLocator,  
                             Class[] scriptInterfaces);  
  
    public RhinoScriptFactory(String scriptSourceLocator,  
                             Class[] scriptInterfaces,  
                             Class extendedClass);  
  
    // Public Methods  
  
    public Class[] getScriptInterfaces();  
  
    public String getScriptSourceLocator();  
  
    public Object getScriptedObject(org.springframework.scripting.ScriptSource actualScriptSource,  
                                   Class[] actualInterfaces)  
        throws IOException, ScriptCompilationException;  
  
    public Class getScriptedObjectType(org.springframework.scripting.ScriptSource src)  
        throws IOException, ScriptCompilationException;  
  
    public boolean requiresConfigInterface();  
  
    public boolean requiresScriptedObjectRefresh(org.springframework.scripting.ScriptSource src);  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

*Since*

0.6

*See Also*

org.springframework.scripting.support.ScriptFactoryPostProcessor,  
org.red5.server.script.rhino.RhinoScriptUtils

## 1.2. RhinoScriptFactory(String, Class[])

```
public RhinoScriptFactory(String scriptSourceLocator,
                         Class[] scriptInterfaces);
```

Create a new RhinoScriptFactory for the given script source.

Parameters	
scriptSourceLocator	a locator that points to the source of the script. Interpreted by the post-processor that actually creates the script.
scriptInterfaces	the Java interfaces that the scripted object is supposed to implement

`IllegalArgumentException`  
 if either of the supplied arguments is `null`; or the supplied `scriptSourceLocator` argument is composed wholly of whitespace; or if the supplied `scriptInterfaces` argument array has no elements

## 1.3. getScriptedObject(DataSource, Class[])

```
public Object getScriptedObject(org.springframework.scripting.ScriptSource actualScriptSource,
                               Class[] actualInterfaces)
throws IOException, ScriptCompilationException;
```

**Specified by:** Method `getScriptedObject` in interface `ScriptFactory`

Load and parse the Rhino script via RhinoScriptUtils.

**See Also**

`org.red5.server.script.rhino.RhinoScriptUtils`

## 1.4. getScriptInterfaces()

```
public Class[] getScriptInterfaces();
```

**Specified by:** Method `getScriptInterfaces` in interface `ScriptFactory`

## 1.5. getScriptSourceLocator()

```
public String getScriptSourceLocator();
```

**Specified by:** Method `getScriptSourceLocator` in interface `ScriptFactory`

## 1.6. requiresConfigInterface()

```
public boolean requiresConfigInterface();
```

**Specified by:** Method `requiresConfigInterface` in interface `ScriptFactory`

Rhino scripts do not require a config interface.

Parameters
------------

<i>return</i>	false always
---------------	--------------

## 2. Class RhinoScriptUtils

Utility methods for handling Rhino / Javascript objects.

### 2.1. Synopsis

```
public class RhinoScriptUtils {
// Public Constructors

    public RhinoScriptUtils();

// Public Static Methods

    public static Object createRhinoObject(String scriptSource,
                                           Class[] interfaces,
                                           Class extendedClass)
        throws ScriptCompilationException, IOException, Exception;

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

*Since*

0.6

### 2.2. createRhinoObject(String, Class[], Class)

```
public static Object createRhinoObject(String scriptSource,
                                       Class[] interfaces,
                                       Class extendedClass)
    throws ScriptCompilationException, IOException, Exception;
```

Create a new Rhino-scripted object from the given script source.

Parameters	
scriptSource	the script source text
interfaces	the interfaces that the scripted Java object is supposed to implement
extendedClass	
<i>return</i>	the scripted Java object

ScriptCompilationException  
in case of Rhino parsing failure

java.io.IOException  
java.io.IOException

# 1. Class Call

Basic service call (remote call) implementation

## 1.1. Synopsis

```
public class Call implements, org.red5.server.api.service.IServiceCall, java.io.Externalizable
// Public Static Fields

    public static final byte STATUS_ACCESS_DENIED = 18;

    public static final byte STATUS_APP_SHUTTING_DOWN = 21;

    public static final byte STATUS_GENERAL_EXCEPTION = 20;

    public static final byte STATUS_INVOCATION_EXCEPTION = 19;

    public static final byte STATUS_METHOD_NOT_FOUND = 17;

    public static final byte STATUS_PENDING = 1;

    public static final byte STATUS_SERVICE_NOT_FOUND = 16;

    public static final byte STATUS_SUCCESS_NULL = 3;

    public static final byte STATUS_SUCCESS_RESULT = 2;

    public static final byte STATUS_SUCCESS_VOID = 4;

// Protected Fields

    protected Object[] arguments ;

    protected Exception exception ;

    protected String serviceMethodName ;

    protected String serviceName ;

    protected byte status ;

// Public Constructors

    public Call();

    public Call(String method);

    public Call(String method,
               Object[] args);

    public Call(String name,
               String method,
               Object[] args);

// Public Methods
```

```

public Object[] getArguments();

public Exception getException();

public String getServiceMethodName();

public String getServiceName();

public byte getStatus();

public boolean isSuccess();

public void readExternal(java.io.ObjectInput in)
    throws IOException, ClassNotFoundException;

public void setArguments(Object[] args);

public void setException(Exception exception);

public void setServiceMethodName(String serviceMethodName);

public void setServiceName(String serviceName);

public void setStatus(byte status);

public String toString();

public void writeExternal(java.io.ObjectOutput out)
    throws IOException;

}

```

**Direct known subclasses:** org.?red5.?server.?service.?PendingCall

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. Call(String)

```
public Call(String method);
```

Creates call from method name

### Parameters

method	Method name
--------	-------------

## 1.3. Call(String, Object[])

```
public Call(String method,
    Object[] args);
```

Creates call from method name and array of call parameters

**Parameters**

method	Method name
args	Call parameters

**1.4. Call(String, String, Object[])**

```
public Call(String name,
           String method,
           Object[] args);
```

Creates call from given service name, method name and array of call parameters

**Parameters**

name	Service name
method	Service method name
args	Call parameters

**1.5. arguments**

```
protected Object[] arguments ;
```

Call arguments

**1.6. exception**

```
protected Exception exception ;
```

Call exception if any, null by default

**1.7. serviceMethodName**

```
protected String serviceMethodName ;
```

Service method name

**1.8. serviceName**

```
protected String serviceName ;
```

Service name

**1.9. status**

```
protected byte status ;
```

Call status, initial one is pending

**1.10. STATUS\_ACCESS\_DENIED**

```
public static final byte STATUS_ACCESS_DENIED = 18;
```

Access denied constant

## 1.11. STATUS\_APP\_SHUTTING\_DOWN

```
public static final byte STATUS_APP_SHUTTING_DOWN = 21;
```

The application for this service is currently shutting down

## 1.12. STATUS\_GENERAL\_EXCEPTION

```
public static final byte STATUS_GENERAL_EXCEPTION = 20;
```

General exception constant

## 1.13. STATUS\_INVOCATION\_EXCEPTION

```
public static final byte STATUS_INVOCATION_EXCEPTION = 19;
```

Exception on invocation constant

## 1.14. STATUS\_METHOD\_NOT\_FOUND

```
public static final byte STATUS_METHOD_NOT_FOUND = 17;
```

Service's method not found constant

## 1.15. STATUS\_PENDING

```
public static final byte STATUS_PENDING = 1;
```

Pending status constant

## 1.16. STATUS\_SERVICE\_NOT\_FOUND

```
public static final byte STATUS_SERVICE_NOT_FOUND = 16;
```

Service not found constant

## 1.17. STATUS\_SUCCESS\_NULL

```
public static final byte STATUS_SUCCESS_NULL = 3;
```

Returned value is null constant

## 1.18. STATUS\_SUCCESS\_RESULT

```
public static final byte STATUS_SUCCESS_RESULT = 2;
```

Success result constant

## 1.19. STATUS\_SUCCESS\_VOID

```
public static final byte STATUS_SUCCESS_VOID = 4;
```

Service returns no value constant

## 1.20. getArguments()

```
public Object[] getArguments();
```

**Specified by:** Method getArguments in interface IServiceCall

Returns array of service method arguments

## 1.21. getException()

```
public Exception getException();
```

**Specified by:** Method getException in interface IServiceCall

Get service call exception

## 1.22. getServiceMethodName()

```
public String getServiceMethodName();
```

**Specified by:** Method getServiceMethodName in interface IServiceCall

Returns service method name

## 1.23. getServiceName()

```
public String getServiceName();
```

**Specified by:** Method getServiceName in interface IServiceCall

Returns service name

## 1.24. getStatus()

```
public byte getStatus();
```

**Specified by:** Method getStatus in interface IServiceCall

Get service call status

## 1.25. isSuccess()

```
public boolean isSuccess();
```

**Specified by:** Method isSuccess in interface IServiceCall

Whether call was successful or not

## 1.26. setArguments(Object[])

```
public void setArguments(Object[] args);
```

Setter for arguments.

**Parameters**

args	Arguments.
------	------------

**1.27. setException(Exception)**

```
public void setException(Exception exception);
```

**Specified by:** Method setException in interface IServiceCall

Sets exception

**1.28. setServiceMethodName(String)**

```
public void setServiceMethodName(String serviceMethodName);
```

Setter for service method name

**Parameters**

serviceMethodName	New service method name value
-------------------	-------------------------------

**1.29. setServiceName(String)**

```
public void setServiceName(String serviceName);
```

Setter for service name

**Parameters**

serviceName	New service name value
-------------	------------------------

**1.30. setStatus(byte)**

```
public void setStatus(byte status);
```

**Specified by:** Method setStatus in interface IServiceCall

Sets status

**1.31. toString()**

```
public String toString();
```

**2. Class ContextServiceResolver**

Resolve services that have been configured in the context of a scope.

**2.1. Synopsis**

```
public class ContextServiceResolver implements org.?red5.?server.?service.?IServiceResolver {
// Public Constructors

    public ContextServiceResolver();
```

```
// Public Methods

    public Object resolveService(org.red5.server.api.IScope scope,
                                String serviceName);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. resolveService(IScope, String)

```
public Object resolveService(org.red5.server.api.IScope scope,
                            String serviceName);
```

**Specified by:** Method resolveService in interface IServiceResolver

Search for a service with the given name in the scope.

# 3. Class ConversionUtils

Misc utils for conversions

## 3.1. Synopsis

```
public class ConversionUtils {
    // Protected Fields

        protected static org.slf4j.Logger log;

    // Public Constructors

        public ConversionUtils();

    // Public Static Methods

        public static Object convert(Object source,
                                    Class<?> target)
        throws ConversionException;

        public static java.util.List<?> convertArrayToList(Object[] source)
        throws ConversionException;

        public static java.util.Set<?> convertArrayToSet(Object[] source);

        public static java.util.Map<?, ?> convertBeanToMap(Object source);

        public static Object convertMapToBean(java.util.Map<?, ?> source,
                                            Class<?> target)
        throws ConversionException;

        public static java.util.List<java.lang.Object> convertMapToList(java.util.Map<?, ?> map);

        public static Object convertNumberToWrapper(Number num,
                                                Class<?> wrapper);
```

```

public static Object[] convertParams(Object[] source,
                                     Class[] target)
throws ConversionException;

public static Object convertStringToWrapper(String str,
                                           Class<?> wrapper);

public static Object convertToArray(Object source,
                                    Class<?> target)
throws ConversionException;

public static Object convertToWrappedPrimitive(Object source,
                                              Class<?> wrapper);

public static java.util.List<java.lang.reflect.Method> findMethodsByNameAndNumParams(Object object,
                                                                                         String method,
                                                                                         int numParam)

// Protected Methods

protected static Object newInstance(String className);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 3.2. convert(Object, Class<?>)

```

public static Object convert(Object source,
                            Class<?> target)
throws ConversionException;

```

Convert source to given class

#### Parameters

source	Source object
target	Target class
<i>return</i>	Converted object

ConversionException

If object can't be converted

### 3.3. convertArrayToList(Object[])

```

public static java.util.List<?> convertArrayToList(Object[] source)
throws ConversionException;

```

#### Parameters

source	
<i>return</i>	

ConversionException  
 org.apache.commons.beanutils.ConversionException

### 3.4. convertArrayToSet(Object[])

```
public static java.util.Set<?> convertArrayToSet(Object[] source);
```

Convert array to set, removing duplicates

#### Parameters

source	Source array
<i>return</i>	Set

### 3.5. convertBeanToMap(Object)

```
public static java.util.Map<?, ?> convertBeanToMap(Object source);
```

Convert bean to map

#### Parameters

source	Source bean
<i>return</i>	Converted map

### 3.6. convertMapToBean(Map<?, ?, Class<?>)

```
public static Object convertMapToBean(java.util.Map<?, ?> source,
                                      Class<?> target)
                                      throws ConversionException;
```

Convert map to bean

#### Parameters

source	Source map
target	Target class
<i>return</i>	Bean of that class

ConversionException  
 org.apache.commons.beanutils.ConversionException

### 3.7. convertNumberToWrapper(Number, Class<?>)

```
public static Object convertNumberToWrapper(Number num,
                                            Class<?> wrapper);
```

Convert number to primitive wrapper like Boolean or Float

#### Parameters

num	Number to convert
-----	-------------------

wrapper	Primitive wrapper type
<i>return</i>	Converted object

### 3.8. convertParams(Object[], Class<?>[])

```
public static Object[] convertParams(Object[] source,
                                     Class[] target)
                                     throws ConversionException;
```

Convert parameters using methods of this utility class

Parameters	
source	Array of source object
target	Array of target classes
<i>return</i>	Array of converted objects

ConversionException

If object can't be converted

### 3.9. convertStringToWrapper(String, Class<?>)

```
public static Object convertStringToWrapper(String str,
                                           Class<?> wrapper);
```

Convert string to primitive wrapper like Boolean or Float

Parameters	
str	String to convert
wrapper	Primitive wrapper type
<i>return</i>	Converted object

### 3.10. convertToArray(Object, Class<?>)

```
public static Object convertToArray(Object source,
                                    Class<?> target)
                                    throws ConversionException;
```

Convert to array

Parameters	
source	Source object
target	Target class
<i>return</i>	Converted object

ConversionException

If object can't be converted

### 3.11. convertToWrappedPrimitive(Object, Class<?>)

```
public static Object convertToWrappedPrimitive(Object source,
                                              Class<?> wrapper);
```

Convert to wrapped primitive

#### Parameters

source	Source object
wrapper	Primitive wrapper type
<i>return</i>	Converted object

### 3.12. findMethodsByNameAndNumParams(Object, String, int)

```
public static java.util.List<java.lang.reflect.Method> findMethodsByNameAndNumParams(Object object,
                                                                                      String method,
                                                                                      int numParam);
```

Find method by name and number of parameters

#### Parameters

object	Object to find method on
method	Method name
numParam	Number of parameters
<i>return</i>	List of methods that match by name and number of parameters

### 3.13. newInstance(String)

```
protected static Object newInstance(String className);
```

Create new class instance

#### Parameters

className	Class name; may not be loaded by JVM yet
<i>return</i>	Instance of given class

## 4. Class HandlerServiceResolver

Allow scope handlers to create service handlers dynamically.

### 4.1. Synopsis

```
public class HandlerServiceResolver implements org.?red5.?server.?service.?IServiceResolver {
// Public Constructors

    public HandlerServiceResolver();

// Public Methods
```

```
public Object resolveService(org.red5.server.api.IScope scope,
                           String serviceName);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 4.2. resolveService(IScope, String)

```
public Object resolveService(org.red5.server.api.IScope scope,
                           String serviceName);
```

**Specified by:** Method resolveService in interface IServiceResolver

Search for a service with the given name in the scope.

## 5. Interface IServiceResolver

Interface for objects that resolve service names to services. This is used by the ServiceInvoker to lookup the service to invoke a method on.

### 5.1. Synopsis

```
public interface IServiceResolver {
    // Public Methods

    public Object resolveService(org.red5.server.api.IScope scope,
                               String serviceName);
}
```

**See Also**

[org.red5.server.service.ServiceInvoker](#)

## 5.2. resolveService(IScope, String)

```
public Object resolveService(org.red5.server.api.IScope scope,
                           String serviceName);
```

Search for a service with the given name in the scope.

#### Parameters

scope	the scope to search in
serviceName	the name of the service
return	the object implementing the service or <code>null</code> if service doesn't exist

## 6. Exception MethodNotFoundException

Thrown if service method is not found so call throws exception

## 6.1. Synopsis

```
public class MethodNotFoundException extends java.lang.RuntimeException {
// Public Constructors

    public MethodNotFoundException(String methodName);

    public MethodNotFoundException(String methodName,
                                  Object[] args);

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 6.2. MethodNotFoundException(String)

```
public MethodNotFoundException(String methodName);
```

Creates exception with given method name

Parameters	
methodName	Service method name that can't be found

## 6.3. MethodNotFoundException(String, Object[])

```
public MethodNotFoundException(String methodName,
                             Object[] args);
```

Creates exception with given method name and arguments

Parameters	
methodName	Service method name that can't be found
args	Arguments given

## 7. Exception NotAllowedException

Thrown when a client is not allowed to execute a method.

## 7.1. Synopsis

```
public class NotAllowedException extends java.lang.RuntimeException {
// Public Constructors
```

```

    public NotAllowedException();

    public NotAllowedException(String message);

}

```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 8. Class PendingCall

Pending call is remote call operation that is in pending state. Remote calls to services are asynchronous, that is, after call but before result callback remote calls are in pending state.

### 8.1. Synopsis

```

public class PendingCall extends, org.?red5.?server.?service.?Call
    implements, org.?red5.?server.?api.?service.?IPendingServiceCall {
// Public Constructors

    public PendingCall();

    public PendingCall(String method);

    public PendingCall(String method,
                      Object[] args);

    public PendingCall(String name,
                      String method,
                      Object[] args);

// Public Methods

    public java.util.Set<org.red5.server.api.service.IPendingServiceCallback> getCallbacks();

    public Object getResult();

    public void readExternal(java.io.ObjectInput in)
        throws IOException, ClassNotFoundException;

    public void registerCallback(org.red5.server.api.service.IPendingServiceCallback callback);

    public void setResult(Object result);

    public void unregisterCallback(org.red5.server.api.service.IPendingServiceCallback callback);

    public void writeExternal(java.io.ObjectOutput out)
        throws IOException;

}

```

**Direct known subclasses:** org.?red5.?server.?net.?remoting.?message.?RemotingCall

**Methods inherited from org.red5.server.service.Call:** getArguments , getException , getServiceMethodName , getServiceName , getStatus , isSuccess , readExternal , setArguments , setException , setServiceMethodName , setServiceName , setStatus , toString , writeExternal

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

**Fields inherited from org.red5.server.service.Call:** arguments , exception , serviceMethodName , serviceName , status , STATUS\_ACCESS\_DENIED , STATUS\_APP\_SHUTTING\_DOWN , STATUS\_GENERAL\_EXCEPTION , STATUS\_INVOCATION\_EXCEPTION , STATUS\_METHOD\_NOT\_FOUND , STATUS\_PENDING , STATUS\_SERVICE\_NOT\_FOUND , STATUS\_SUCCESS\_NULL , STATUS\_SUCCESS\_RESULT , STATUS\_SUCCESS\_VOID

## 8.2. PendingCall(String)

```
public PendingCall(String method);
```

Creates pending call with given method name

Parameters	
method	Method name

## 8.3. PendingCall(String, Object[])

```
public PendingCall(String method,
                   Object[] args);
```

Creates pending call with given method name and array of parameters

Parameters	
method	Method name
args	Parameters

## 8.4. PendingCall(String, String, Object[])

```
public PendingCall(String name,
                   String method,
                   Object[] args);
```

Creates pending call with given method name, service name and array of parameters

Parameters	
name	Service name
method	Method name

args	Parameters
------	------------

## 8.5. getCallbacks()

```
public java.util.Set<org.red5.server.api.service.IPendingServiceCallback> getCallbacks();
```

**Specified by:** Method getCallbacks in interface IPendingServiceCall

Returns list of callback objects, usually callback object represented as an anonymous class instance that implements IPendingServiceCallback interface.

## 8.6. getResult()

```
public Object getResult();
```

**Specified by:** Method getResult in interface IPendingServiceCall

Returns service call result

## 8.7. registerCallback(IPendingServiceCallback)

```
public void registerCallback(org.red5.server.api.service.IPendingServiceCallback callback);
```

**Specified by:** Method registerCallback in interface IPendingServiceCall

Registers callback object usually represented as an anonymous class instance that implements IPendingServiceCallback interface.

## 8.8. setResult(Object)

```
public void setResult(Object result);
```

**Specified by:** Method setResult in interface IPendingServiceCall

Setter for property 'result'.

## 8.9. unregisterCallback(IPendingServiceCallback)

```
public void unregisterCallback(org.red5.server.api.service.IPendingServiceCallback callback);
```

**Specified by:** Method unregisterCallback in interface IPendingServiceCall

Unregisters callback object usually represented as an anonymous class instance that implements IPendingServiceCallback interface.

# 9. Class ScopeServiceResolver

Resolves service names in custom configured services of a scope.

## 9.1. Synopsis

```
public class ScopeServiceResolver implements, org.?red5.?server.?service.?IServiceResolver {
    // Public Constructors
```

```

public ScopeServiceResolver();

// Public Methods

public Object resolveService(org.red5.server.api.IScope scope,
                           String serviceName);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

#### See Also

registerServiceHandler(java.lang.String, java.lang.Object)

## 9.2. resolveService(IScope, String)

```

public Object resolveService(org.red5.server.api.IScope scope,
                           String serviceName);

```

**Specified by:** Method resolveService in interface IServiceResolver

Search for a service with the given name in the scope.

# 10. Class ServiceInvoker

Makes remote calls, invoking services, resolves service handlers

## 10.1. Synopsis

```

public class ServiceInvoker implements org.?red5.?server.?api.?service.?IServiceInvoker {
// Public Static Fields

    public static final String SERVICE_NAME = "serviceInvoker";

// Public Constructors

    public ServiceInvoker();

// Public Methods

    public boolean invoke(org.red5.server.api.service.IServiceCall call,
                         Object service);

    public boolean invoke(org.red5.server.api.service.IServiceCall call,
                         org.red5.server.api.IScope scope);

    public void setServiceResolvers(java.util.Set<org.red5.server.service.IServiceResolver> resolvers)
}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 10.2. SERVICE\_NAME

```
public static final String SERVICE_NAME = "serviceInvoker";
```

Service name

## 10.3. invoke(IServiceCall, IScope)

```
public boolean invoke(org.red5.server.api.service.IServiceCall call,
                     org.red5.server.api.IScope scope);
```

**Specified by:** Method invoke in interface IServicelInvoker

Execute the passed service call in the given scope. This looks up the handler for the call in the scope and the context of the scope.

## 10.4. invoke(IServiceCall, Object)

```
public boolean invoke(org.red5.server.api.service.IServiceCall call,
                     Object service);
```

**Specified by:** Method invoke in interface IServicelInvoker

Execute the passed service call in the given object.

## 10.5. setServiceResolvers(Set<IServiceResolver>)

```
public void setServiceResolvers(java.util.Set<org.red5.server.service.IServiceResolver> resolvers);
```

Setter for service resolvers.

Parameters	
resolvers	Service resolvers

# 11. Exception ServiceNotFoundException

Thrown when service can't be found thus remote call throws an exception

## 11.1. Synopsis

```
public class ServiceNotFoundException extends java.lang.RuntimeException {
    // Public Constructors

    public ServiceNotFoundException(String serviceName);

    // Public Methods

    public String getServiceName();

}
```

**Methods inherited from java.lang.Throwable:** fillInStackTrace , getCause , getLocalizedMessage , getMessage , getStackTrace , initCause , printStackTrace , setStackTrace , toString

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , wait

## 11.2. ServiceNotFoundException(String)

```
public ServiceNotFoundException(String serviceName);
```

Creates new exception with service name

### Parameters

serviceName	Name of service that couldn't been found
-------------	--

## 11.3. getServiceName()

```
public String getServiceName();
```

Get the name of the service that doesn't exist.

### Parameters

return	name of the service
--------	---------------------

## 12. Class ServiceUtils

```
public class ServiceUtils {
// Public Constructors

    public ServiceUtils();

// Public Static Methods

    public static Object[] findMethodWithExactParameters(Object service,
                                                       String methodName,
                                                       Object[] args);

    public static Object[] findMethodWithExactParameters(Object service,
                                                       String methodName,
                                                       java.util.List args);

    public static Object[] findMethodWithListParameters(Object service,
                                                       String methodName,
                                                       Object[] args);

    public static Object[] findMethodWithListParameters(Object service,
                                                       String methodName,
                                                       java.util.List args);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 12.1. findMethodWithExactParameters(Object, String, List)

```
public static Object[] findMethodWithExactParameters(Object service,
                                                    String methodName,
                                                    java.util.List args);
```

Returns (method, params) for the given service or (null, null) if no method was found.

### Parameters

service	Service
methodName	Method name
args	Arguments
<i>return</i>	Method/params pairs

## 12.2. findMethodWithExactParameters(Object, String, Object[])

```
public static Object[] findMethodWithExactParameters(Object service,
                                                    String methodName,
                                                    Object[] args);
```

Returns (method, params) for the given service or (null, null) if no method was found. XXX use ranking for method matching rather than exact type matching plus type conversion.

### Parameters

service	Service
methodName	Method name
args	Arguments
<i>return</i>	Method/params pairs

## 12.3. findMethodWithListParameters(Object, String, List)

```
public static Object[] findMethodWithListParameters(Object service,
                                                    String methodName,
                                                    java.util.List args);
```

Returns (method, params) for the given service or (null, null) if no method was found.

### Parameters

service	Service
methodName	Method name
args	Arguments
<i>return</i>	Method/params pairs

## 12.4. findMethodWithListParameters(Object, String, Object[])

```
public static Object[] findMethodWithListParameters(Object service,
```

```
String methodName,  
Object[] args);
```

Returns (method, params) for the given service or (null, null) if not method was found.

#### Parameters

service	Service
methodName	Method name
args	Arguments
<i>return</i>	Method/params pairs

# 1. Class ClientSharedObject

Works with client-side shared object

## 1.1. Synopsis

```
public class ClientSharedObject extends org.red5.server.so.SharedObject
    implements org.red5.server.api.so.IClientSharedObject, org.red5.server.api.event.IEvent
// Protected Fields

    protected static org.slf4j.Logger log;

// Public Constructors

    public ClientSharedObject(String name,
                           boolean persistent);

// Public Methods

    public void addSharedObjectListener(org.red5.server.api.so.ISharedObjectListener listener);

    public void beginUpdate();

    public void beginUpdate(org.red5.server.api.event.IEventListener listener);

    public boolean clear();

    public void close();

    public void connect(org.red5.server.api.IConnection conn);

    public void disconnect();

    public void dispatchEvent(org.red5.server.api.event.IEvent e);

    public void endUpdate();

    public Object getAttribute(String name,
                           Object defaultValue);

    public Object getServiceHandler(String name);

    public java.util.Set<java.lang.String> getServiceHandlerNames();

    public boolean isConnected();

    public boolean isLocked();

    public void lock();

    public void registerServiceHandler(Object handler);

    public void registerServiceHandler(String name,
                           Object handler);

    public boolean removeAttribute(String name);
```

## Package org.?red5.?server.?so

```
public void removeAttributes();

public void removeSharedObjectListener(org.red5.server.api.so.ISharedObjectListener listener);

public void sendMessage(String handler,
                      java.util.List arguments);

public boolean setAttribute(String name,
                           Object value);

public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);

public void setAttributes(org.red5.server.api.IAttributeStore values);

public void unlock();

public void unregisterServiceHandler(String name);

// Protected Methods

protected void notifyClear();

protected void notifyConnect();

protected void notifyDelete(String key);

protected void notifyDisconnect();

protected void notifySendMessage(String method,
                               java.util.List<?> params);

protected void notifyUpdate(String key,
                           Object value);

protected void notifyUpdate(String key,
                           java.util.Map<java.lang.String, java.lang.Object> value);

}
```

**Methods inherited from org.red5.server.so.SharedObject:** acquire , beginUpdate , checkRelease , clear , close , deserialize , endUpdate , getActiveListeners , getAttribute , getCreationTime , getData , getLastModified , getListeners , getMaxListeners , getName , getPath , getStore , getTotalChanges , getTotalDeletes , getTotalListeners , getTotalSends , getType , getVersion , isAcquired , isPersistent , isPersistentObject , notifyModified , register , release , removeAttribute , removeAttributes , returnAttributeValue , returnError , sendMessage , sendUpdates , serialize , setAttribute , setAttributes , setName , setPath , setPersistent , setStore , unregister

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute , getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute , getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute , hasAttribute

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.so.SharedObject:** acquireCount , changeStats , deleteStats , lastModified , listeners , listenerStats , log , modified , name , ownerMessage , path , persistent , persistentSO , sendStats , source , storage , syncEvents , updateCounter , version

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 1.2. ClientSharedObject(String, boolean)

```
public ClientSharedObject(String name,
                         boolean persistent);
```

Create new client SO with

Parameters	
name	Shared Object name
persistent	Persistence flag

## 1.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 1.4. addSharedObjectListener(ISharedObjectListener)

```
public void addSharedObjectListener(org.red5.server.api.so.ISharedObjectListener listener);
```

## 1.5. beginUpdate()

```
public void beginUpdate();
```

## 1.6. beginUpdate(IEventListener)

```
public void beginUpdate(org.red5.server.api.event.IEventListener listener);
```

## 1.7. clear()

```
public boolean clear();
```

## 1.8. close()

```
public void close();
```

## 1.9. connect(IConnection)

```
public void connect(org.red5.server.api.IConnection conn);
```

**Specified by:** Method connect in interface IClientSharedObject

Connect the shared object using the passed connection.

#### Parameters

conn	Attach SO to given connection
------	-------------------------------

## 1.10. disconnect()

```
public void disconnect();
```

**Specified by:** Method disconnect in interface IClientSharedObject

Disconnect the shared object.

## 1.11. dispatchEvent(IEvent)

```
public void dispatchEvent(org.red5.server.api.event.IEvent e);
```

**Specified by:** Method dispatchEvent in interface IEventDispatcher

Dispatches event

## 1.12. endUpdate()

```
public void endUpdate();
```

## 1.13. getAttribute(String, Object)

```
public Object getAttribute(String name,
                           Object defaultValue);
```

## 1.14. getServiceHandler(String)

```
public Object getServiceHandler(String name);
```

## 1.15. getServiceHandlerNames()

```
public java.util.Set<java.lang.String> getServiceHandlerNames();
```

## 1.16. isConnected()

```
public boolean isConnected();
```

**Specified by:** Method isConnected in interface IClientSharedObject

Check if the shared object is connected to the server.

## 1.17. isLocked()

```
public boolean isLocked();
```

## 1.18. lock()

```
public void lock();
```

## 1.19. notifyClear()

```
protected void notifyClear();
```

Notify listeners on clear

## 1.20. notifyConnect()

```
protected void notifyConnect();
```

Notify listeners on event

## 1.21. notifyDelete(String)

```
protected void notifyDelete(String key);
```

Notify listeners on attribute delete

Parameters	
key	Attribute name

## 1.22. notifyDisconnect()

```
protected void notifyDisconnect();
```

Notify listeners on disconnect

## 1.23. notifySendMessage(String, List<?>)

```
protected void notifySendMessage(String method,
                               java.util.List<?> params);
```

Broadcast send event to listeners

Parameters	
method	Method name
params	Params

## 1.24. notifyUpdate(String, Map<String, Object>)

```
protected void notifyUpdate(String key,
                           java.util.Map<java.lang.String, java.lang.Object> value);
```

Notify listeners on map attribute update

**Parameters**

key	Updated attribute key
value	Updated attribute value

**1.25. notifyUpdate(String, Object)**

```
protected void notifyUpdate(String key,
                           Object value);
```

Notify listeners on update

**Parameters**

key	Updated attribute key
value	Updated attribute value

**1.26. registerServiceHandler(Object)**

```
public void registerServiceHandler(Object handler);
```

**1.27. registerServiceHandler(String, Object)**

```
public void registerServiceHandler(String name,
                                   Object handler);
```

**1.28. removeAttribute(String)**

```
public boolean removeAttribute(String name);
```

**1.29. removeAttributes()**

```
public void removeAttributes();
```

**1.30. removeSharedObjectListener(ISharedObjectListener)**

```
public void removeSharedObjectListener(org.red5.server.api.so.ISharedObjectListener listener);
```

**1.31. sendMessage(String, List)**

```
public void sendMessage(String handler,
                       java.util.List arguments);
```

**1.32. setAttribute(String, Object)**

```
public boolean setAttribute(String name,
                           Object value);
```

**1.33. setAttributes(IAttributeStore)**

```
public void setAttributes(org.red5.server.api.IAttributeStore values);
```

## 1.34. setAttributes(Map<String, Object>)

```
public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);
```

## 1.35. unlock()

```
public void unlock();
```

## 1.36. unregisterServiceHandler(String)

```
public void unregisterServiceHandler(String name);
```

# 2. Class FlexSharedObjectMessage

```
public class FlexSharedObjectMessage extends org.?red5.?server.?so.?SharedObjectMessage {
    // Public Constructors

    public FlexSharedObjectMessage();

    public FlexSharedObjectMessage(String name,
                                  int version,
                                  boolean persistent);

    public FlexSharedObjectMessage(org.red5.server.api.event.IEventListener source,
                                  String name,
                                  int version,
                                  boolean persistent);

    // Public Methods

    public byte getDataType();

}
```

**Methods inherited from org.red5.server.so.SharedObjectMessage:** addEvent, addEvents, clear, getDataType, getEvents, getName, getObject, getType, getVersion, isEmpty, isPersistent, readExternal, releaseInternal, setIsPersistent, setName, setVersion, toString, writeExternal

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** getHeader, getSource, getTimestamp, hasSource, release, retain, setHeader, setSource, setTimestamp

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** header, object, refcount, source, timestamp

## 2.1. FlexSharedObjectMessage(IEventListener, String, int, boolean)

```
public FlexSharedObjectMessage(org.red5.server.api.event.IEventListener source,
                           String name,
```

```
    int version,
    boolean persistent);
```

Creates Flex2 Shared Object event with given listener, name, SO version and persistence flag

#### Parameters

source	Event listener
--------	----------------

name	Event name
------	------------

version	SO version
---------	------------

persistent	SO persistence flag
------------	---------------------

## 2.2. FlexSharedObjectMessage(String, int, boolean)

```
public FlexSharedObjectMessage(String name,
                               int version,
                               boolean persistent);
```

Creates Flex2 Shared Object event with given name, version and persistence flag

#### Parameters

name	Event name
------	------------

version	SO version
---------	------------

persistent	SO persistence flag
------------	---------------------

## 2.3. getDataType()

```
public byte getDataType();
```

## 3. Interface ISharedObjectEvent

One update event for a shared object received through a connection.

### 3.1. Synopsis

```
public interface ISharedObjectEvent {
// Public Methods

    public String getKey();

    public org.red5.server.so.ISharedObjectEvent.Type getType();

    public Object getValue();

}
```

### 3.2. getKey()

```
public String getKey();
```

Returns the key of the event. Depending on the type this contains:

- the attribute name to set for SET\_ATTRIBUTE
- the attribute name to delete for DELETE\_ATTRIBUTE
- the handler name to call for SEND\_MESSAGE

In all other cases the key is `null`.

#### Parameters

<i>return</i>	the key of the event
---------------	----------------------

### 3.3. getType()

```
public org.red5.server.so.ISharedObjectEvent.Type getType();
```

Returns the type of the event.

#### Parameters

<i>return</i>	the type of the event.
---------------	------------------------

### 3.4. getValue()

```
public Object getValue();
```

Returns the value of the event. Depending on the type this contains:

- the attribute value to set for SET\_ATTRIBUTE
- a list of parameters to pass to the handler for SEND\_MESSAGE

In all other cases the value is `null`.

#### Parameters

<i>return</i>	the value of the event
---------------	------------------------

## 4. Class ISharedObjectEvent.Type

```
public static final class ISharedObjectEvent.Type extends java.lang.Enum {
// Public Static Fields

    public static final org.red5.server.so.ISharedObjectEvent.Type CLIENT_CLEAR_DATA ;

    public static final org.red5.server.so.ISharedObjectEvent.Type CLIENT_DELETE_ATTRIBUTE ;
```

```

public static final org.red5.server.so.ISharedObjectEvent.Type CLIENT_DELETE_DATA ;

public static final org.red5.server.so.ISharedObjectEvent.Type CLIENT_INITIAL_DATA ;

public static final org.red5.server.so.ISharedObjectEvent.Type CLIENT_SEND_MESSAGE ;

public static final org.red5.server.so.ISharedObjectEvent.Type CLIENT_STATUS ;

public static final org.red5.server.so.ISharedObjectEvent.Type CLIENT_UPDATE_ATTRIBUTE ;

public static final org.red5.server.so.ISharedObjectEvent.Type CLIENT_UPDATE_DATA ;

public static final org.red5.server.so.ISharedObjectEvent.Type SERVER_CONNECT ;

public static final org.red5.server.so.ISharedObjectEvent.Type SERVER_DELETE_ATTRIBUTE ;

public static final org.red5.server.so.ISharedObjectEvent.Type SERVER_DISCONNECT ;

public static final org.red5.server.so.ISharedObjectEvent.Type SERVER_SEND_MESSAGE ;

public static final org.red5.server.so.ISharedObjectEvent.Type SERVER_SET_ATTRIBUTE ;

// Public Static Methods

public static org.red5.server.so.ISharedObjectEvent.Type valueOf(String name);

public static org.red5.server.so.ISharedObjectEvent.Type[] values();

}

```

**Methods inherited from java.lang.Enum:** clone, compareTo, equals, finalize, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

**Methods inherited from java.lang.Object:** getClass, notify, notifyAll, wait

## 5. Interface ISharedObjectMessage

Shared object message

### 5.1. Synopsis

```

public interface ISharedObjectMessage extends, org.?red5.?server.?net.?rtmp.?event.?IRTMPEvent {
// Public Methods

public void addEvent(ISharedObjectEvent event);

public void addEvent(org.red5.server.so.ISharedObjectEvent.Type type,
                    String key,
                    Object value);

public void clear();

public java.util.Queue<org.red5.server.so.ISharedObjectEvent> getEvents();

```

```

    public String getName();

    public int getVersion();

    public boolean isEmpty();

    public boolean isPersistent();

}

```

## 5.2. addEvent(ISharedObjectEvent.Type, String, Object)

```

public void addEvent(org.red5.server.so.ISharedObjectEvent.Type type,
                     String key,
                     Object value);

```

Addition event handler

### Parameters

type	Event type
key	Handler key
value	Event value (like arguments)

## 5.3. addEvent(ISharedObjectEvent)

```

public void addEvent(ISharedObjectEvent event);

```

Add event handler

### Parameters

event	SO event
-------	----------

## 5.4. clear()

```

public void clear();

```

Clear shared object

## 5.5. getEvents()

```

public java.util.Queue<org.red5.server.so.ISharedObjectEvent> getEvents();

```

Returns a set of ISharedObjectEvent objects containing informations what to change.

### Parameters

return	set of ISharedObjectEvents
--------	----------------------------

## 5.6. getName()

```
public String getName();
```

Returns the name of the shared object this message belongs to.

### Parameters

<i>return</i>	name of the shared object
---------------	---------------------------

## 5.7. getVersion()

```
public int getVersion();
```

Returns the version to modify.

### Parameters

<i>return</i>	version to modify
---------------	-------------------

## 5.8. isEmpty()

```
public boolean isEmpty();
```

Is empty?

### Parameters

<i>return</i>	true if shared object is empty, false otherwise
---------------	---

## 5.9. isPersistent()

```
public boolean isPersistent();
```

Does the message affect a persistent shared object?

### Parameters

<i>return</i>	true if a persistent shared object should be updated otherwise false
---------------	---

# 6. Class SharedObject

Represents shared object on server-side. Shared Objects in Flash are like cookies that are stored on client side. In Red5 and Flash Media Server there's one more special type of SOs : remote Shared Objects. These are shared by multiple clients and synchronized between them automatically on each data change. This is done asynchronously, used as events handling and is widely used in multiplayer Flash online games. Shared object can be persistent or transient. The difference is that first are saved to the disk and can be accessed later on next connection, transient objects are not saved and get lost each time they last client disconnects from it. Shared Objects has name identifiers and

path on server's HD (if persistent). On deeper level server-side Shared Object in this implementation actually uses IPersistenceStore to delegate all (de)serialization work. SOs store data as simple map, that is, "name-value" pairs. Each value in turn can be complex object or map. All access to methods that change properties in the SO must be properly synchronized for multi-threaded access.

## 6.1. Synopsis

```

public class SharedObject extends, org.?red5.?server.?AttributeStore
    implements, org.?red5.?server.?api.?statistics.?ISharedObjectStatistics, org.?red5.?server.?api.?p
// Protected Fields

    protected java.util.concurrent.atomic.AtomicInteger acquireCount ;

    protected java.util.concurrent.atomic.AtomicInteger changeStats ;

    protected java.util.concurrent.atomic.AtomicInteger deleteStats ;

    protected long lastModified ;

    protected org.red5.server.api.statistics.support.StatisticsCounter listenerStats ;

    protected java.util.Set<org.red5.server.api.event.IEventListener> listeners ;

    protected static org.slf4j.Logger log ;

    protected boolean modified ;

    protected String name ;

    protected SharedObjectMessage ownerMessage ;

    protected String path ;

    protected boolean persistent ;

    protected boolean persistentSO ;

    protected java.util.concurrent.atomic.AtomicInteger sendStats ;

    protected org.red5.server.api.event.IEventListener source ;

    protected org.red5.server.api.persistence.IPersistenceStore storage ;

    protected java.util.concurrent.ConcurrentLinkedQueue<org.red5.server.so.ISharedObjectEvent> syncEv
// Public Constructors

    public SharedObject();

    public SharedObject(java.util.Map<java.lang.String, java.lang.Object> data,
                      String name,

```

## Package org.?red5.?server.?so

```
String path,  
boolean persistent);  
  
public SharedObject(java.util.Map<java.lang.String, java.lang.Object> data,  
String name,  
String path,  
boolean persistent,  
org.red5.server.api.persistence.IPersistenceStore storage);  
  
public SharedObject(org.red5.io.object.Input input)  
throws IOException;  
  
// Public Methods  
  
public void acquire();  
  
public void deserialize(org.red5.io.object.Input input)  
throws IOException;  
  
public int getActiveListeners();  
  
public Object getAttribute(String name,  
Object value);  
  
public long getCreationTime();  
  
public java.util.Map<java.lang.String, java.lang.Object> getData();  
  
public long getLastModified();  
  
public java.util.Set<org.red5.server.api.event.IEventListener> getListeners();  
  
public int getMaxListeners();  
  
public String getName();  
  
public String getPath();  
  
public org.red5.server.api.persistence.IPersistenceStore getStore();  
  
public int getTotalChanges();  
  
public int getTotalDeletes();  
  
public int getTotalListeners();  
  
public int getTotalSends();  
  
public String getType();  
  
public int getVersion();  
  
public boolean isAcquired();  
  
public boolean isPersistent();  
  
public boolean isPersistentObject();
```

## Package org.?red5.?server.?so

```
public void release();

public boolean removeAttribute(String name);

public void removeAttributes();

public void serialize(org.red5.io.object.Output output)
throws IOException;

public boolean setAttribute(String name,
                           Object value);

public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);

public void setAttributes(org.red5.server.api.IAttributeStore values);

public void setName(String name);

public void setPath(String path);

public void setPersistent(boolean persistent);

public void setStore(org.red5.server.api.persistence.IPersistenceStore store);

// Protected Methods

protected void beginUpdate();

protected void beginUpdate(org.red5.server.api.event.IEventListener listener);

protected void checkRelease();

protected boolean clear();

protected void close();

protected void endUpdate();

protected void notifyModified();

protected void register(org.red5.server.api.event.IEventListener listener);

protected void returnAttributeValue(String name);

protected void returnError(String message);

protected void sendMessage(String handler,
                         java.util.List<?> arguments);

protected void sendUpdates();

protected void unregister(org.red5.server.api.event.IEventListener listener);

}
```

**Direct known subclasses:** org.?red5.?server.?so.?ClientSharedObject

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttribute ,  
getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute ,  
getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute ,  
getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute ,  
hasAttribute , removeAttribute , removeAttributes , setAttribute , setAttributes

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 6.2. SharedObject()

```
public SharedObject();
```

Constructs a new SharedObject.

## 6.3. SharedObject(Input)

```
public SharedObject(org.red5.io.object.Input input)  
throws IOException;
```

Constructs new SO from Input object

### Parameters

input	Input source
-------	--------------

IOException

I/O exception

### See Also

[org.red5.io.object.Input](#)

## 6.4. SharedObject(Map<String, Object>, String, String, boolean)

```
public SharedObject(java.util.Map<java.lang.String, java.lang.Object> data,  
String name,  
String path,  
boolean persistent);
```

Creates new SO from given data map, name, path and persistence option

### Parameters

data	Data
name	SO name
path	SO path
persistent	SO persistence

## 6.5. SharedObject(Map<String, Object>, String, String, boolean, IPersistenceStore)

```
public SharedObject(java.util.Map<java.lang.String, java.lang.Object> data,
                   String name,
                   String path,
                   boolean persistent,
                   org.red5.server.api.persistence.IPersistenceStore storage);
```

Creates new SO from given data map, name, path, storage object and persistence option

### Parameters

data	Data
name	SO name
path	SO path
persistent	SO persistence
storage	Persistence storage

## 6.6. acquireCount

```
protected java.util.concurrent.atomic.AtomicInteger acquireCount ;
```

Number of times the SO has been acquired

## 6.7. changeStats

```
protected java.util.concurrent.atomic.AtomicInteger changeStats ;
```

Counts number of "change" events.

## 6.8. deleteStats

```
protected java.util.concurrent.atomic.AtomicInteger deleteStats ;
```

Counts number of "delete" events.

## 6.9. lastModified

```
protected long lastModified ;
```

Last modified timestamp

## 6.10. listeners

```
protected java.util.Set<org.red5.server.api.event.IEventListener> listeners ;
```

Listeners

## 6.11. listenerStats

```
protected org.red5.server.api.statistics.support.StatisticsCounter listenerStats ;
```

Manages listener statistics.

## 6.12. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 6.13. modified

```
protected boolean modified ;
```

Has changes? flag

## 6.14. name

```
protected String name ;
```

Shared Object name (identifier)

## 6.15. ownerMessage

```
protected SharedObjectMessage ownerMessage ;
```

Owner event

## 6.16. path

```
protected String path ;
```

SO path

## 6.17. persistent

```
protected boolean persistent ;
```

true if the SharedObject was stored by the persistence framework (NOT in database, just plain serialization to the disk) and can be used later on reconnection

## 6.18. persistentSO

```
protected boolean persistentSO ;
```

true if the client / server created the SO to be persistent

## 6.19. sendStats

```
protected java.util.concurrent.atomic.AtomicInteger sendStats ;
```

Counts number of "send message" events.

## 6.20. source

```
protected org.red5.server.api.event.IEventListener source ;
```

Event listener, actually RTMP connection

## 6.21. storage

```
protected org.red5.server.api.persistence.IPersistenceStore storage ;
```

Object that is delegated with all storage work for persistent SOs

## 6.22. syncEvents

```
protected java.util.concurrent.ConcurrentLinkedQueue<org.red5.server.so.ISharedObjectEvent> syncEvents ;
```

Synchronization events

## 6.23. updateCounter

```
protected java.util.concurrent.atomic.AtomicInteger updateCounter ;
```

Number of pending update operations

## 6.24. version

```
protected java.util.concurrent.atomic.AtomicInteger version ;
```

Version. Used on synchronization purposes.

## 6.25. acquire()

```
public void acquire();
```

Prevent shared object from being released. Each call to `acquire` must be paired with a call to `release` so the SO isn't held forever. This is only valid for non-persistent SOs.

## 6.26. beginUpdate()

```
protected void beginUpdate();
```

Begin update of this Shared Object. Increases number of pending update operations

## 6.27. beginUpdate(IEventListener)

```
protected void beginUpdate(org.red5.server.api.event.IEventListener listener);
```

Begin update of this Shared Object and setting listener

### Parameters

listener	Update with listener
----------	----------------------

## 6.28. checkRelease()

```
protected void checkRelease();
```

Check if shared object must be released.

## 6.29. clear()

```
protected boolean clear();
```

Deletes all the attributes and sends a clear event to all listeners. The persistent data object is also removed from a persistent shared object.

### Parameters

<i>return</i>	true on success, false otherwise
---------------	----------------------------------

## 6.30. close()

```
protected void close();
```

Detaches a reference from this shared object, reset it's state, this will destroy the reference immediately. This is useful when you don't want to proxy a shared object any longer.

## 6.31. deserialize(Input)

```
public void deserialize(org.red5.io.object.Input input)
throws IOException;
```

**Specified by:** Method deserialize in interface IPersistable

Load the object from the passed input stream.

## 6.32. endUpdate()

```
protected void endUpdate();
```

End update of this Shared Object. Decreases number of pending update operations and broadcasts modified event if it is equal to zero (i.e. no more pending update operations).

## 6.33. getActiveListeners()

```
public int getActiveListeners();
```

**Specified by:** Method getActiveListeners in interface ISharedObjectStatistics

Return current number of subscribed listeners.

## 6.34. getAttribute(String, Object)

```
public Object getAttribute(String name,
                           Object value);
```

Return attribute by name and set if it doesn't exist yet.

### Parameters

<i>name</i>	Attribute name
<i>value</i>	Value to set if attribute doesn't exist
<i>return</i>	Attribute value

## 6.35. getCreationTime()

```
public long getCreationTime();
```

## 6.36. getData()

```
public java.util.Map<java.lang.String, java.lang.Object> getData();
```

Getter for data.

### Parameters

<i>return</i>	SO data as unmodifiable map
---------------	-----------------------------

## 6.37. getLastModified()

```
public long getLastModified();
```

**Specified by:** Method getLastModified in interface IPersistable

Returns the timestamp when the object was last modified.

## 6.38. getListeners()

```
public java.util.Set<org.red5.server.api.event.IEventListener> getListeners();
```

Get event listeners.

### Parameters

<i>return</i>	Value for property 'listeners'.
---------------	---------------------------------

## 6.39. getMaxListeners()

```
public int getMaxListeners();
```

**Specified by:** Method getMaxListeners in interface ISharedObjectStatistics

Return maximum number of concurrent subscribed listeners.

## 6.40. getName()

```
public String getName();
```

**Specified by:** Method getName in interface ISharedObjectStatistics

Return the name of the shared object.

## 6.41. getPath()

```
public String getPath();
```

**Specified by:** Method getPath in interface IPersistable

Returns the path of the persistent object.

## 6.42. getStore()

```
public org.red5.server.api.persistence.IPersistenceStore getStore();
```

**Specified by:** Method getStore in interface IPersistable

Returns the persistence store this object is stored in

## 6.43. getTotalChanges()

```
public int getTotalChanges();
```

**Specified by:** Method getTotalChanges in interface ISharedObjectStatistics

Return number of attribute changes.

## 6.44. getTotalDeletes()

```
public int getTotalDeletes();
```

**Specified by:** Method getTotalDeletes in interface ISharedObjectStatistics

Return number of attribute deletes.

## 6.45. getTotalListeners()

```
public int getTotalListeners();
```

**Specified by:** Method getTotalListeners in interface ISharedObjectStatistics

Return total number of subscribed listeners.

## 6.46. getTotalSends()

```
public int getTotalSends();
```

**Specified by:** Method getTotalSends in interface ISharedObjectStatistics

Return number of times a message was sent.

## 6.47. getType()

```
public String getType();
```

**Specified by:** Method `getType` in interface `IPersistable`

Returns the type of the persistent object.

## 6.48. getVersion()

```
public int getVersion();
```

**Specified by:** Method `getVersion` in interface `ISharedObjectStatistics`

Getter for version.

### Parameters

<i>return</i>	SO version.
---------------	-------------

## 6.49. isAcquired()

```
public boolean isAcquired();
```

Check if shared object currently is acquired.

### Parameters

<i>return</i>	<code>true</code> if the SO is acquired, otherwise <code>false</code>
---------------	---

## 6.50. isPersistent()

```
public boolean isPersistent();
```

**Specified by:** Method `isPersistent` in interface `IPersistable`

Returns `true` if the object is persistent, `false` otherwise.

## 6.51. isPersistentObject()

```
public boolean isPersistentObject();
```

**Specified by:** Method `isPersistentObject` in interface `ISharedObjectStatistics`

Getter for persistent object

### Parameters

<i>return</i>	Persistent object
---------------	-------------------

## 6.52. notifyModified()

```
protected void notifyModified();
```

Send notification about modification of SO

## 6.53. register(IEventListener)

```
protected void register(org.red5.server.api.event.IEventListener listener);
```

Register event listener

### Parameters

listener	Event listener
----------	----------------

## 6.54. release()

```
public void release();
```

Release previously acquired shared object. If the SO is non-persistent, no more clients are connected the SO isn't acquired any more, the data is released.

## 6.55. removeAttribute(String)

```
public boolean removeAttribute(String name);
```

Removes attribute with given name

### Parameters

name	Attribute
<i>return</i>	<code>true</code> if there's such an attribute and it was removed, <code>false</code> otherwise

## 6.56. removeAttributes()

```
public void removeAttributes();
```

Remove all attributes (clear Shared Object)

## 6.57. returnAttributeValue(String)

```
protected void returnAttributeValue(String name);
```

Return an attribute value to the owner.

### Parameters

name
------

## 6.58. returnError(String)

```
protected void returnError(String message);
```

Return an error message to the client.

### Parameters

message	
---------	--

## 6.59. sendMessage(String, List<?>)

```
protected void sendMessage(String handler,
                          java.util.List<?> arguments);
```

Broadcast event to event handler

Parameters	
handler	Event handler
arguments	Arguments

## 6.60. sendUpdates()

```
protected void sendUpdates();
```

Send update notification over data channel of RTMP connection

## 6.61. serialize(Output)

```
public void serialize(org.red5.io.object.Output output)
                      throws IOException;
```

**Specified by:** Method serialize in interface IPersistable

Write the object to the passed output stream.

## 6.62. setAttribute(String, Object)

```
public boolean setAttribute(String name,
                           Object value);
```

Set value of attribute with given name

Parameters	
name	Attribute name
value	Attribute value
return	true if there's such attribute and value was set, false otherwise

## 6.63. setAttributes(IAttributeStore)

```
public void setAttributes(org.red5.server.api.IAttributeStore values);
```

Set attributes as attributes store.

Parameters	
------------	--

values	Attributes.
--------	-------------

## 6.64. setAttributes(Map<String, Object>)

```
public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);
```

Set attributes as map.

### Parameters

values	Attributes.
--------	-------------

## 6.65. setName(String)

```
public void setName(String name);
```

**Specified by:** Method setName in interface IPersistable

Set the name of the persistent object.

## 6.66. setPath(String)

```
public void setPath(String path);
```

**Specified by:** Method setPath in interface IPersistable

Set the path of the persistent object.

## 6.67. setPersistent(boolean)

```
public void setPersistent(boolean persistent);
```

**Specified by:** Method setPersistent in interface IPersistable

Set the persistent flag of the object.

## 6.68. setStore(IPersistenceStore)

```
public void setStore(org.red5.server.api.persistence.IPersistenceStore store);
```

**Specified by:** Method setStore in interface IPersistable

Store a reference to the persistence store in the object.

## 6.69. unregister(IEventListener)

```
protected void unregister(org.red5.server.api.event.IEventListener listener);
```

Unregister event listener

### Parameters

listener

Event listener

## 7. Class SharedObjectEvent

```

public class SharedObjectEvent implements, org.?red5.?server.?so.?ISharedObjectEvent, java.?io.?Externalizable
// Public Constructors

    public SharedObjectEvent();

    public SharedObjectEvent(org.red5.server.so.ISharedObjectEvent.Type type,
                           String key,
                           Object value);

// Public Methods

    public String getKey();

    public org.red5.server.so.ISharedObjectEvent.Type getType();

    public Object getValue();

    public void readExternal(java.io.ObjectInput in)
                           throws IOException, ClassNotFoundException;

    public String toString();

    public void writeExternal(java.io.ObjectOutput out)
                           throws IOException;

}

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

### 7.1. SharedObjectEvent(ISharedObjectEvent.Type, String, Object)

```

public SharedObjectEvent(org.red5.server.so.ISharedObjectEvent.Type type,
                       String key,
                       Object value);

```

### 7.2. getKey()

```

public String getKey();

```

**Specified by:** Method getKey in interface ISharedObjectEvent

Returns the key of the event. Depending on the type this contains:

- the attribute name to set for SET\_ATTRIBUTE
- the attribute name to delete for DELETE\_ATTRIBUTE
- the handler name to call for SEND\_MESSAGE

In all other cases the key is `null`.

### 7.3. `getType()`

```
public org.red5.server.so.ISharedObjectEvent.Type getType();
```

**Specified by:** Method `getType` in interface `ISharedObjectEvent`

Returns the type of the event.

### 7.4. `getValue()`

```
public Object getValue();
```

**Specified by:** Method `getValue` in interface `ISharedObjectEvent`

Returns the value of the event. Depending on the type this contains:

- the attribute value to set for `SET_ATTRIBUTE`
- a list of parameters to pass to the handler for `SEND_MESSAGE`

In all other cases the value is `null`.

### 7.5. `toString()`

```
public String toString();
```

## 8. Class SharedObjectMessage

Shared object event

### 8.1. Synopsis

```
public class SharedObjectMessage extends, org.?red5.?server.?net.?rtmp.?event.?BaseEvent
    implements, org.?red5.?server.?so.?ISharedObjectMessage {
// Public Constructors

    public SharedObjectMessage();

    public SharedObjectMessage(String name,
                                int version,
                                boolean persistent);

    public SharedObjectMessage(org.red5.server.api.event.IEventListener source,
                                String name,
                                int version,
                                boolean persistent);

// Public Methods

    public void addEvent(ISharedObjectEvent event);

    public void addEvent(org.red5.server.so.ISharedObjectEvent.Type type,
                        String key,
                        Object value);
```

## Package org.?red5.?server.?so

```
public void addEvents(java.util.List<org.red5.server.so.ISharedObjectEvent> events);

public void addEvents(java.util.Queue<org.red5.server.so.ISharedObjectEvent> events);

public void clear();

public byte getDataType();

public java.util.concurrent.ConcurrentLinkedQueue<org.red5.server.so.ISharedObjectEvent> getEvents();

public String getName();

public Object getObject();

public org.red5.server.api.event.IEvent.Type getType();

public int getVersion();

public boolean isEmpty();

public boolean isPersistent();

public void readExternal(java.io.ObjectInput in)
    throws IOException, ClassNotFoundException;

public String toString();

public void writeExternal(java.io.ObjectOutput out)
    throws IOException;

// Protected Methods

protected void releaseInternal();

protected void setIsPersistent(boolean persistent);

protected void setName(String name);

protected void setVersion(int version);

}
```

**Direct known subclasses:** org.?red5.?server.?so.?FlexSharedObjectMessage

**Methods inherited from org.red5.server.net.rtmp.event.BaseEvent:** getDataType ,  
getHeader , getObject , getSource , getTimestamp , getType , hasSource ,  
readExternal , release , releaseInternal , retain , setHeader , setSource ,  
setTimestamp , writeExternal

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.net.rtmp.event.BaseEvent:** header , object ,  
refcount , source , timestamp

## 8.2. SharedObjectMessage(IEventListener, String, int, boolean)

```
public SharedObjectMessage(org.red5.server.api.event.IEventListener source,
                           String name,
                           int version,
                           boolean persistent);
```

Creates Shared Object event with given listener, name, SO version and persistence flag

### Parameters

source	Event listener
name	Event name
version	SO version
persistent	SO persistence flag

## 8.3. SharedObjectMessage(String, int, boolean)

```
public SharedObjectMessage(String name,
                           int version,
                           boolean persistent);
```

Creates Shared Object event with given name, version and persistence flag

### Parameters

name	Event name
version	SO version
persistent	SO persistence flag

## 8.4. addEvent(ISharedObjectEvent.Type, String, Object)

```
public void addEvent(org.red5.server.so.ISharedObjectEvent.Type type,
                     String key,
                     Object value);
```

**Specified by:** Method addEvent in interface ISharedObjectMessage

Addition event handler

## 8.5. addEvent(ISharedObjectEvent)

```
public void addEvent(ISharedObjectEvent event);
```

**Specified by:** Method addEvent in interface ISharedObjectMessage

Add event handler

## 8.6. clear()

```
public void clear();
```

**Specified by:** Method clear in interface ISharedObjectMessage

Clear shared object

## 8.7. getDataType()

```
public byte getDataType();
```

## 8.8. getEvents()

```
public java.util.concurrent.ConcurrentLinkedQueue<org.red5.server.so.ISharedObjectEvent> getEvents()
```

**Specified by:** Method getEvents in interface ISharedObjectMessage

Returns a set of ISharedObjectEvent objects containing informations what to change.

## 8.9. getName()

```
public String getName();
```

**Specified by:** Method getName in interface ISharedObjectMessage

Returns the name of the shared object this message belongs to.

## 8.10. getObject()

```
public Object getObject();
```

## 8.11. getType()

```
public org.red5.server.api.event.IEvent.Type getType();
```

## 8.12. getVersion()

```
public int getVersion();
```

**Specified by:** Method getVersion in interface ISharedObjectMessage

Returns the version to modify.

## 8.13. isEmpty()

```
public boolean isEmpty();
```

**Specified by:** Method isEmpty in interface ISharedObjectMessage

Is empty?

## 8.14. isPersistent()

```
public boolean isPersistent();
```

**Specified by:** Method isPersistent in interface ISharedObjectMessage

Does the message affect a persistent shared object?

## 8.15. releaseInternal()

```
protected void releaseInternal();
```

## 8.16. setIsPersistent(boolean)

```
protected void setIsPersistent(boolean persistent);
```

Setter for persistence flag

### Parameters

persistent	Persistence flag
------------	------------------

## 8.17. setName(String)

```
protected void setName(String name);
```

Setter for name

### Parameters

name	Event name
------	------------

## 8.18. setVersion(int)

```
protected void setVersion(int version);
```

Setter for version

### Parameters

version	New version
---------	-------------

## 8.19. toString()

```
public String toString();
```

# 9. Class SharedObjectScope

Special scope for shared objects

## 9.1. Synopsis

```
public class SharedObjectScope extends, org.?red5.?server.?BasicScope
    implements, org.?red5.?server.?api.?so.?ISharedObject, org.?red5.?server.?net.?rtmp.?status.?Statu
// Protected Fields

    protected SharedObject so ;

// Public Constructors

    public SharedObjectScope(org.red5.server.api.IScope parent,
                           String name,
                           boolean persistent,
```

## Package org.?red5.?server.?so

```
        org.red5.server.api.persistence.IPersistenceStore store);  
  
    // Public Methods  
  
    public void acquire();  
  
    public void addEventListener(org.red5.server.api.event.IEventListener listener);  
  
    public void addSharedObjectListener(org.red5.server.api.so.ISharedObjectListener listener);  
  
    public void beginUpdate();  
  
    public void beginUpdate(org.red5.server.api.event.IEventListener listener);  
  
    public boolean clear();  
  
    public void close();  
  
    public void dispatchEvent(org.red5.server.api.event.IEvent e);  
  
    public void endUpdate();  
  
    public Object getAttribute(String name);  
  
    public Object getAttribute(String name,  
                               Object value);  
  
    public java.util.Set<java.lang.String> getAttributeNames();  
  
    public java.util.Map<java.lang.String, java.lang.Object> getAttributes();  
  
    public java.util.Map<java.lang.String, java.lang.Object> getData();  
  
    public String getName();  
  
    public String getPath();  
  
    public Object getServiceHandler(String name);  
  
    public java.util.Set<java.lang.String> getServiceHandlerNames();  
  
    public java.util.Set<org.red5.server.api.so.ISharedObjectSecurity> getSharedObjectSecurity();  
  
    public org.red5.server.api.statistics.ISharedObjectStatistics getStatistics();  
  
    public org.red5.server.api.persistence.IPersistenceStore getStore();  
  
    public String getType();  
  
    public int getVersion();  
  
    public boolean hasAttribute(String name);  
  
    public boolean isAcquired();  
  
    public boolean isLocked();
```

## Package org.?red5.?server.?so

```
public boolean isPersistentObject();

public void lock();

public void registerServiceHandler(Object handler);

public void registerServiceHandler(String name,
                                  Object handler);

public void registerSharedObjectSecurity(org.red5.server.api.so.ISharedObjectSecurity handler);

public void release();

public boolean removeAttribute(String name);

public void removeAttributes();

public void removeEventListener(org.red5.server.api.event.IEventListener listener);

public void removeSharedObjectListener(org.red5.server.api.so.ISharedObjectListener listener);

public void sendMessage(String handler,
                      java.util.List arguments);

public boolean setAttribute(String name,
                           Object value);

public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);

public void setAttributes(org.red5.server.api.IAttributeStore values);

public void setName(String name);

public void setPath(String path);

public String toString();

public void unlock();

public void unregisterServiceHandler();

public void unregisterServiceHandler(String name);

public void unregisterSharedObjectSecurity(org.red5.server.api.so.ISharedObjectSecurity handler);

// Protected Methods

protected boolean isConnectionAllowed();

protected boolean isDeleteAllowed(String key);

protected boolean isSendAllowed(String message,
                               java.util.List<?> arguments);

protected boolean isWriteAllowed(String key,
                                Object value);
```

```
}
```

**Methods inherited from org.red5.server.BasicScope:** addEventListener , dispatchEvent , getDepth , getEventListeners , getParent , getPath , handleEvent , hasParent , iterator , notifyEvent , removeEventListener

**Methods inherited from org.red5.server.PersistableAttributeStore:** deserialize , getAttribute , getLastModified , getName , getStore , getType , isPersistent , modified , removeAttribute , removeAttributes , serialize , setAttribute , setAttributes , setName , setPath , setPersistent , setStore

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute , getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute , getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute , hasAttribute

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.BasicScope:** keepOnDisconnect , listeners , parent , persistenceClass

**Fields inherited from org.red5.server.PersistableAttributeStore:** lastModified , name , path , persistent , store , type

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 9.2. SharedObjectScope(IScope, String, boolean, IPersistenceStore)

```
public SharedObjectScope(org.red5.server.api.IScope parent,
                        String name,
                        boolean persistent,
                        org.red5.server.api.persistence.IPersistenceStore store);
```

Creates shared object with given parent scope, name, persistence flag state and store object

### Parameters

parent	Parent scope
name	Name
persistent	Persistence flag state
store	Persistence store

## 9.3. so

```
protected SharedObject so ;
```

Scoped shared object

## 9.4. acquire()

```
public void acquire();
```

**Specified by:** Method acquire in interface ISharedObject

Prevent shared object from being released. Each call to `acquire` must be paired with a call to `release` so the SO isn't held forever. This method basically is a noop for persistent SOs as their data is stored and they can be released without losing their contents.

## 9.5. addEventListener(IEventListener)

```
public void addEventListener(org.red5.server.api.event.IEventListener listener);
```

## 9.6. addSharedObjectListener(ISharedObjectListener)

```
public void addSharedObjectListener(org.red5.server.api.so.ISharedObjectListener listener);
```

## 9.7. beginUpdate()

```
public void beginUpdate();
```

## 9.8. beginUpdate(IEventListener)

```
public void beginUpdate(org.red5.server.api.event.IEventListener listener);
```

## 9.9. clear()

```
public boolean clear();
```

## 9.10. close()

```
public void close();
```

## 9.11. dispatchEvent(IEvent)

```
public void dispatchEvent(org.red5.server.api.event.IEvent e);
```

## 9.12. endUpdate()

```
public void endUpdate();
```

## 9.13. getAttribute(String)

```
public Object getAttribute(String name);
```

## 9.14. getAttribute(String, Object)

```
public Object getAttribute(String name,
                           Object value);
```

**9.15. getAttributeNames()**

```
public java.util.Set<java.lang.String> getAttributeNames();
```

**9.16. getAttributes()**

```
public java.util.Map<java.lang.String, java.lang.Object> getAttributes();
```

**9.17. getData()**

```
public java.util.Map<java.lang.String, java.lang.Object> getData();
```

**9.18. getName()**

```
public String getName();
```

**9.19. getPath()**

```
public String getPath();
```

**9.20. getServiceHandler(String)**

```
public Object getServiceHandler(String name);
```

**9.21. getServiceHandlerNames()**

```
public java.util.Set<java.lang.String> getServiceHandlerNames();
```

**9.22. getSharedObjectSecurity()**

```
public java.util.Set<org.red5.server.api.so.ISharedObjectSecurity> getSharedObjectSecurity();
```

**9.23. getStatistics()**

```
public org.red5.server.api.statistics.ISharedObjectStatistics getStatistics();
```

**Specified by:** Method getStatistics in interface ISharedObject

Return statistics about the shared object.

**9.24. getStore()**

```
public org.red5.server.api.persistence.IPersistenceStore getStore();
```

**9.25. getType()**

```
public String getType();
```

**9.26. getVersion()**

```
public int getVersion();
```

## 9.27. hasAttribute(String)

```
public boolean hasAttribute(String name);
```

## 9.28. isAcquired()

```
public boolean isAcquired();
```

**Specified by:** Method isAcquired in interface ISharedObject

Check if shared object currently is acquired.

## 9.29. isConnectionAllowed()

```
protected boolean isConnectionAllowed();
```

Call handlers and check if connection to the existing SO is allowed.

Parameters
<i>return</i>

## 9.30. isDeleteAllowed(String)

```
protected boolean isDeleteAllowed(String key);
```

Call handlers and check if deleting a property from the SO is allowed.

Parameters
<i>key</i>
<i>return</i>

## 9.31. isLocked()

```
public boolean isLocked();
```

Returns the locked state of this SharedObject.

Parameters	
<i>return</i>	true if in a locked state; false otherwise

## 9.32. isPersistentObject()

```
public boolean isPersistentObject();
```

## 9.33. isSendAllowed(String, List<?>)

```
protected boolean isSendAllowed(String message,
```

```
java.util.List<?> arguments);
```

Call handlers and check if sending a message to the clients connected to the SO is allowed.

#### Parameters

message	
arguments	
<i>return</i>	

### 9.34. isWriteAllowed(String, Object)

```
protected boolean isWriteAllowed(String key,
                                Object value);
```

Call handlers and check if writing to the SO is allowed.

#### Parameters

key	
value	
<i>return</i>	

### 9.35. lock()

```
public void lock();
```

Locks the shared object instance. Prevents any changes to this object by clients until the SharedObject.unlock() method is called.

### 9.36. registerServiceHandler(Object)

```
public void registerServiceHandler(Object handler);
```

### 9.37. registerServiceHandler(String, Object)

```
public void registerServiceHandler(String name,
                                  Object handler);
```

### 9.38. registerSharedObjectSecurity(ISharedObjectSecurity)

```
public void registerSharedObjectSecurity(org.red5.server.api.so.ISharedObjectSecurity handler);
```

### 9.39. release()

```
public void release();
```

**Specified by:** Method release in interface ISharedObject

Release previously acquired shared object. If the SO is non-persistent, no more clients are connected the SO isn't acquired any more, the data is released.

## 9.40. removeAttribute(String)

```
public boolean removeAttribute(String name);
```

## 9.41. removeAttributes()

```
public void removeAttributes();
```

## 9.42. removeEventListener(IEventListener)

```
public void removeEventListener(org.red5.server.api.event.IEventListener listener);
```

## 9.43. removeSharedObjectListener(ISharedObjectListener)

```
public void removeSharedObjectListener(org.red5.server.api.so.ISharedObjectListener listener);
```

## 9.44. sendMessage(String, List)

```
public void sendMessage(String handler,  
                      java.util.List arguments);
```

## 9.45. setAttribute(String, Object)

```
public boolean setAttribute(String name,  
                           Object value);
```

## 9.46. setAttributes(IAttributeStore)

```
public void setAttributes(org.red5.server.api.IAttributeStore values);
```

## 9.47. setAttributes(Map<String, Object>)

```
public void setAttributes(java.util.Map<java.lang.String, java.lang.Object> values);
```

## 9.48. setName(String)

```
public void setName(String name);
```

## 9.49. setPath(String)

```
public void setPath(String path);
```

## 9.50. toString()

```
public String toString();
```

## 9.51. unlock()

```
public void unlock();
```

Unlocks a shared object instance that was locked with SharedObject.lock().

## 9.52. unregisterServiceHandler(String)

```
public void unregisterServiceHandler(String name);
```

## 9.53. unregisterSharedObjectSecurity(ISharedObjectSecurity)

```
public void unregisterSharedObjectSecurity(org.red5.server.api.so.ISharedObjectSecurity handler);
```

# 10. Class SharedObjectService

Shared object service

## 10.1. Synopsis

```
public class SharedObjectService implements org.?red5.?server.?api.?so.?ISharedObjectService {
    // Public Constructors

    public SharedObjectService();

    // Public Methods

    public boolean clearSharedObjects(org.red5.server.api.IScope scope,
                                      String name);

    public boolean createSharedObject(org.red5.server.api.IScope scope,
                                      String name,
                                      boolean persistent);

    public org.red5.server.api.so.ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                                               String name);

    public org.red5.server.api.so.ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                                               String name,
                                                               boolean persistent);

    public java.util.Set<java.lang.String> getSharedObjectNames(org.red5.server.api.IScope scope);

    public boolean hasSharedObject(org.red5.server.api.IScope scope,
                                 String name);

    public void setPersistenceClassName(String name);
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 10.2. clearSharedObjects(IScope, String)

```
public boolean clearSharedObjects(org.red5.server.api.IScope scope,
                                 String name);
```

**Specified by:** Method clearSharedObjects in interface ISharedObjectService

Deletes persistent shared objects specified by name and clears all properties from active shared objects (persistent and nonpersistent). The name parameter specifies the name of a shared object, which can include a slash (/) as a delimiter between directories in the path. The last element in the path can contain wildcard patterns (for example, a question mark [?] and an asterisk [\*]) or a shared object name. The clearSharedObjects() method traverses the shared object hierarchy along the specified path and clears all the shared objects. Specifying a slash (/) clears all the shared objects associated with an application instance.

The following values are possible for the soPath parameter:

/ clears all local and persistent shared objects associated with the instance.

/foo/bar clears the shared object /foo/bar; if bar is a directory name, no shared objects are deleted.

/foo/bar/\* clears all shared objects stored under the instance directory /foo/bar. The bar directory is also deleted if no persistent shared objects are in use within this namespace.

/foo/bar/XX?? clears all shared objects that begin with XX, followed by any two characters. If a directory name matches this specification, all the shared objects within this directory are cleared.

If you call the clearSharedObjects() method and the specified path matches a shared object that is currently active, all its properties are deleted, and a "clear" event is sent to all subscribers of the shared object. If it is a persistent shared object, the persistent store is also cleared.

### 10.3. createSharedObject(IScope, String, boolean)

```
public boolean createSharedObject(org.red5.server.api.IScope scope,
                                String name,
                                boolean persistent);
```

**Specified by:** Method createSharedObject in interface ISharedObjectService

Create a new shared object.

### 10.4. getSharedObject(IScope, String)

```
public org.red5.server.api.so.ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                                               String name);
```

**Specified by:** Method getSharedObject in interface ISharedObjectService

Get a shared object by name.

### 10.5. getSharedObject(IScope, String, boolean)

```
public org.red5.server.api.so.ISharedObject getSharedObject(org.red5.server.api.IScope scope,
                                                               String name,
                                                               boolean persistent);
```

**Specified by:** Method getSharedObject in interface ISharedObjectService

Get a shared object by name and create it if it doesn't exist.

## 10.6. getSharedObjectNames(IScope)

```
public java.util.Set<java.lang.String> getSharedObjectNames(org.red5.server.api.IScope scope);
```

**Specified by:** Method getSharedObjectNames in interface ISharedObjectService

Get a set of the shared object names.

## 10.7. hasSharedObject(IScope, String)

```
public boolean hasSharedObject(org.red5.server.api.IScope scope,  
                           String name);
```

**Specified by:** Method hasSharedObject in interface ISharedObjectService

Check if a shared object exists.

## 10.8. setPersistenceClassName(String)

```
public void setPersistenceClassName(String name);
```

Setter for persistence class name.

### Parameters

name	Setter for persistence class name
------	-----------------------------------

# 1. Class StatisticsService

Implementation of the statistics service.

## 1.1. Synopsis

```
public class StatisticsService implements org.red5.server.api.statistics.IStatisticsService {
    // Public Constructors

    public StatisticsService();

    // Public Methods

    public org.red5.server.api.so.ISharedObject getScopeStatisticsSO(org.red5.server.api.IScope scope);

    public java.util.Set<java.lang.String> getScopes();

    public java.util.Set<java.lang.String> getScopes(String path)
        throws ScopeNotFoundException;

    public org.red5.server.api.so.ISharedObject getSharedObjectStatisticsSO(org.red5.server.api.IScope scope);

    public java.util.Set<org.red5.server.api.statistics.ISharedObjectStatistics> getSharedObjects(String path);

    public void setGlobalScope(org.red5.server.api.IScope scope);

    public void updateScopeStatistics(String path)
        throws ScopeNotFoundException;

    public void updateSharedObjectStatistics(String path,
                                            String name)
        throws ScopeNotFoundException, SharedObjectException;
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. getScopes()

```
public java.util.Set<java.lang.String> getScopes();
```

**Specified by:** Method getScopes in interface IStatisticsService

Return a list of all scopes that currently exist on the server.

### Parameters

return	list of scope names
--------	---------------------

**Description copied from interface: getScopes**

## 1.3. getScopes(String)

```
public java.util.Set<java.lang.String> getScopes(String path)
```

```
throws ScopeNotFoundException;
```

**Specified by:** Method getScopes in interface IStatisticsService

Return a list of all scopes that currently exist on the server below a current path.

**Parameters**

<i>path</i>	Path to start looking for scopes.
<i>return</i>	list of scope names

```
ScopeNotFoundException  
if the path on the server doesn't exist
```

**Description copied from interface: getScopes****1.4. getScopeStatisticsSO(IScope)**

```
public org.red5.server.api.so.ISharedObject getScopeStatisticsSO(org.red5.server.api.IScope scope);
```

**Specified by:** Method getScopeStatisticsSO in interface IStatisticsService

Return the shared object that will be used to keep scope statistics.

**Parameters**

<i>scope</i>	A scope to return the shared object for.
<i>return</i>	the shared object containing scope statistics

**Description copied from interface: getScopeStatisticsSO****1.5. getSharedObjects(String)**

```
public java.util.Set<org.red5.server.api.statistics.ISharedObjectStatistics> getSharedObjects(String
```

**Specified by:** Method getSharedObjects in interface IStatisticsService

Return informations about shared objects for a given scope.

**Parameters**

<i>path</i>	Path to scope to return shared object names for.
<i>return</i>	list of informations about shared objects

**Description copied from interface: getSharedObjects****1.6. getSharedObjectStatisticsSO(IScope)**

```
public org.red5.server.api.so.ISharedObject getSharedObjectStatisticsSO(org.red5.server.api.IScope s
```

**Specified by:** Method getSharedObjectStatisticsSO in interface IStatisticsService

Return the shared object that will be used to keep SO statistics.

#### Parameters

scope	A scope to return the shared object for.
<i>return</i>	the shared object containing SO statistics

**Description copied from interface:** `getSharedObjectStatisticsSO`

## 1.7. `updateScopeStatistics(String)`

```
public void updateScopeStatistics(String path)
throws ScopeNotFoundException;
```

**Specified by:** Method `updateScopeStatistics` in interface `IStatisticsService`

Update statistics for a given scope.

#### Parameters

path	Path to scope to update.
------	--------------------------

`ScopeNotFoundException`  
if the given scope doesn't exist

**Description copied from interface:** `updateScopeStatistics`

## 1.8. `updateSharedObjectStatistics(String, String)`

```
public void updateSharedObjectStatistics(String path,
                                         String name)
throws ScopeNotFoundException, SharedObjectException;
```

**Specified by:** Method `updateSharedObjectStatistics` in interface `IStatisticsService`

Update informations about a shared object in a given scope.

#### Parameters

path	Path to scope that contains the shared object.
name	Name of shared object to update.

`ScopeNotFoundException`  
if the given scope doesn't exist

`SharedObjectException`  
if no shared object with the given name exists

**Description copied from interface:** `updateSharedObjectStatistics`

## 2. Class `XmlRpcScopeStatistics`

Public methods for XML-RPC scope statistics service.

## 2.1. Synopsis

```

public class XmlRpcScopeStatistics {
    // Public Constructors

    public XmlRpcScopeStatistics();

    public XmlRpcScopeStatistics(org.red5.server.api.IScope globalScope);

    // Public Methods

    public java.util.Map<java.lang.String, java.lang.Object> getScopeAttributes();

    public java.util.Map<java.lang.String, java.lang.Object> getScopeAttributes(String path);

    public String[] getScopes();

    public String[] getScopes(String path);

    public java.util.Map<java.lang.String, java.lang.Object> getSharedObjects(String path);

    public void setGlobalScope(org.red5.server.api.IScope scope);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. XmlRpcScopeStatistics()

```
public XmlRpcScopeStatistics();
```

Constructs a new XmlScopeStatistics.

## 2.3. XmlRpcScopeStatistics(IScope)

```
public XmlRpcScopeStatistics(org.red5.server.api.IScope globalScope);
```

Create new scope statistic.

### Parameters

globalScope	Global scope ref
-------------	------------------

## 2.4. getScopeAttributes()

```
public java.util.Map<java.lang.String, java.lang.Object> getScopeAttributes();
```

Return attributes of the global scope.

### Parameters

return	The scope's attributes
--------	------------------------

## 2.5. getScopeAttributes(String)

```
public java.util.Map<java.lang.String, java.lang.Object> getScopeAttributes(String path);
```

Return attributes of a given scope.

### Parameters

<i>path</i>	Path of scope to return attributes of
<i>return</i>	The scope's attributes

## 2.6. getScopes()

```
public String[] getScopes();
```

Return available applications.

### Parameters

<i>return</i>	list of application names
---------------	---------------------------

## 2.7. getScopes(String)

```
public String[] getScopes(String path);
```

Return subscopes of another scope.

### Parameters

<i>path</i>	Path of scope to return subscopes of
<i>return</i>	List of subscope names

## 2.8. getSharedObjects(String)

```
public java.util.Map<java.lang.String, java.lang.Object> getSharedObjects(String path);
```

Return informations about shared objects of a given scope.

### Parameters

<i>path</i>	Path of scope to return shared objects for
<i>return</i>	A mapping containing the shared object name -> (persistent, data)

## 2.9. setGlobalScope(IScope)

```
public void setGlobalScope(org.red5.server.api.IScope scope);
```

Setter for global scope.

### Parameters

scope	Value to set for property 'globalScope'.
-------	--

# 1. Class AbstractClientStream

Abstract base for client streams

## 1.1. Synopsis

```
public abstract class AbstractClientStream extends, org.red5.server.stream.AbstractStream
    implements, org.red5.server.api.stream.IClientStream {
// Public Constructors

    public AbstractClientStream();

// Public Methods

    public org.red5.server.api.IBandwidthConfigure getBandwidthConfigure();

    public int getClientBufferDuration();

    public org.red5.server.api.stream.IStreamCapableConnection getConnection();

    public org.red5.server.api.IBWControllable getParentBWControllable();

    public int getStreamId();

    public void setBandwidthConfigure(org.red5.server.api.IBandwidthConfigure config);

    public void setClientBufferDuration(int duration);

    public void setConnection(org.red5.server.api.stream.IStreamCapableConnection conn);

    public void setStreamId(int streamId);

}
```

**Direct known subclasses:** org.red5.server.stream.ClientBroadcastStream , org.red5.server.stream.PlaylistSubscriberStream

**Methods inherited from org.red5.server.stream.AbstractStream:** getCodecInfo , getName , getScope , getStreamAwareHandler , setCodecInfo , setName , setScope

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.stream.AbstractStream:** state

## 1.2. getBandwidthConfigure()

```
public org.red5.server.api.IBandwidthConfigure getBandwidthConfigure();
```

Return stream bandwidth configuration

Parameters

<i>return</i>	Bandwidth config
---------------	------------------

### 1.3. getClientBufferDuration()

<code>public int getClientBufferDuration();</code>	
--	--

Get duration in ms as requested by the client.

Parameters	
------------	--

<i>return</i>	
---------------	--

### 1.4. getConnection()

<code>public org.red5.server.api.stream.IStreamCapableConnection getConnection();</code>	
--	--

**Specified by:** Method `getConnection` in interface `IClientStream`

Return connection associated with stream

Parameters	
------------	--

<i>return</i>	
---------------	--

Stream capable connection object	
----------------------------------	--

### 1.5. getParentBWControllable()

<code>public org.red5.server.api.IBWControllable getParentBWControllable();</code>	
--	--

Return parent flow controllable object (bandwidth preferences holder)

Parameters	
------------	--

<i>return</i>	
---------------	--

IFlowControllable object	
--------------------------	--

### 1.6. getStreamId()

<code>public int getStreamId();</code>	
--	--

**Specified by:** Method `getStreamId` in interface `IClientStream`

Return stream id

Parameters	
------------	--

<i>return</i>	
---------------	--

Stream id	
-----------	--

### 1.7. setBandwidthConfigure(IBandwidthConfigure)

<code>public void setBandwidthConfigure(org.red5.server.api.IBandwidthConfigure config);</code>	
---	--

Setter for bandwidth config

Parameters	
------------	--

config

Bandwidth config

## 1.8. setClientBufferDuration(int)

```
public void setClientBufferDuration(int duration);
```

**Specified by:** Method setClientBufferDuration in interface IClientStream

Set the buffer duration for this stream as requested by the client.

## 1.9. setConnection(IStreamCapableConnection)

```
public void setConnection(org.red5.server.api.stream.IStreamCapableConnection conn);
```

Setter for stream capable connection

Parameters

conn	IStreamCapableConnection object
------	---------------------------------

## 1.10. setStreamId(int)

```
public void setStreamId(int streamId);
```

Setter for stream id

Parameters

streamId	Stream id
----------	-----------

# 2. Class AbstractStream

Abstract base implementation of IStream. Contains codec information, stream name, scope, event handling meand, provides stream start and stop operations.

## 2.1. Synopsis

```
public abstract class AbstractStream implements org.?red5.?server.?api.?stream.?IStream {
    // Protected Fields

    protected org.red5.server.stream.AbstractStream.State state;

    // Public Constructors

    public AbstractStream();

    // Public Methods

    public org.red5.server.api.stream.IStreamCodecInfo getCodecInfo();

    public String getName();

    public org.red5.server.api.IScope getScope();

    public void setCodecInfo(org.red5.server.api.stream.IStreamCodecInfo codecInfo);
}
```

```

    public void setName(String name);

    public void setScope(org.red5.server.api.IScope scope);

// Protected Methods

    protected org.red5.server.api.stream.IStreamAwareScopeHandler getStreamAwareHandler();

}
```

**Direct known subclasses:** org.?red5.?server.?stream.?AbstractClientStream , org.?red5.?server.?stream.?ServerStream

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

#### See Also

[org.red5.server.api.stream.IStream](#)

## 2.2. state

```
protected org.red5.server.stream.AbstractStream.State state ;
```

Current state

## 2.3. getCodecInfo()

```
public org.red5.server.api.stream.IStreamCodecInfo getCodecInfo();
```

**Specified by:** Method getCodecInfo in interface IStream

Return codec information

#### Parameters

<i>return</i>	Stream codec information
---------------	--------------------------

## 2.4. getName()

```
public String getName();
```

**Specified by:** Method getName in interface IStream

Return stream name

#### Parameters

<i>return</i>	Stream name
---------------	-------------

## 2.5. getScope()

```
public org.red5.server.api.IScope getScope();
```

**Specified by:** Method getScope in interface IStream

Return scope

Parameters

<i>return</i>	Scope
---------------	-------

## 2.6. **getStreamAwareHandler()**

```
protected org.red5.server.api.stream.IStreamAwareScopeHandler getStreamAwareHandler();
```

Return stream aware scope handler or null if scope is null

Parameters

<i>return</i>	IStreamAwareScopeHandler implementation
---------------	---

## 2.7. **setCodecInfo(IStreamCodecInfo)**

```
public void setCodecInfo(org.red5.server.api.stream.IStreamCodecInfo codecInfo);
```

Setter for codec info

Parameters

codecInfo	Codec info
-----------	------------

## 2.8. **setName(String)**

```
public void setName(String name);
```

Setter for name

Parameters

name	Stream name
------	-------------

## 2.9. **setScope(IScope)**

```
public void setScope(org.red5.server.api.IScope scope);
```

Setter for scope

Parameters

scope	Scope
-------	-------

## 3. Class AbstractStream.State

Enumeration for states

### 3.1. Synopsis

```
protected static final class AbstractStream.State extends java.lang.Enum {
// Public Static Fields
```

```

    public static final org.red5.server.stream.AbstractStream.State CLOSED ;
    public static final org.red5.server.stream.AbstractStream.State PAUSED ;
    public static final org.red5.server.stream.AbstractStream.State PLAYING ;
    public static final org.red5.server.stream.AbstractStream.State STOPPED ;
    public static final org.red5.server.stream.AbstractStream.State UNINIT ;

    // Public Static Methods

    public static org.red5.server.stream.AbstractStream.State valueOf(String name);
    public static org.red5.server.stream.AbstractStream.State[] values();
}

```

**Methods inherited from java.lang.Enum:** clone, compareTo, equals, finalize, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

**Methods inherited from java.lang.Object:** getClass, notify, notifyAll, wait

## 4. Class BroadcastScope

Scope type for publishing that deals with pipe connection events, like async message listening in JMS

### 4.1. Synopsis

```

public class BroadcastScope extends, org.?red5.?server.?BasicScope
    implements, org.?red5.?server.?stream.?IBroadcastScope, org.?red5.?server.?messaging.?IPipeConnect
// Public Constructors

    public BroadcastScope(org.red5.server.api.IScope parent,
                        String name);

// Public Methods

    public void addPipeConnectionListener(org.red5.server.messaging.IPipeConnectionListener listener);

    public java.util.List<org.red5.server.messaging.IConsumer> getConsumers();

    public java.util.List<org.red5.server.messaging.IProvider> getProviders();

    public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);

    public org.red5.server.messaging.IMessage pullMessage();

    public org.red5.server.messaging.IMessage pullMessage(long wait);

    public void pushMessage(org.red5.server.messaging.IMessage message)
        throws IOException;

    public void removePipeConnectionListener(org.red5.server.messaging.IPipeConnectionListener listener)
}

```

```

public void sendOOBControlMessage(org.red5.server.messaging.IConsumer consumer,
                                    org.red5.server.messaging.OOBControlMessage oobCtrlMsg);

public void sendOOBControlMessage(org.red5.server.messaging.IProvider provider,
                                    org.red5.server.messaging.OOBControlMessage oobCtrlMsg);

public boolean subscribe(org.red5.server.messaging.IConsumer consumer,
                        java.util.Map paramMap);

public boolean subscribe(org.red5.server.messaging.IProvider provider,
                        java.util.Map paramMap);

public boolean unsubscribe(org.red5.server.messaging.IConsumer consumer);

public synchronized boolean unsubscribe(org.red5.server.messaging.IProvider provider);

}

```

**Methods inherited from org.red5.server.BasicScope:** addEventListener , dispatchEvent , getDepth , getEventListeners , getParent , getPath , handleEvent , hasParent , iterator , notifyEvent , removeEventListener

**Methods inherited from org.red5.server.PersistableAttributeStore:** deserialize , getAttribute , getLastModified , getName , getStore , getType , isPersistent , modified , removeAttribute , removeAttributes , serialize , setAttribute , setAttributes , setName , setPath , setPersistent , setStore

**Methods inherited from org.red5.server.AttributeStore:** filterNull , getAttributeNames , getAttributes , getBoolAttribute , getByteAttribute , getDoubleAttribute , getIntAttribute , getListAttribute , getLongAttribute , getMapAttribute , getSetAttribute , getShortAttribute , getStringAttribute , hasAttribute

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.BasicScope:** keepOnDisconnect , listeners , parent , persistenceClass

**Fields inherited from org.red5.server.PersistableAttributeStore:** lastModified , name , path , persistent , store , type

**Fields inherited from org.red5.server.AttributeStore:** attributes

## 4.2. BroadcastScope(IScope, String)

```

public BroadcastScope(org.red5.server.api.IScope parent,
                      String name);

```

Creates broadcast scope

Parameters

parent	Parent scope
name	Scope name

### 4.3. addPipeConnectionListener(IPipeConnectionListener)

```
public void addPipeConnectionListener(org.red5.server.messaging.IPipeConnectionListener listener);
```

Register pipe connection event listener with this scope's pipe. A listener that wants to listen to events when provider/consumer connects to or disconnects from a specific pipe.

Parameters	
listener	Pipe connection event listener

#### See Also

[org.red5.server.messaging.IPipeConnectionListener](#)

### 4.4. getConsumers()

```
public java.util.List<org.red5.server.messaging.IConsumer> getConsumers();
```

Getter for pipe consumers

Parameters	
return	Pipe consumers

### 4.5. getProviders()

```
public java.util.List<org.red5.server.messaging.IProvider> getProviders();
```

Getter for providers list

Parameters	
return	List of providers

### 4.6. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler

Parameters	
event	Pipe connection event

### 4.7. pullMessage()

```
public org.red5.server.messaging.IMessage pullMessage();
```

Pull message from pipe

#### Parameters

<i>return</i>	Message object
---------------	----------------

See Also

[org.red5.server.messaging.IMessage](#)

## 4.8. pullMessage(long)

```
public org.red5.server.messaging.IMessage pullMessage(long wait);
```

Pull message with timeout

#### Parameters

<i>wait</i>	Timeout
<i>return</i>	Message object

See Also

[org.red5.server.messaging.IMessage](#)

## 4.9. pushMessage(IMessage)

```
public void pushMessage(org.red5.server.messaging.IMessage message)
throws IOException;
```

Push a message to this output endpoint. May block the pusher when output can't handle the message at the time.

#### Parameters

<i>message</i>	Message to be pushed.
----------------	-----------------------

*IOException*

If message could not be pushed.

## 4.10. removePipeConnectionListener(IPipeConnectionListener)

```
public void removePipeConnectionListener(org.red5.server.messaging.IPipeConnectionListener listener)
```

Unregisters pipe connection event listener with this scope's pipe

#### Parameters

<i>listener</i>	Pipe connection event listener
-----------------	--------------------------------

See Also

[org.red5.server.messaging.IPipeConnectionListener](#)

## 4.11. sendOOBControlMessage(IConsumer, OOBControlMessage)

```
public void sendOOBControlMessage(org.red5.server.messaging.IConsumer consumer,
```

```
org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

Send out-of-band ("special") control message

Parameters	
consumer	Consumer, may be used in concrete implementations
oobCtrlMsg	Out-of-band control message

## 4.12. sendOOBControlMessage(IProvider, OOBControlMessage)

```
public void sendOOBControlMessage(org.red5.server.messaging.IProvider provider,
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

Send out-of-band ("special") control message

Parameters	
provider	Provider, may be used in concrete implementations
oobCtrlMsg	Out-of-band control message

## 4.13. subscribe(IConsumer, Map)

```
public boolean subscribe(org.red5.server.messaging.IConsumer consumer,
                       java.util.Map paramMap);
```

Connect scope's pipe to given consumer

Parameters	
consumer	Consumer
paramMap	Parameters passed with connection
<i>return</i>	true on success, false otherwise

## 4.14. subscribe(IProvider, Map)

```
public boolean subscribe(org.red5.server.messaging.IProvider provider,
                       java.util.Map paramMap);
```

Connect scope's pipe with given provider

Parameters	
provider	Provider
paramMap	Parameters passed on connection
<i>return</i>	true on success, false otherwise

## 4.15. unsubscribe(IConsumer)

```
public boolean unsubscribe(org.red5.server.messaging.IConsumer consumer);
```

Disconnects scope's pipe from given consumer

#### Parameters

consumer	Consumer
<i>return</i>	true on success, false otherwise

## 4.16. unsubscribe(IProvider)

```
public synchronized boolean unsubscribe(org.red5.server.messaging.IProvider provider);
```

Disconnects scope's pipe from given provider

#### Parameters

provider	Provider
<i>return</i>	true on success, false otherwise

# 5. Class ClientBroadcastStream

Represents live stream broadcasted from client. As Flash Media Server, Red5 supports recording mode for live streams, that is, broadcasted stream has broadcast mode. It can be either "live" or "record" and latter causes server-side application to record broadcasted stream. Note that recorded streams are recorded as FLV files. The same is correct for audio, because NellyMoser codec that Flash Player uses prohibits on-the-fly transcoding to audio formats like MP3 without paying of licensing fee or buying SDK. This type of stream uses two different pipes for live streaming and recording.

## 5.1. Synopsis

```
public class ClientBroadcastStream extends, org.?red5.?server.?stream.?AbstractClientStream
    implements, org.?red5.?server.?api.?stream.?IClientBroadcastStream, org.?red5.?server.?messaging.?

// Public Constructors

    public ClientBroadcastStream();

// Public Methods

    public void addStreamListener(org.red5.server.api.stream.IStreamListener listener);

    public void close();

    public void dispatchEvent(org.red5.server.api.event.IEvent event);

    public int getActiveSubscribers();

    public long getBytesReceived();

    public long getCreationTime();

    public int getCurrentTimestamp();

    public int getMaxSubscribers();
```

## Package org.?red5.?server.?stream

```
public org.red5.server.messaging.IProvider getProvider();

public String getPublishedName();

public String getSaveFilename();

public org.red5.server.api.statistics.IClientBroadcastStreamStatistics getStatistics();

public java.util.Collection<org.red5.server.api.stream.IStreamListener> getStreamListeners();

public int getTotalSubscribers();

public boolean isRecording();

public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                org.red5.server.messaging.IPipe pipe,
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);

public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);

public void pushMessage(org.red5.server.messaging.IPipe pipe,
                       org.red5.server.messaging.IMessage message);

public void removeStreamListener(org.red5.server.api.stream.IStreamListener listener);

public void saveAs(String name,
                   boolean isAppend)
throws IOException, ResourceNotFoundException, ResourceExistException;

public void setPublishedName(String name);

public void start();

public void startPublishing();

public void stop();

public void stopRecording();

}
```

### Methods inherited from org.red5.server.stream.AbstractClientStream:

getBandwidthConfigure , getClientBufferDuration , getConnection  
, getParentBWControllable , getStreamId , setBandwidthConfigure ,  
setClientBufferDuration , setConnection , setStreamId

### Methods inherited from org.red5.server.stream.AbstractStream: `getCodecInfo` , `getName` , `getScope` , `getStreamAwareHandler` , `setCodecInfo` , `setName` , `setScope`

### Methods inherited from `java.lang.Object`: `clone` , `equals` , `finalize` , `getClass` , `hashCode` , `notify` , `notifyAll` , `toString` , `wait`

### Fields inherited from org.red5.server.stream.AbstractStream: `state`

## 5.2. addStreamListener(IStreamListener)

```
public void addStreamListener(org.red5.server.api.stream.IStreamListener listener);
```

## 5.3. close()

```
public void close();
```

**Specified by:** Method close in interface ClientBroadcastStreamMBean

Closes stream, unsubscribes provides, sends stoppage notifications and broadcast close notification.

## 5.4. dispatchEvent(IEvent)

```
public void dispatchEvent(org.red5.server.api.event.IEvent event);
```

**Specified by:** Method dispatchEvent in interface IEventDispatcher

Dispatches event

### Parameters

event	Event to dispatch
-------	-------------------

## 5.5. getActiveSubscribers()

```
public int getActiveSubscribers();
```

**Specified by:** Method getActiveSubscribers in interface IClientBroadcastStreamStatistics

Return current number of subscribers.

## 5.6. getBytesReceived()

```
public long getBytesReceived();
```

**Specified by:** Method getBytesReceived in interface IClientBroadcastStreamStatistics

Return total number of bytes received from client for this stream.

## 5.7. getCreationTime()

```
public long getCreationTime();
```

## 5.8. getCurrentTimestamp()

```
public int getCurrentTimestamp();
```

## 5.9. getMaxSubscribers()

```
public int getMaxSubscribers();
```

**Specified by:** Method getMaxSubscribers in interface IClientBroadcastStreamStatistics

Return maximum number of concurrent subscribers.

## 5.10. getProvider()

```
public org.red5.server.messaging.IProvider getProvider();
```

**Specified by:** Method getProvider in interface ClientBroadcastStreamMBean

Getter for provider

### Parameters

<i>return</i>	Provider
---------------	----------

## 5.11. getPublishedName()

```
public String getPublishedName();
```

**Specified by:** Method getPublishedName in interface IClientBroadcastStreamStatistics

Getter for published name

### Parameters

<i>return</i>	Stream published name
---------------	-----------------------

## 5.12. getSaveFilename()

```
public String getSaveFilename();
```

**Specified by:** Method getSaveFilename in interface IClientBroadcastStreamStatistics

Get the filename the stream is being saved as.

## 5.13. getStatistics()

```
public org.red5.server.api.statistics.IClientBroadcastStreamStatistics getStatistics();
```

**Specified by:** Method getStatistics in interface IClientBroadcastStream

Return statistics about the stream.

## 5.14. getStreamListeners()

```
public java.util.Collection<org.red5.server.api.stream.IStreamListener> getStreamListeners();
```

## 5.15. getTotalSubscribers()

```
public int getTotalSubscribers();
```

**Specified by:** Method getTotalSubscribers in interface IClientBroadcastStreamStatistics

Return total number of subscribers.

## 5.16. onOOBControlMessage(IMessageComponent, IPipe, OOBControlMessage)

```
public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                org.red5.server.messaging.IPipe pipe,
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

Out-of-band control message handler

### Parameters

source	OOB message source
pipe	Pipe that used to send OOB message
oobCtrlMsg	Out-of-band control message

## 5.17. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler

### Parameters

event	Pipe connection event
-------	-----------------------

## 5.18. pushMessage(IPipe, IMessage)

```
public void pushMessage(org.red5.server.messaging.IPipe pipe,
                      org.red5.server.messaging.IMessage message);
```

**Specified by:** Method pushMessage in interface IPushableConsumer

Currently not implemented

### Parameters

pipe	Pipe
message	Message

## 5.19. removeStreamListener(IStreamListener)

```
public void removeStreamListener(org.red5.server.api.stream.IStreamListener listener);
```

## 5.20. saveAs(String, boolean)

```
public void saveAs(String name,
                   boolean isAppend)
throws IOException, ResourceNotFoundException, ResourceExistException;
```

**Specified by:** Method saveAs in interface ClientBroadcastStreamMBean

Save broadcasted stream.

**Parameters**

name	Stream name
isAppend	Append mode

`IOException`

File could not be created/written to.

`ResourceNotFoundException`

Resource doesn't exist when trying to append.

`ResourceExistException`

Resource exist when trying to create.

## 5.21. **setPublishedName(String)**

```
public void setPublishedName(String name);
```

**Specified by:** Method `setPublishedName` in interface `ClientBroadcastStreamMBean`

Setter for stream published name

**Parameters**

name	Name that used for publishing. Set at client side when begin to broadcast with <code>NetStream#publish</code> .
------	---

## 5.22. **start()**

```
public void start();
```

**Specified by:** Method `start` in interface `ClientBroadcastStreamMBean`

Starts stream. Creates pipes, video codec from video codec factory bean, connects

## 5.23. **startPublishing()**

```
public void startPublishing();
```

**Specified by:** Method `startPublishing` in interface `IClientBroadcastStream`

Notify client that stream is ready for publishing.

## 5.24. **stop()**

```
public void stop();
```

**Specified by:** Method `stop` in interface `ClientBroadcastStreamMBean`

## 5.25. **stopRecording()**

```
public void stopRecording();
```

Stops any currently active recordings.

## 6. Interface ClientBroadcastStreamMBean

Represents live stream broadcasted from client. As Flash Media Server, Red5 supports recording mode for live streams, that is, broadcasted stream has broadcast mode. It can be either "live" or "record" and latter causes server-side application to record broadcasted stream. Note that recorded streams are recorded as FLV files. The same is correct for audio, because NellyMoser codec that Flash Player uses prohibits on-the-fly transcoding to audio formats like MP3 without paying of licensing fee or buying SDK. This type of stream uses two different pipes for live streaming and recording.

### 6.1. Synopsis

```
public interface ClientBroadcastStreamMBean {
// Public Methods

    public void close();

    public org.red5.server.messaging.IProvider getProvider();

    public String getPublishedName();

    public String getSaveFilename();

    public void saveAs(String name,
                      boolean isAppend)
        throws IOException, ResourceNotFoundException, ResourceExistException;

    public void setPublishedName(String name);

    public void start();

    public void startPublishing();

    public void stop();

}
```

## 7. Class ConsumerService

Basic consumer service implementation. Used to get pushed messages at consumer endpoint.

### 7.1. Synopsis

```
public class ConsumerService implements org.?red5.?server.?stream.?IConsumerService {
// Public Constructors

    public ConsumerService();

// Public Methods

    public org.red5.server.messaging.IMessageOutput getConsumerOutput(org.red5.server.api.stream.IClie
```

}

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 7.2. getConsumerOutput(IClientStream)

```
public org.red5.server.messaging.IMessageOutput getConsumerOutput(org.red5.server.api.stream.IClient
```

**Specified by:** Method getConsumerOutput in interface IConsumerService

Handles pushed messages

# 8. Class DefaultStreamFilenameGenerator

Default filename generator for streams. The files will be stored in a directory "streams" in the application folder. Option for changing directory streams are saved to is investigated as of 0.6RC1.

## 8.1. Synopsis

```
public class DefaultStreamFilenameGenerator implements org.?red5.?server.?api.?stream.?IStreamFilename
// Public Constructors

    public DefaultStreamFilenameGenerator();

// Public Methods

    public String generateFilename(org.red5.server.api.IScope scope,
                                  String name,
                                  String extension,
                                  org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType type);

    public String generateFilename(org.red5.server.api.IScope scope,
                                  String name,
                                  org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType type);

    public boolean resolvesToAbsolutePath();

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 8.2. generateFilename(IScope, String, IStreamFilenameGenerator.GenerationType)

```
public String generateFilename(org.red5.server.api.IScope scope,
                            String name,
                            org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType type);
```

**Specified by:** Method generateFilename in interface IStreamFilenameGenerator

Generate a filename without an extension.

### 8.3. generateFilename(IScope, String, String, IStreamFilenameGenerator.GenerationType)

```
public String generateFilename(org.red5.server.api.IScope scope,
                           String name,
                           String extension,
                           org.red5.server.api.stream.IStreamFilenameGenerator.GenerationType ty)
```

**Specified by:** Method generateFilename in interface IStreamFilenameGenerator

Generate a filename with an extension.

### 8.4. resolvesToAbsolutePath()

```
public boolean resolvesToAbsolutePath();
```

**Specified by:** Method resolvesToAbsolutePath in interface IStreamFilenameGenerator

The default filenames are relative to the scope path, so always return `false`.

#### Parameters

<i>return</i>	always <code>false</code>
---------------	---------------------------

## 9. Class DummyBWControlService

A dummy bandwidth control service (bandwidth controller) that always has token available.

### 9.1. Synopsis

```
public class DummyBWControlService implements org.?red5.?server.?stream.?IBWControlService {
    // Public Constructors

    public DummyBWControlService();

    // Public Methods

    public ITokenBucket getAudioBucket(IBWControlContext context);

    public ITokenBucket getDataBucket(IBWControlContext context);

    public ITokenBucket getVideoBucket(IBWControlContext context);

    public IBWControlContext lookupContext(org.red5.server.api.IBWControllable bc);

    public IBWControlContext registerBWControllable(org.red5.server.api.IBWControllable bc);

    public void resetBuckets(IBWControlContext context);

    public void unregisterBWControllable(IBWControlContext context);

    public void updateBWConfigure(IBWControlContext context);
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 9.2. getAudioBucket(IBWControlContext)

```
public ITokenBucket getAudioBucket(IBWControlContext context);
```

**Specified by:** Method getAudioBucket in interface IBWControlService

Return the token bucket for audio channel.

Parameters	
context	The registry context.
<i>return</i>	Token bucket for audio channel.

**Description copied from interface: getAudioBucket**

## 9.3. getDataBucket(IBWControlContext)

```
public ITokenBucket getDataBucket(IBWControlContext context);
```

**Specified by:** Method getDataBucket in interface IBWControlService

Return the token bucket for data channel.

Parameters	
context	The registry context.
<i>return</i>	Token bucket for data channel.

**Description copied from interface: getDataBucket**

## 9.4. getVideoBucket(IBWControlContext)

```
public ITokenBucket getVideoBucket(IBWControlContext context);
```

**Specified by:** Method getVideoBucket in interface IBWControlService

Return the token bucket for video channel.

Parameters	
context	The registry context.
<i>return</i>	Token bucket for video channel.

**Description copied from interface: getVideoBucket**

## 9.5. lookupContext(IBWControllable)

```
public IBWControlContext lookupContext(org.red5.server.api.IBWControllable bc);
```

**Specified by:** Method lookupContext in interface IBWControlService

Lookup the registry context according to the controllable.

Parameters	
bc	The bandwidth controllable.
<i>return</i>	The registry context.

**Description copied from interface: lookupContext**

## 9.6. registerBWControllable(IBWControllable)

```
public IBWControlContext registerBWControllable(org.red5.server.api.IBWControllable bc);
```

**Specified by:** Method registerBWControllable in interface IBWControlService

Register a bandwidth controllable. The necessary resources will be allocated and assigned to the controllable.

Parameters	
bc	The bandwidth controllable.
<i>return</i>	The registry context. It's used in the subsequent calls to controller's method.

**Description copied from interface: registerBWControllable**

## 9.7. resetBuckets(IBWControlContext)

```
public void resetBuckets(IBWControlContext context);
```

**Specified by:** Method resetBuckets in interface IBWControlService

Reset all the token buckets for a controllable. All the callback will be reset and all blocked threads will be woken up.

Parameters	
context	The registry context.

**Description copied from interface: resetBuckets**

## 9.8. unregisterBWControllable(IBWControlContext)

```
public void unregisterBWControllable(IBWControlContext context);
```

**Specified by:** Method unregisterBWControllable in interface IBWControlService

Unregister the bandwidth controllable. The resources that were allocated will be freed.

Parameters	

context	The registry context.
---------	-----------------------

**Description copied from interface: unregisterBWControllable**

## 9.9. updateBWConfigure(IBWControlContext)

```
public void updateBWConfigure(IBWControlContext context);
```

**Specified by:** Method updateBWConfigure in interface IBWControlService

Update the bandwidth configuration of a controllable. Each time when the controllable changes the bandwidth config and wants to make the changes take effect, this method should be called.

### Parameters

context	The registry context.
---------	-----------------------

**Description copied from interface: updateBWConfigure**

# 10. Class FileStreamSource

Represents stream source that is file

## 10.1. Synopsis

```
public class FileStreamSource implements, org.?red5.?server.?stream.?ISeekableStreamSource, org.?red5.

// Protected Fields

protected static org.slf4j.Logger log;

// Public Constructors

public FileStreamSource(org.red5.io.ITagReader reader);

// Public Methods

public void close();

public org.red5.server.net.rtmp.event.IRTMPEvent dequeue();

public boolean hasMore();

public synchronized int seek(int ts);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 10.2. FileStreamSource(ITagReader)

```
public FileStreamSource(org.red5.io.ITagReader reader);
```

Creates file stream source with tag reader

#### Parameters

reader	Tag reader
--------	------------

## 10.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 10.4. close()

```
public void close();
```

Closes tag reader

## 10.5. dequeue()

```
public org.red5.server.net.rtmp.event.IRTMPEvent dequeue();
```

Get tag from queue and convert to message

#### Parameters

return	RTMP event
--------	------------

## 10.6. hasMore()

```
public boolean hasMore();
```

## 10.7. seek(int)

```
public synchronized int seek(int ts);
```

**Specified by:** Method seek in interface ISearchableStreamSource

Seek the stream source to timestamp ts (in milliseconds).

# 11. Interface IBWControlContext

The registry context for a controllable.

## 11.1. Synopsis

```
public interface IBWControlContext {
    // Public Methods

    public org.red5.server.api.IBWControllable getIBWControllable();

}
```

## 11.2. getBWControllable()

```
public org.red5.server.api.IBWControllable getBWControllable();
```

Return the controllable that registered.

### Parameters

<i>return</i>
---------------

## 12. Interface IBWControlService

Bandwidth controller service interface.

The bandwidth controllable is registered in the bandwidth controller which provides the three token buckets used for bandwidth control.

The bandwidth controller manages the token buckets assigned to the bandwidth controllable and distributes the tokens to the buckets in an implementation-specific way.  
(eg timely distribute the tokens according to the bandwidth config of the controllable)

### 12.1. Synopsis

```
public interface IBWControlService {
    // Public Static Fields

    public static final String KEY = "BWControlService";

    // Public Methods

    public ITokenBucket getAudioBucket(IBWControlContext context);

    public ITokenBucket getDataBucket(IBWControlContext context);

    public ITokenBucket getVideoBucket(IBWControlContext context);

    public IBWControlContext lookupContext(org.red5.server.api.IBWControllable bc);

    public IBWControlContext registerBWControllable(org.red5.server.api.IBWControllable bc);

    public void resetBuckets(IBWControlContext context);

    public void unregisterBWControllable(IBWControlContext context);

    public void updateBWConfigure(IBWControlContext context);
}
```

### 12.2. getAudioBucket(IBWControlContext)

```
public ITokenBucket getAudioBucket(IBWControlContext context);
```

Return the token bucket for audio channel.

**Parameters**

context	The registry context.
---------	-----------------------

<i>return</i>	Token bucket for audio channel.
---------------	---------------------------------

**12.3. getDataBucket(IBWControlContext)**

```
public ITokenBucket getDataBucket(IBWControlContext context);
```

Return the token bucket for data channel.

**Parameters**

context	The registry context.
---------	-----------------------

<i>return</i>	Token bucket for data channel.
---------------	--------------------------------

**12.4. getVideoBucket(IBWControlContext)**

```
public ITokenBucket getVideoBucket(IBWControlContext context);
```

Return the token bucket for video channel.

**Parameters**

context	The registry context.
---------	-----------------------

<i>return</i>	Token bucket for video channel.
---------------	---------------------------------

**12.5. lookupContext(IBWControllable)**

```
public IBWControlContext lookupContext(org.red5.server.api.IBWControllable bc);
```

Lookup the registry context according to the controllable.

**Parameters**

bc	The bandwidth controllable.
----	-----------------------------

<i>return</i>	The registry context.
---------------	-----------------------

**12.6. registerBWControllable(IBWControllable)**

```
public IBWControlContext registerBWControllable(org.red5.server.api.IBWControllable bc);
```

Register a bandwidth controllable. The necessary resources will be allocated and assigned to the controllable.

**Parameters**

bc	The bandwidth controllable.
----	-----------------------------

<i>return</i>	The registry context. It's used in the subsequent calls to controller's method.
---------------	---

## 12.7. resetBuckets(IBWControlContext)

```
public void resetBuckets(IBWControlContext context);
```

Reset all the token buckets for a controllable. All the callback will be reset and all blocked threads will be woken up.

### Parameters

context	The registry context.
---------	-----------------------

## 12.8. unregisterBWControllable(IBWControlContext)

```
public void unregisterBWControllable(IBWControlContext context);
```

Unregister the bandwidth controllable. The resources that were allocated will be freed.

### Parameters

context	The registry context.
---------	-----------------------

## 12.9. updateBWConfigure(IBWControlContext)

```
public void updateBWConfigure(IBWControlContext context);
```

Update the bandwidth configuration of a controllable. Each time when the controllable changes the bandwidth config and wants to make the changes take effect, this method should be called.

### Parameters

context	The registry context.
---------	-----------------------

## 13. Interface IBroadcastScope

Broadcast scope is marker interface that represents object that works as basic scope and has pipe connection event dispatching capabilities.

### 13.1. Synopsis

```
public interface IBroadcastScope extends, org.?red5.?server.?api.?IBasicScope, org.?red5.?server.?mes
// Public Static Fields

    public static final String STREAM_ATTRIBUTE = "_transient_publishing_stream";

    public static final String TYPE = "bs";

}
```

## 14. Interface IConsumerService

Service for consumer objects, used to get pushed messages at consumer endpoint.

## 14.1. Synopsis

```
public interface IConsumerService {
// Public Static Fields

    public static final String KEY = "consumerService";

// Public Methods

    public org.red5.server.messaging.IMessageOutput getConsumerOutput(org.red5.server.api.stream.IClientStream);
}
```

## 14.2. getConsumerOutput(IClientStream)

```
public org.red5.server.messaging.IMessageOutput getConsumerOutput(org.red5.server.api.stream.IClientStream);
```

Handles pushed messages

### Parameters

stream	Client stream object
return	Message object

## 15. Interface IFrameDropper

Interface for classes that implement logic to drop frames.

## 15.1. Synopsis

```
public interface IFrameDropper {
// Public Static Fields

    public static final int SEND_ALL = 0;

    public static final int SEND_INTERFRAMES = 1;

    public static final int SEND_KEYFRAMES = 2;

    public static final int SEND_KEYFRAMES_CHECK = 3;

// Public Methods

    public boolean canSendPacket(org.red5.server.stream.message.RTMPMessage message,
                                long pending);

    public void dropPacket(org.red5.server.stream.message.RTMPMessage message);

    public void reset();

    public void reset(int state);
```

```
public void sendPacket(org.red5.server.stream.message.RTMPMessage message);

}
```

## 15.2. SEND\_ALL

```
public static final int SEND_ALL = 0;
```

Send keyframes, interframes and disposable interframes.

## 15.3. SEND\_INTERFRAMES

```
public static final int SEND_INTERFRAMES = 1;
```

Send keyframes and interframes.

## 15.4. SEND\_KEYFRAMES

```
public static final int SEND_KEYFRAMES = 2;
```

Send keyframes only.

## 15.5. SEND\_KEYFRAMES\_CHECK

```
public static final int SEND_KEYFRAMES_CHECK = 3;
```

Send keyframes only and switch to SEND\_INTERFRAMES later.

## 15.6. canSendPacket(RTMPMessage, long)

```
public boolean canSendPacket(org.red5.server.stream.message.RTMPMessage message,
                           long pending);
```

Checks if a message may be sent to the subscriber.

Parameters	
message	the message to check
pending	the number of pending messages
return	true if the packet may be sent, otherwise false

## 15.7. dropPacket(RTMPMessage)

```
public void dropPacket(org.red5.server.stream.message.RTMPMessage message);
```

Notify that a packet has been dropped.

Parameters	
message	the message that was dropped

## 15.8. reset()

```
public void reset();
```

Reset the frame dropper.

## 15.9. reset(int)

```
public void reset(int state);
```

Reset the frame dropper to a given state.

### Parameters

state	the state to reset the frame dropper to
-------	---

## 15.10. sendPacket(RTMPMessage)

```
public void sendPacket(org.red5.server.stream.message.RTMPMessage message);
```

Notify that a message has been sent.

### Parameters

message	the message that was sent
---------	---------------------------

# 16. Interface IProviderService

Central unit to get access to different types of provider inputs

## 16.1. Synopsis

```
public interface IProviderService extends, org.?red5.?server.?api.?IScopeService {
// Public Static Fields

    public static final String BEAN_NAME = "providerService";

// Public Methods

    public java.util.List<java.lang.String> getBroadcastStreamNames(org.red5.server.api.IScope scope);

    public org.red5.server.messaging.IMessageInput getLiveProviderInput(org.red5.server.api.IScope scope,
                                                                       String name,
                                                                       boolean needCreate);

    public org.red5.server.messaging.IMessageInput getProviderInput(org.red5.server.api.IScope scope,
                                                                   String name);

    public java.io.File getVODProviderFile(org.red5.server.api.IScope scope,
                                            String name);

    public org.red5.server.messaging.IMessageInput getVODProviderInput(org.red5.server.api.IScope scope,
                                                                       String name);

    public org.red5.server.stream.IProviderService.INPUT_TYPE lookupProviderInput(org.red5.server.api.IScope scope,
                                                                               String name);
```

```

public boolean registerBroadcastStream(org.red5.server.api.IScope scope,
                                      String name,
                                      org.red5.server.api.stream.IBroadcastStream bs);

public boolean unregisterBroadcastStream(org.red5.server.api.IScope scope,
                                         String name);

}

```

## 16.2. getBroadcastStreamNames(IScope)

```
public java.util.List<java.lang.String> getBroadcastStreamNames(org.red5.server.api.IScope scope);
```

Get names of existing broadcast streams in a scope.

### Parameters

scope	Scope to get stream names from
<i>return</i>	List of stream names

## 16.3. getLiveProviderInput(IScope, String, boolean)

```
public org.red5.server.messaging.IMessageInput getLiveProviderInput(org.red5.server.api.IScope scope,
                                                                    String name,
                                                                    boolean needCreate);
```

Get a named Live provider as the source of input.

### Parameters

scope	Scope of provider
name	Name of provider
needCreate	Whether there's need to create basic scope / live provider if they don't exist
<i>return</i>	null if not found.

## 16.4. getProviderInput(IScope, String)

```
public org.red5.server.messaging.IMessageInput getProviderInput(org.red5.server.api.IScope scope,
                                                               String name);
```

Get a named provider as the source of input. Live stream first, VOD stream second.

### Parameters

scope	Scope of provider
name	Name of provider
<i>return</i>	null if nothing found.

## 16.5. getVODProviderFile(IScope, String)

```
public java.io.File getVODProviderFile(org.red5.server.api.IScope scope,
                                         String name);
```

Get a named VOD source file.

### Parameters

scope	Scope of provider
name	Name of provider
<i>return</i>	<code>null</code> if not found.

## 16.6. getVODProviderInput(IScope, String)

```
public org.red5.server.messaging.IMessageInput getVODProviderInput(org.red5.server.api.IScope scope,
                                                                     String name);
```

Get a named VOD provider as the source of input.

### Parameters

scope	Scope of provider
name	Name of provider
<i>return</i>	<code>null</code> if not found.

## 16.7. lookupProviderInput(IScope, String)

```
public org.red5.server.stream.IProviderService.INPUT_TYPE lookupProviderInput(org.red5.server.api.IScope scope,
                                                                           String name);
```

Returns the input type for a named provider if a source of input exists. Live is checked first and VOD second.

### Parameters

scope	Scope of provider
name	Name of provider
<i>return</i>	LIVE if live, VOD if VOD, and NOT_FOUND otherwise

## 16.8. registerBroadcastStream(IScope, String, IBroadcastStream)

```
public boolean registerBroadcastStream(org.red5.server.api.IScope scope,
                                         String name,
                                         org.red5.server.api.stream.IBroadcastStream bs);
```

Register a broadcast stream to a scope.

### Parameters

scope	Scope
name	Name of stream
bs	Broadcast stream to register
return	true if register successfully.

## 16.9. unregisterBroadcastStream(IScope, String)

```
public boolean unregisterBroadcastStream(org.red5.server.api.IScope scope,
                                         String name);
```

Unregister a broadcast stream of a specific name from a scope.

### Parameters

scope	Scope
name	Stream name
return	true if unregister successfully.

## 17. Class IProviderService.INPUT\_TYPE

```
public static final class IProviderService.INPUT_TYPE extends java.lang.Enum {
// Public Static Fields

    public static final org.red5.server.stream.IProviderService.INPUT_TYPE LIVE ;

    public static final org.red5.server.stream.IProviderService.INPUT_TYPE NOT_FOUND ;

    public static final org.red5.server.stream.IProviderService.INPUT_TYPE VOD ;

// Public Static Methods

    public static org.red5.server.stream.IProviderService.INPUT_TYPE valueOf(String name);

    public static org.red5.server.stream.IProviderService.INPUT_TYPE[] values();

}
```

**Methods inherited from java.lang.Enum:** clone, compareTo, equals, finalize, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

**Methods inherited from java.lang.Object:** getClass, notify, notifyAll, wait

## 18. Interface ISearchableProvider

Provider that is seekable

### 18.1. Synopsis

```
public interface ISearchableProvider extends org.red5.server.messaging.IProvider {
```

```
// Public Static Fields

    public static final String KEY ;

// Public Methods

    public int seek(int ts);

}
```

## 18.2. seek(int)

```
public int seek(int ts);
```

Seek the provider to timestamp ts (in milliseconds).

Parameters	
ts	Timestamp to seek to
<i>return</i>	Actual timestamp seeked to

# 19. Interface ISeekableStreamSource

Stream source that can be seeked in timeline

## 19.1. Synopsis

```
public interface ISeekableStreamSource extends, org.?red5.?server.?stream.?IStreamSource {
// Public Methods

    public int seek(int ts);

}
```

## 19.2. seek(int)

```
public int seek(int ts);
```

Seek the stream source to timestamp ts (in milliseconds).

Parameters	
ts	Timestamp to seek to
<i>return</i>	Actual timestamp seeked to

# 20. Interface IStreamControl

Stream Control Event. Marker interface for stream control operations. Play, Pause, Resume, Seek, Stop, etc.

## 20.1. Synopsis

```
public interface IStreamControl {
}
```

# 21. Interface IStreamData

Stream data packet

## 21.1. Synopsis

```
public interface IStreamData {
// Public Methods

    public org.apache.mina.common.ByteBuffer getData();

}
```

## 21.2. getData()

```
public org.apache.mina.common.ByteBuffer getData();
```

Getter for property 'data'.

Parameters

<i>return</i>	Value for property 'data'.
---------------	----------------------------

# 22. Interface IStreamSource

Source for streams

## 22.1. Synopsis

```
public interface IStreamSource {
// Public Methods

    public void close();

    public org.red5.server.api.event.IEvent dequeue();

    public boolean hasMore();

}
```

## 22.2. close()

```
public void close();
```

Close stream source

## 22.3. dequeue()

```
public org.red5.server.api.event.IEvent dequeue();
```

Double ended queue of event objects

### Parameters

<i>return</i>	Event from queue
---------------	------------------

## 22.4. hasMore()

```
public boolean hasMore();
```

Is there something more to stream?

### Parameters

<i>return</i>	true if there's streamable data, false otherwise
---------------	--

## 23. Interface IStreamTypeAwareProvider

Interface for providers that know if they contain video frames.

### 23.1. Synopsis

```
public interface IStreamTypeAwareProvider extends, org.?red5.?server.?messaging.?IProvider {
    // Public Static Fields

    public static final String KEY ;

    // Public Methods

    public boolean hasVideo();
}
```

## 23.2. hasVideo()

```
public boolean hasVideo();
```

Check if the provider contains video tags.

### Parameters

<i>return</i>	
---------------	--

## 24. Interface IToknBucket

Basically token bucket is used to control the bandwidth used by a stream or a connection or a client. There's a background thread that distributes tokens to the buckets in the system according to the configuration of the bucket. The configuration includes how fast the tokens are distributed. When a stream, for example, needs to send out a packet, the packet's

byte count is calculated and each byte corresponds to a token in the bucket. The stream is assigned a bucket and the tokens in the bucket are acquired before the packet can be sent out. So if the speed(or bandwidth) in configuration is low, the stream can't send out packets fast.

## 24.1. Synopsis

```
public interface ITokenBucket {
    // Public Methods

    public boolean acquireToken(long tokenCount,
                               long wait);

    public long acquireTokenBestEffort(long upperLimitCount);

    public boolean acquireTokenNonblocking(long tokenCount,
                                         org.red5.server.stream.ITokenBucket.ITokenBucketCallback callback);

    public long getCapacity();

    public double getSpeed();

    public void reset();

}
```

## 24.2. acquireToken(long, long)

```
public boolean acquireToken(long tokenCount,
                           long wait);
```

Acquire tokens amount of `tokenCount` waiting `wait` milliseconds if token not available.

Parameters	
<code>tokenCount</code>	The count of tokens to acquire.
<code>wait</code>	Milliseconds to wait. 0 means no wait and any value below zero means wait forever.
<code>return</code>	<code>true</code> if successfully acquired or <code>false</code> if not acquired.

## 24.3. acquireTokenBestEffort(long)

```
public long acquireTokenBestEffort(long upperLimitCount);
```

Nonblockingly acquire token. The upper limit is specified. If not enough tokens are left in bucket, all remaining will be returned.

Parameters	
<code>upperLimitCount</code>	Upper limit of aquisition
<code>return</code>	Remaining tokens from bucket

## 24.4. acquireTokenNonblocking(long, ITokenBucket.ITokenBucketCallback)

```
public boolean acquireTokenNonblocking(long tokenCount,
                                     org.red5.server.stream.ITokenBucket.ITokenBucketCallback call
```

Nonblockingly acquire token. If the token is not available and `task` is not null, the callback will be executed when the token is available. The tokens are not consumed automatically before callback, so it's recommended to acquire token again in callback function.

### Parameters

<code>tokenCount</code>	Number of tokens
<code>callback</code>	Callback
<code>return</code>	<code>true</code> if successfully acquired or <code>false</code> if not acquired.

## 24.5. getCapacity()

```
public long getCapacity();
```

Get the capacity of this bucket in Byte.

### Parameters

<code>return</code>	Capacity of this bucket in bytes
---------------------	----------------------------------

## 24.6. getSpeed()

```
public double getSpeed();
```

The amount of tokens increased per millisecond.

### Parameters

<code>return</code>	Amount of tokens increased per millisecond.
---------------------	---

## 24.7. reset()

```
public void reset();
```

Reset this token bucket. All pending threads are woken up with `false` returned for acquiring token and callback is removed without calling back.

## 25. Interface ITokenBucket.ITokenBucketCallback

Callback for socket bucket

### 25.1. Synopsis

```
public static interface ITokenBucket.ITokenBucketCallback {
    // Public Methods
```

```

    public void available(ITokenBucket bucket,
                          long tokenCount);

    public void reset(ITokenBucket bucket,
                      long tokenCount);

}

```

## 25.2. available(ITokenBucket, long)

```
public void available(ITokenBucket bucket,
                      long tokenCount);
```

Being called when the tokens requested are available.

### Parameters

bucket	Bucket
tokenCount	Number of tokens

## 25.3. reset(ITokenBucket, long)

```
public void reset(ITokenBucket bucket,
                  long tokenCount);
```

Resets tokens in bucket

### Parameters

bucket	Bucket
tokenCount	Number of tokens

# 26. Interface ITOKENBUCKETSERVICE

A service used to create and manage token buckets.

## 26.1. Synopsis

```

public interface ITOKENBUCKETSERVICE {
// Public Static Fields

    public static final String KEY = "TOKENBUCKETSERVICE";

// Public Methods

    public ITOKENBUCKET createTOKENBUCKET(long capacity,
                                           long speed);

    public void removeTOKENBUCKET(ITOKENBUCKET bucket);

}

```

## 26.2. createTokenBucket(long, long)

```
public ITokenBucket createTokenBucket(long capacity,
                                     long speed);
```

Create a token bucket.

### Parameters

capacity	Capacity of the bucket.
speed	Speed of the bucket. Bytes per millisecond.
return	null if fail to create.

## 26.3. removeTokenBucket(ITokenBucket)

```
public void removeTokenBucket(ITokenBucket bucket);
```

Remove this bucket.

### Parameters

bucket	Bucket to remove
--------	------------------

# 27. Class OutputStream

Output stream that consists of audio, video and data channels

## 27.1. Synopsis

```
public class OutputStream {
// Protected Fields

    protected static org.slf4j.Logger log ;

// Public Constructors

    public OutputStream(org.red5.server.net.rtmp.Channel video,
                      org.red5.server.net.rtmp.Channel audio,
                      org.red5.server.net.rtmp.Channel data);

// Public Methods

    public void close();

    public org.red5.server.net.rtmp.Channel getAudio();

    public org.red5.server.net.rtmp.Channel getData();

    public org.red5.server.net.rtmp.Channel getVideo();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

**See Also**

[org.red5.server.net.rtmp.Channel](#)

## 27.2. OutputStream(Channel, Channel, Channel)

```
public OutputStream(org.red5.server.net.rtmp.Channel video,
                  org.red5.server.net.rtmp.Channel audio,
                  org.red5.server.net.rtmp.Channel data);
```

Creates output stream from channels

**Parameters**

video	Video channel
audio	Audio channel
data	Data channel

## 27.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 27.4. close()

```
public void close();
```

Closes audion, video and data channels

## 27.5. getAudio()

```
public org.red5.server.net.rtmp.Channel getAudio();
```

Getter for audio channel

**Parameters**

return	Audio channel
--------	---------------

## 27.6. getData()

```
public org.red5.server.net.rtmp.Channel getData();
```

Getter for data channel

**Parameters**

return	Data channel
--------	--------------

## 27.7. getVideo()

```
public org.red5.server.net.rtmp.Channel getVideo();
```

Getter for video channel

#### Parameters

<i>return</i>	Video channel
---------------	---------------

## 28. Class PlayBuffer

A Play buffer for sending VOD. The implementation is not synchronized.

### 28.1. Synopsis

```
public class PlayBuffer {
// Public Constructors

    public PlayBuffer(long capacity);

// Public Methods

    public void clear();

    public long getCapacity();

    public int getMessageCount();

    public long getMessageSize();

    public org.red5.server.stream.message.RTMPMessage peekMessage();

    public boolean putMessage(org.red5.server.stream.message.RTMPMessage message);

    public void setCapacity(long capacity);

    public org.red5.server.stream.message.RTMPMessage takeMessage();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 28.2. PlayBuffer(long)

```
public PlayBuffer(long capacity);
```

Create play buffer with given capacity

#### Parameters

capacity	Capacity of buffer
----------	--------------------

### 28.3. clear()

```
public void clear();
```

Empty this buffer.

## 28.4. getCapacity()

```
public long getCapacity();
```

Buffer capacity in bytes.

### Parameters

<i>return</i>	Buffer capacity in bytes
---------------	--------------------------

## 28.5. getMessageCount()

```
public int getMessageCount();
```

Number of messages in buffer.

### Parameters

<i>return</i>	Number of messages in buffer
---------------	------------------------------

## 28.6. getMessageSize()

```
public long getMessageSize();
```

Total message size in bytes.

### Parameters

<i>return</i>	Total message size in bytes
---------------	-----------------------------

## 28.7. peekMessage()

```
public org.red5.server.stream.message.RTMPMessage peekMessage();
```

Peek a message but not take it from the buffer. The message count doesn't change.

### Parameters

<i>return</i>	null if buffer is empty.
---------------	--------------------------

## 28.8. putMessage(RTMPMessage)

```
public boolean putMessage(org.red5.server.stream.message.RTMPMessage message);
```

Put a message into this buffer.

### Parameters

message	RTMP message
---------	--------------

<i>return</i>	true indicates success and false indicates buffer is full.
---------------	--

## 28.9. setCapacity(long)

```
public void setCapacity(long capacity);
```

Setter for capacity

### Parameters

capacity	New capacity
----------	--------------

## 28.10. takeMessage()

```
public org.red5.server.stream.message.RTMPMessage takeMessage();
```

Take a message from this buffer. The message count decreases.

### Parameters

return	null if buffer is empty.
--------	--------------------------

# 29. Class PlayEngine

A play engine for playing an IPlayItem.

## 29.1. Synopsis

```
public final class PlayEngine implements org.?red5.?server.?messaging.?IFilter, org.?red5.?server.?me
// Public Methods

    public synchronized void available(ITokenBucket bucket,
                                      long tokenCount);

    public synchronized void close();

    public org.red5.server.net.rtmp.event.IRTMPEvent getLastMessage();

    public long getPlaybackStart();

    public boolean isPaused();

    public boolean isPullMode();

    public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                    org.red5.server.messaging.IPipe pipe,
                                    org.red5.server.messaging.OOBControlMessage oobCtrlMsg);

    public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);

    public synchronized void pause(int position)
        throws IllegalStateException;

    public void play(org.red5.server.api.stream.IPlayItem item)
        throws StreamNotFoundException, IllegalStateException, IOException;

    public synchronized void play(org.red5.server.api.stream.IPlayItem item,
```

```

        boolean withReset)
throws StreamNotFoundException, IllegalStateException, IOException;

public synchronized void pushMessage(org.red5.server.messaging.IPipe pipe,
                                     org.red5.server.messaging.IMessage message)
throws IOException;

public boolean receiveAudio();

public boolean receiveAudio(boolean receive);

public boolean receiveVideo();

public boolean receiveVideo(boolean receive);

public void reset(ITokenBucket bucket,
                  long tokenCount);

public synchronized void resume(int position)
throws IllegalStateException;

public synchronized void seek(int position)
throws IllegalStateException, OperationNotSupportedException;

public void sendBlankAudio(boolean sendBlankAudio);

public void setBandwidthController(IBWControlService bwController,
                                   IBWControlContext bwContext);

public void setBufferCheckInterval(int bufferCheckInterval);

public void setUnderrunTrigger(int underrunTrigger);

public synchronized void start();

public synchronized void stop()
throws IllegalStateException;

public void updateBandwithConfigure();

// Protected Methods

protected synchronized void pullAndPush()
throws IOException;
}

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 29.2. available(ITokenBucket, long)

```

public synchronized void available(ITokenBucket bucket,
                                 long tokenCount);
}

```

**Specified by:** Method available in interface ITokenBucket.ITokenBucketCallback

Being called when the tokens requested are available.

### 29.3. close()

```
public synchronized void close();
```

Close stream

### 29.4. isPullMode()

```
public boolean isPullMode();
```

Get informations about bytes send and number of bytes the client reports to have received.

#### Parameters

<i>return</i>	Written bytes and number of bytes the client received
---------------	---

### 29.5. onOOBControlMessage(IMessageComponent, IPipe, OOBControlMessage)

```
public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                org.red5.server.messaging.IPipe pipe,
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

### 29.6. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler

### 29.7. pause(int)

```
public synchronized void pause(int position)
                               throws IllegalStateException;
```

Pause at position

#### Parameters

<i>position</i>	Position in file
-----------------	------------------

`IllegalStateException`

If stream is stopped

### 29.8. play(IPlayItem)

```
public void play(org.red5.server.api.stream.IPlayItem item)
                throws StreamNotFoundException, IllegalStateException, IOException;
```

Play stream

**Parameters**

item	Playlist item
------	---------------

StreamNotFoundException

Stream not found

IllegalStateException

Stream is in stopped state

IOException

java.io.IOException

**29.9. play(IPlayItem, boolean)**

```
public synchronized void play(org.red5.server.api.stream.IPlayItem item,
                             boolean withReset)
throws StreamNotFoundException, IllegalStateException, IOException;
```

Play stream

**Parameters**

item	Playlist item
withReset	Send reset status before playing.

StreamNotFoundException

Stream not found

IllegalStateException

Stream is in stopped state

IOException

java.io.IOException

**29.10. pullAndPush()**

```
protected synchronized void pullAndPush()
throws IOException;
```

Recieve then send if message is data (not audio or video)

**29.11. pushMessage(IPipe, IMessage)**

```
public synchronized void pushMessage(org.red5.server.messaging.IPipe pipe,
                                     org.red5.server.messaging.IMessage message)
throws IOException;
```

**Specified by:** Method pushMessage in interface IPushableConsumer

Pushes message through pipe

**29.12. receiveAudio()**

```
public boolean receiveAudio();
```

Returns true if the engine currently receives audio.

#### Parameters

<i>return</i>
---------------

### 29.13. receiveAudio(boolean)

public boolean receiveAudio(boolean receive);
---

Returns true if the engine currently receives audio and sets the new value.

#### Parameters

<i>return</i>
---------------

### 29.14. receiveVideo()

public boolean receiveVideo();
--------------------------------

Returns true if the engine currently receives video.

#### Parameters

<i>return</i>
---------------

### 29.15. receiveVideo(boolean)

public boolean receiveVideo(boolean receive);
---

Returns true if the engine currently receives video and sets the new value.

#### Parameters

<i>return</i>
---------------

### 29.16. reset(ITokenBucket, long)

public void reset(ITokenBucket bucket, long tokenCount);
---

**Specified by:** Method `reset` in interface `ITokenBucket.ITokenBucketCallback`

Resets tokens in bucket

### 29.17. resume(int)

public synchronized void resume(int position) throws IllegalStateException;
--

Resume playback

#### Parameters

position	Resumes playback
----------	------------------

IllegalStateException  
If stream is stopped

## 29.18. seek(int)

```
public synchronized void seek(int position)
    throws IllegalStateException, OperationNotSupportedException;
```

Seek position in file

Parameters	
position	Position

IllegalStateException  
If stream is in stopped state

## 29.19. start()

```
public synchronized void start();
```

Start stream

## 29.20. stop()

```
public synchronized void stop()
    throws IllegalStateException;
```

Stop playback

IllegalStateException  
If stream is in stopped state

## 29.21. updateBandwidthConfigure()

```
public void updateBandwidthConfigure();
```

Update bandwidth configuration

# 30. Class PlayEngine.Builder

Builder pattern

## 30.1. Synopsis

```
public static class PlayEngine.Builder {
    // Public Constructors

    public PlayEngine.Builder(PlaylistSubscriberStream playlistSubscriberStream,
        org.red5.server.api.scheduling.ISchedulingService schedulingService,
        IConsumerService consumerService,
        IProviderService providerService);

    // Public Methods
```

```
public PlayEngine build();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 31. Class PlaylistSubscriberStream

Stream of playlist subscriber

### 31.1. Synopsis

```
public class PlaylistSubscriberStream extends, org.?red5.?server.?stream.?AbstractClientStream
    implements, org.?red5.?server.?api.?stream.?IPlaylistSubscriberStream, org.?red5.?server.?api.?sta
// Public Constructors

    public PlaylistSubscriberStream();

// Public Methods

    public void addItem(org.red5.server.api.stream.IPlayItem item);

    public void addItem(org.red5.server.api.stream.IPlayItem item,
                      int index);

    public void close();

    public long getBytesSent();

    public long getCreationTime();

    public org.red5.server.api.stream.IPlayItem getCurrentItem();

    public int getCurrentItemIndex();

    public int getCurrentTimestamp();

    public double getEstimatedBufferFill();

    public java.util.concurrent.ScheduledThreadPoolExecutor getExecutor();

    public org.red5.server.api.stream.IPlayItem getItem(int index);

    public int getItemCount();

    public org.red5.server.api.statistics.IPlaylistSubscriberStatistics getStatistics();

    public boolean hasMoreItems();

    public boolean isPaused();

    public boolean isRandom();
```

## Package org.?red5.?server.?stream

```
public boolean isRepeat();

public boolean isRewind();

public void nextItem();

public void pause(int position);

public void play()
    throws IOException;

public void previousItem();

public void receiveAudio(boolean receive);

public void receiveVideo(boolean receive);

public void removeAllItems();

public void removeItem(int index);

public void resume(int position);

public void seek(int position)
    throws OperationNotSupportedException;

public void setBandwidthConfigure(org.red5.server.api.IBandwidthConfigure config);

public void setBandwidthController(IBWControlService bwController);

public void setBufferCheckInterval(int bufferCheckInterval);

public void setExecutor(java.util.concurrent.ScheduledThreadPoolExecutor executor);

public void setItem(int index);

public void setPlaylistController(org.red5.server.api.stream.IPlaylistController controller);

public void setRandom(boolean random);

public void setRepeat(boolean repeat);

public void setRewind(boolean rewind);

public void setUnderrunTrigger(int underrunTrigger);

public void start();

public void stop();

public void written(Object message);

// Protected Methods

protected void notifyItemPause(org.red5.server.api.stream.IPlayItem item,
                               int position);
```

```

protected void notifyItemPlay(org.red5.server.api.stream.IPlayItem item,
                            boolean isLive);

protected void notifyItemResume(org.red5.server.api.stream.IPlayItem item,
                               int position);

protected void notifyItemSeek(org.red5.server.api.stream.IPlayItem item,
                            int position);

protected void notifyItemStop(org.red5.server.api.stream.IPlayItem item);

protected void notifySubscriberClose();

protected void notifySubscriberStart();

protected void onItemEnd();

}

```

**Methods inherited from org.red5.server.stream.AbstractClientStream:**

getBandwidthConfigure , getClientBufferDuration , getConnection  
, getParentBWControllable , getStreamId , setBandwidthConfigure ,  
setClientBufferDuration , setConnection , setStreamId

**Methods inherited from org.red5.server.stream.AbstractStream:** getCodecInfo ,  
getName , getScope , getStreamAwareHandler , setCodecInfo , setName , setScope

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.stream.AbstractStream:** state

## 31.2. PlaylistSubscriberStream()

```
public PlaylistSubscriberStream();
```

Constructs a new PlaylistSubscriberStream.

## 31.3. addItem(IPlayItem)

```
public void addItem(org.red5.server.api.stream.IPlayItem item);
```

## 31.4. addItem(IPlayItem, int)

```
public void addItem(org.red5.server.api.stream.IPlayItem item,
                   int index);
```

## 31.5. close()

```
public void close();
```

## 31.6. getBytesSent()

```
public long getBytesSent();
```

**Specified by:** Method `getBytesSent` in interface `IPlaylistSubscriberStreamStatistics`

Return total number of bytes sent to the client from this stream.

### 31.7. `getCreationTime()`

```
public long getCreationTime();
```

### 31.8. `getCurrentItem()`

```
public org.red5.server.api.stream.IPlayItem getCurrentItem();
```

### 31.9. `getCurrentItemIndex()`

```
public int getCurrentItemIndex();
```

### 31.10. `getCurrentTimestamp()`

```
public int getCurrentTimestamp();
```

### 31.11. `getEstimatedBufferFill()`

```
public double getEstimatedBufferFill();
```

**Specified by:** Method `getEstimatedBufferFill` in interface `IPlaylistSubscriberStreamStatistics`

Return estimated fill ratio of the client buffer.

### 31.12. `getExecutor()`

```
public java.util.concurrent.ScheduledThreadPoolExecutor getExecutor();
```

Return the executor to use.

#### Parameters

<i>return</i>	the executor
---------------	--------------

### 31.13. `getItem(int)`

```
public org.red5.server.api.stream.IPlayItem getItem(int index);
```

### 31.14. `getItemSize()`

```
public int getItemSize();
```

### 31.15. `getStatistics()`

```
public org.red5.server.api.statistics.IPlaylistSubscriberStreamStatistics getStatistics();
```

**Specified by:** Method `getStatistics` in interface `IPlaylistSubscriberStream`

Return statistics about this stream.

**31.16. hasMoreItems()**

```
public boolean hasMoreItems();
```

**31.17. isPaused()**

```
public boolean isPaused();
```

**31.18. isRandom()**

```
public boolean isRandom();
```

**31.19. isRepeat()**

```
public boolean isRepeat();
```

**31.20. isRewind()**

```
public boolean isRewind();
```

**31.21. nextItem()**

```
public void nextItem();
```

**31.22. notifyItemPause(IPlayItem, int)**

```
protected void notifyItemPause(org.red5.server.api.stream.IPlayItem item,
                           int position);
```

Notifies subscribers on pause

**Parameters**

item	Item that just has been paused
position	Playback head position

**31.23. notifyItemPlay(IPlayItem, boolean)**

```
protected void notifyItemPlay(org.red5.server.api.stream.IPlayItem item,
                           boolean isLive);
```

Notifies subscribers on item playback

**Parameters**

item	Item being played
isLive	Is it a live broadcasting?

**31.24. notifyItemResume(IPlayItem, int)**

```
protected void notifyItemResume(org.red5.server.api.stream.IPlayItem item,
                               int position);
```

Notifies subscribers on resume

#### Parameters

item	Item that just has been resumed
position	Playback head position

### 31.25. notifyItemSeek(IPlayItem, int)

```
protected void notifyItemSeek(org.red5.server.api.stream.IPlayItem item,
                           int position);
```

Notify on item seek

#### Parameters

item	Playlist item
position	Seek position

### 31.26. notifyItemStop(IPlayItem)

```
protected void notifyItemStop(org.red5.server.api.stream.IPlayItem item);
```

Notifies subscribers on item stop

#### Parameters

item	Item that just has been stopped
------	---------------------------------

### 31.27. notifySubscriberClose()

```
protected void notifySubscriberClose();
```

Notifies subscribers on stop

### 31.28. notifySubscriberStart()

```
protected void notifySubscriberStart();
```

Notifies subscribers on start

### 31.29. onItemEnd()

```
protected void onItemEnd();
```

Notified by the play engine when the current item reaches the end.

### 31.30. pause(int)

```
public void pause(int position);
```

### 31.31. play()

```
public void play();
```

```
throws IOException;
```

**31.32. previousItem()**

```
public void previousItem();
```

**31.33. receiveAudio(boolean)**

```
public void receiveAudio(boolean receive);
```

**31.34. receiveVideo(boolean)**

```
public void receiveVideo(boolean receive);
```

**31.35. removeAllItems()**

```
public void removeAllItems();
```

**31.36. removeItem(int)**

```
public void removeItem(int index);
```

**31.37. resume(int)**

```
public void resume(int position);
```

**31.38. seek(int)**

```
public void seek(int position)
    throws OperationNotSupportedException;
```

**31.39. setBandwidthConfigure(IBandwidthConfigure)**

```
public void setBandwidthConfigure(org.red5.server.api.IBandwidthConfigure config);
```

**31.40. setBufferCheckInterval(int)**

```
public void setBufferCheckInterval(int bufferCheckInterval);
```

Set interval to check for buffer underruns. Set to 0 to disable.

**Parameters**

bufferCheckInterval	interval in ms
---------------------	----------------

**31.41. setExecutor(ScheduledThreadPoolExecutor)**

```
public void setExecutor(java.util.concurrent.ScheduledThreadPoolExecutor executor);
```

Set the executor to use.

**Parameters**

executor	the executor
----------	--------------

**31.42. setItem(int)**

```
public void setItem(int index);
```

**31.43. setPlaylistController(IPlaylistController)**

```
public void setPlaylistController(org.red5.server.api.stream.IPlaylistController controller);
```

**31.44. setRandom(boolean)**

```
public void setRandom(boolean random);
```

**31.45. setRepeat(boolean)**

```
public void setRepeat(boolean repeat);
```

**31.46. setRewind(boolean)**

```
public void setRewind(boolean rewind);
```

**31.47. setUnderrunTrigger(int)**

```
public void setUnderrunTrigger(int underrunTrigger);
```

Set maximum number of pending messages at which a `NetStream.Play.InsufficientBW` message will be generated for VOD streams

## Parameters

underrunTrigger	the maximum number of pending messages
-----------------	--

**31.48. start()**

```
public void start();
```

**31.49. stop()**

```
public void stop();
```

**31.50. written(Object)**

```
public void written(Object message);
```

Notified by RTMPHandler when a message has been sent. Glue for old code base.

## Parameters

message	Message that has been written
---------	-------------------------------

**32. Class ProviderService**

```
public class ProviderService implements org.?red5.?server.?stream.?IProviderService {
// Public Constructors
```

```

public ProviderService();

// Public Methods

public java.util.List<java.lang.String> getBroadcastStreamNames(org.red5.server.api.IScope scope);

public org.red5.server.messaging.IMessageInput getLiveProviderInput(org.red5.server.api.IScope scope,
    String name,
    boolean needCreate);

public org.red5.server.messaging.IMessageInput getProviderInput(org.red5.server.api.IScope scope,
    String name);

public java.io.File getVODProviderFile(org.red5.server.api.IScope scope,
    String name);

public org.red5.server.messaging.IMessageInput getVODProviderInput(org.red5.server.api.IScope scope,
    String name);

public org.red5.server.stream.IProviderService.INPUT_TYPE lookupProviderInput(org.red5.server.api.IScope scope,
    String name);

public boolean registerBroadcastStream(org.red5.server.api.IScope scope,
    String name,
    org.red5.server.api.stream.IBroadcastStream bs);

public boolean unregisterBroadcastStream(org.red5.server.api.IScope scope,
    String name);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

## 32.1. getBroadcastStreamNames(IScope)

```
public java.util.List<java.lang.String> getBroadcastStreamNames(org.red5.server.api.IScope scope);
```

**Specified by:** Method getBroadcastStreamNames in interface IProviderService

Get names of existing broadcast streams in a scope.

## 32.2. getLiveProviderInput(IScope, String, boolean)

```
public org.red5.server.messaging.IMessageInput getLiveProviderInput(org.red5.server.api.IScope scope,
    String name,
    boolean needCreate);
```

**Specified by:** Method getLiveProviderInput in interface IProviderService

Get a named Live provider as the source of input.

## 32.3. getProviderInput(IScope, String)

```
public org.red5.server.messaging.IMessageInput getProviderInput(org.red5.server.api.IScope scope,
```

```
String name);
```

**Specified by:** Method getProviderInput in interface IProviderService

Get a named provider as the source of input. Live stream first, VOD stream second.

### 32.4. getVODProviderFile(IScope, String)

```
public java.io.File getVODProviderFile(org.red5.server.api.IScope scope,
                                         String name);
```

**Specified by:** Method getVODProviderFile in interface IProviderService

Get a named VOD source file.

### 32.5. getVODProviderInput(IScope, String)

```
public org.red5.server.messaging.IMessageInput getVODProviderInput(org.red5.server.api.IScope scope,
                                                                    String name);
```

**Specified by:** Method getVODProviderInput in interface IProviderService

Get a named VOD provider as the source of input.

### 32.6. lookupProviderInput(IScope, String)

```
public org.red5.server.stream.IProviderService.INPUT_TYPE lookupProviderInput(org.red5.server.api.IScope scope,
                                                                           String name);
```

**Specified by:** Method lookupProviderInput in interface IProviderService

Returns the input type for a named provider if a source of input exists. Live is checked first and VOD second.

### 32.7. registerBroadcastStream(IScope, String, IBroadcastStream)

```
public boolean registerBroadcastStream(org.red5.server.api.IScope scope,
                                       String name,
                                       org.red5.server.api.stream.IBroadcastStream bs);
```

**Specified by:** Method registerBroadcastStream in interface IProviderService

Register a broadcast stream to a scope.

### 32.8. unregisterBroadcastStream(IScope, String)

```
public boolean unregisterBroadcastStream(org.red5.server.api.IScope scope,
                                         String name);
```

**Specified by:** Method unregisterBroadcastStream in interface IProviderService

Unregister a broadcast stream of a specific name from a scope.

## 33. Class ServerStream

An implementation for server side stream.

### 33.1. Synopsis

```

public class ServerStream extends org.?red5.?server.?stream.?AbstractStream
    implements org.?red5.?server.?api.?stream.?IServerStream, org.?red5.?server.?messaging.?IFilter,
// Protected Fields

    protected org.red5.server.api.stream.IPlaylistController controller ;

    protected org.red5.server.api.stream.IPlayItem currentItem ;

    protected org.red5.server.api.stream.IPlaylistController defaultController ;

    protected java.util.List<org.red5.server.api.stream.IPlayItem> items ;

    protected String publishedName ;

    protected String recordingFilename ;

// Public Constructors

    public ServerStream();

// Public Methods

    public synchronized void addItem(org.red5.server.api.stream.IPlayItem item);

    public synchronized void addItem(org.red5.server.api.stream.IPlayItem item,
                                    int index);

    public void addStreamListener(org.red5.server.api.stream.IStreamListener listener);

    public synchronized void close();

    public org.red5.server.api.stream.IPlayItem getCurrentItem();

    public int getCurrentItemIndex();

    public org.red5.server.api.stream.IPlayItem getItem(int index);

    public int getItemCount();

    public org.red5.server.messaging.IProvider getProvider();

    public String getPublishedName();

    public String getSaveFilename();

    public java.util.Collection<org.red5.server.api.stream.IStreamListener> getStreamListeners();

    public synchronized boolean hasMoreItems();

    public boolean isRandom();

    public boolean isRepeat();

```

## Package org.?red5.?server.?stream

```
public boolean isRewind();

public synchronized void nextItem();

public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                org.red5.server.messaging.IPipe pipe,
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);

public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);

public void pause();

public synchronized void previousItem();

public void pushMessage(org.red5.server.messaging.IPipe pipe,
                       org.red5.server.messaging.IMessage message)
throws IOException;

public synchronized void removeAllItems();

public synchronized void removeItem(int index);

public void removeStreamListener(org.red5.server.api.stream.IStreamListener listener);

public void saveAs(String name,
                   boolean isAppend)
throws IOException, ResourceNotFoundException, ResourceExistException;

public void seek(int position);

public synchronized void setItem(int index);

public void setPlaylistController(org.red5.server.api.stream.IPlaylistController controller);

public void setPublishedName(String name);

public void setRandom(boolean random);

public void setRepeat(boolean repeat);

public void setRewind(boolean rewind);

public void start();

public synchronized void stop();

// Protected Methods

protected org.red5.server.stream.message.RTMPMessage getNextRTMPMessage();

protected void moveToNext();

protected void moveToPrevious();

protected void notifyBroadcastClose();
```

```

protected void notifyBroadcastStart();

protected void onItemEnd();

protected void play(org.red5.server.api.stream.IPlayItem item);

protected void scheduleNextMessage();

protected void startBroadcastVOD();

}

```

**Methods inherited from org.red5.server.stream.AbstractStream:** getCodecInfo ,  
getName , getScope , getStreamAwareHandler , setCodecInfo , setName , setScope

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.stream.AbstractStream:** state

## 33.2. ServerStream()

```
public ServerStream();
```

Constructs a new ServerStream.

## 33.3. controller

```
protected org.red5.server.api.stream.IPlaylistController controller ;
```

Actual playlist controller

## 33.4. currentItem

```
protected org.red5.server.api.stream.IPlayItem currentItem ;
```

Current item

## 33.5. defaultController

```
protected org.red5.server.api.stream.IPlaylistController defaultController ;
```

Default playlist controller

## 33.6. items

```
protected java.util.List<org.red5.server.api.stream.IPlayItem> items ;
```

List of items in this playlist

## 33.7. publishedName

```
protected String publishedName ;
```

Stream published name

### 33.8. recordingFilename

```
protected String recordingFilename ;
```

The filename we are recording to.

### 33.9. addItem(IPlayItem)

```
public synchronized void addItem(org.red5.server.api.stream.IPlayItem item);
```

### 33.10. addItem(IPlayItem, int)

```
public synchronized void addItem(org.red5.server.api.stream.IPlayItem item,  
                                int index);
```

### 33.11. close()

```
public synchronized void close();
```

### 33.12. getCurrentItem()

```
public org.red5.server.api.stream.IPlayItem getCurrentItem();
```

### 33.13. getCurrentItemIndex()

```
public int getCurrentItemIndex();
```

### 33.14. getItem(int)

```
public org.red5.server.api.stream.IPlayItem getItem(int index);
```

### 33.15. getItemSize()

```
public int getItemSize();
```

### 33.16. getNextRTMPMessage()

```
protected org.red5.server.stream.message.RTMPMessage getNextRTMPMessage();
```

Getter for next RTMP message.

#### Parameters

<i>return</i>	Next RTMP message
---------------	-------------------

### 33.17. getProvider()

```
public org.red5.server.messaging.IProvider getProvider();
```

**33.18. getPublishedName()**

```
public String getPublishedName();
```

**33.19. getSaveFilename()**

```
public String getSaveFilename();
```

**33.20. hasMoreItems()**

```
public synchronized boolean hasMoreItems();
```

**33.21. isRandom()**

```
public boolean isRandom();
```

**33.22. isRepeat()**

```
public boolean isRepeat();
```

**33.23. isRewind()**

```
public boolean isRewind();
```

**33.24. moveToNext()**

```
protected void moveToNext();
```

Move to the next item updating the currentItemIndex. Should be called in synchronized context.

**33.25. moveToPrevious()**

```
protected void moveToPrevious();
```

Move to the previous item updating the currentItemIndex. Should be called in synchronized context.

**33.26. nextItem()**

```
public synchronized void nextItem();
```

**33.27. notifyBroadcastClose()**

```
protected void notifyBroadcastClose();
```

Notifies handler on stream broadcast stop

**33.28. notifyBroadcastStart()**

```
protected void notifyBroadcastStart();
```

Notifies handler on stream broadcast start

### 33.29. onItemEnd()

```
protected void onItemEnd();
```

Play next item on item end

### 33.30. onOOBControlMessage(IMessageComponent, IPipe, OOBControlMessage)

```
public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                org.red5.server.messaging.IPipe pipe,
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

### 33.31. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler. There are two types of pipe connection events so far, provider push connection event and provider disconnection event. Pipe events handling is the most common way of working with pipes.

#### Parameters

event	Pipe connection event context
-------	-------------------------------

### 33.32. pause()

```
public void pause();
```

**Specified by:** Method pause in interface IServerStream

Toggles the paused state.

### 33.33. play(IPlayItem)

```
protected void play(org.red5.server.api.stream.IPlayItem item);
```

Play a specific IPlayItem. The strategy for now is VOD first, Live second. Should be called in a synchronized context.

#### Parameters

item	Item to play
------	--------------

### 33.34. previousItem()

```
public synchronized void previousItem();
```

### 33.35. pushMessage(IPipe, IMessage)

```
public void pushMessage(org.red5.server.messaging.IPipe pipe,
                      org.red5.server.messaging.IMessage message)
```

```
throws IOException;
```

**Specified by:** Method pushMessage in interface IPushableConsumer

Pushes message through pipe

### 33.36. removeAllItems()

```
public synchronized void removeAllItems();
```

### 33.37. removeItem(int)

```
public synchronized void removeItem(int index);
```

### 33.38. saveAs(String, boolean)

```
public void saveAs(String name,
                   boolean isAppend)
throws IOException, ResourceNotFoundException, ResourceExistException;
```

### 33.39. scheduleNextMessage()

```
protected void scheduleNextMessage();
```

Pull the next message from IMessageInput and schedule it for push according to the timestamp.

### 33.40. seek(int)

```
public void seek(int position);
```

**Specified by:** Method seek in interface IServerStream

Seek to a given position in the stream.

### 33.41. setItem(int)

```
public synchronized void setItem(int index);
```

### 33.42. setPlaylistController(IPlaylistController)

```
public void setPlaylistController(org.red5.server.api.stream.IPlaylistController controller);
```

### 33.43. setPublishedName(String)

```
public void setPublishedName(String name);
```

### 33.44. setRandom(boolean)

```
public void setRandom(boolean random);
```

### 33.45. setRepeat(boolean)

```
public void setRepeat(boolean repeat);
```

### 33.46. setRewind(boolean)

```
public void setRewind(boolean rewind);
```

### 33.47. start()

```
public void start();
```

Start this server-side stream

### 33.48. startBroadcastVOD()

```
protected void startBroadcastVOD();
```

Begin VOD broadcasting

### 33.49. stop()

```
public synchronized void stop();
```

Stop this server-side stream

## 34. Class SimpleBWControlService

A simple implementation of bandwidth controller. The initial burst, if not specified by user, is half of the property "defaultCapacity".

Following is the reference information for the future optimization on threading: The threads that may access this object concurrently are: \* Thread A that makes token request. \* Thread B that makes token request. \* Thread C that distributes tokens and call the callbacks. (Timer) \* Thread D that updates the bw config of a controllable. \* Thread E that resets a bucket. \* Thread F that unregisters a controllable. The implementation now synchronizes on each context to make sure only one thread is accessing the context object at a time.

### 34.1. Synopsis

```
public class SimpleBWControlService extends, java.util.TimerTask
    implements, org.red5.server.stream.IBWControlService {
// Protected Fields

    protected java.util.Map<org.red5.server.api.IBWControllable, org.red5.server.stream.SimpleBWControlService> bwMap;
    protected long defaultCapacity ;
    protected long interval ;
    protected java.util.Timer tokenDistributor ;

// Public Constructors

    public SimpleBWControlService();

// Public Methods
```

```

public ITokenBucket getAudioBucket(IBWControlContext context);

public ITokenBucket getDataBucket(IBWControlContext context);

public ITokenBucket getVideoBucket(IBWControlContext context);

public void init();

public IBWControlContext lookupContext(org.red5.server.api.IBWControllable bc);

public IBWControlContext registerBWControllable(org.red5.server.api.IBWControllable bc);

public void resetBuckets(IBWControlContext context);

public void run();

public void setDefaultCapacity(long capacity);

public void setInterval(long interval);

public void shutdown();

public void unregisterBWControllable(IBWControlContext context);

public void updateBWConfigure(IBWControlContext context);

// Protected Methods

protected void invokeCallback(org.red5.server.stream.SimpleBWControlService.BWContext context);

protected boolean processRequest(org.red5.server.stream.SimpleBWControlService.TokenRequest request);

protected void rollbackRequest(org.red5.server.stream.SimpleBWControlService.TokenRequest request)

}

```

**Methods inherited from java.util.TimerTask:** cancel , run , scheduledExecutionTime

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 34.2. getAudioBucket(IBWControlContext)

```
public ITokenBucket getAudioBucket(IBWControlContext context);
```

**Specified by:** Method getAudioBucket in interface IBWControlService

Return the token bucket for audio channel.

### Parameters

context	The registry context.
---------	-----------------------

return	Token bucket for audio channel.
--------	---------------------------------

**Description copied from interface: getAudioBucket**

### 34.3. getDataBucket(IBWControlContext)

```
public ITokenBucket getDataBucket(IBWControlContext context);
```

**Specified by:** Method getDataBucket in interface IBWControlService

Return the token bucket for data channel.

**Parameters**

<i>context</i>	The registry context.
<i>return</i>	Token bucket for data channel.

**Description copied from interface: getDataBucket**

### 34.4. getVideoBucket(IBWControlContext)

```
public ITokenBucket getVideoBucket(IBWControlContext context);
```

**Specified by:** Method getVideoBucket in interface IBWControlService

Return the token bucket for video channel.

**Parameters**

<i>context</i>	The registry context.
<i>return</i>	Token bucket for video channel.

**Description copied from interface: getVideoBucket**

### 34.5. lookupContext(IBWControllable)

```
public IBWControlContext lookupContext(org.red5.server.api.IBWControllable bc);
```

**Specified by:** Method lookupContext in interface IBWControlService

Lookup the registry context according to the controllable.

**Parameters**

<i>bc</i>	The bandwidth controllable.
<i>return</i>	The registry context.

**Description copied from interface: lookupContext**

### 34.6. registerBWControllable(IBWControllable)

```
public IBWControlContext registerBWControllable(org.red5.server.api.IBWControllable bc);
```

**Specified by:** Method registerBWControllable in interface IBWControlService

Register a bandwidth controllable. The necessary resources will be allocated and assigned to the controllable.

Parameters	
bc	The bandwidth controllable.
return	The registry context. It's used in the subsequent calls to controller's method.

Description copied from interface: **registerBWControllable**

### 34.7. resetBuckets(IBWControlContext)

```
public void resetBuckets(IBWControlContext context);
```

**Specified by:** Method resetBuckets in interface IBWControlService

Reset all the token buckets for a controllable. All the callback will be reset and all blocked threads will be woken up.

Parameters	
context	The registry context.

Description copied from interface: **resetBuckets**

### 34.8. rollbackRequest(SimpleBWControlService.TokenRequest)

```
protected void rollbackRequest(org.red5.server.stream.SimpleBWControlService.TokenRequest request);
```

Give back the acquired tokens due to failing to accomplish the requested operation or over-charged tokens in the case of best-effort request.

Parameters	
request	

### 34.9. unregisterBWControllable(IBWControlContext)

```
public void unregisterBWControllable(IBWControlContext context);
```

**Specified by:** Method unregisterBWControllable in interface IBWControlService

Unregister the bandwidth controllable. The resources that were allocated will be freed.

Parameters	
context	The registry context.

Description copied from interface: **unregisterBWControllable**

### 34.10. updateBWConfigure(IBWControlContext)

```
public void updateBWConfigure(IBWControlContext context);
```

**Specified by:** Method updateBWConfigure in interface IBWControlService

Update the bandwidth configuration of a controllable. Each time when the controllable changes the bandwidth config and wants to make the changes take effect, this method should be called.

#### Parameters

context	The registry context.
---------	-----------------------

**Description copied from interface: updateBWConfigure**

## 35. Class SimpleBWControlService.BWContext

```
protected class SimpleBWControlService.BWContext implements org.?red5.?server.?stream.?IBWControlCont
// Public Constructors

    public SimpleBWControlService.BWContext(org.red5.server.api.IBWControllable controllable);

// Public Methods

    public org.red5.server.api.IBWControllable getBWControllable();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 35.1. getBWControllable()

```
public org.red5.server.api.IBWControllable getBWControllable();
```

**Specified by:** Method getBWControllable in interface IBWControlContext

Return the controllable that registered.

#### Parameters

return
--------

**Description copied from interface: getBWControllable**

## 36. Class SimpleBWControlService.TokenRequest

```
protected class SimpleBWControlService.TokenRequest {
// Protected Constructors

    protected SimpleBWControlService.TokenRequest();
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 37. Class SimpleBWControlService.TokenRequestContext

```
protected class SimpleBWControlService.TokenRequestContext {
// Protected Constructors

    protected SimpleBWControlService.TokenRequestContext();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 38. Class SimpleBWControlService.TokenRequestType

```
protected static final class SimpleBWControlService.TokenRequestType extends, java.?lang.?Enum {
// Public Static Fields

    public static final org.red5.server.stream.SimpleBWControlService.TokenRequestType BEST_EFFORT;

    public static final org.red5.server.stream.SimpleBWControlService.TokenRequestType BLOCKING;

    public static final org.red5.server.stream.SimpleBWControlService.TokenRequestType NONBLOCKING;

// Public Static Methods

    public static org.red5.server.stream.SimpleBWControlService.TokenRequestType valueOf(String name);

    public static org.red5.server.stream.SimpleBWControlService.TokenRequestType[] values();

}
```

**Methods inherited from java.lang.Enum:** clone, compareTo, equals, finalize, getDeclaringClass, hashCode, name, ordinal, toString, valueOf

**Methods inherited from java.lang.Object:** getClass, notify, notifyAll, wait

## 39. Class SimplePlaylistController

Simple playlist controller implementation

### 39.1. Synopsis

```
public class SimplePlaylistController implements, org.?red5.?server.?api.?stream.?IPlaylistController
// Public Constructors

    public SimplePlaylistController();

// Public Methods

    public int nextItem(org.red5.server.api.stream.IPlaylist playlist,
```

```

        int itemIndex);

public int previousItem(org.red5.server.api.stream.IPlaylist playlist,
                      int itemIndex);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 39.2. nextItem(IPlaylist, int)

```

public int nextItem(org.red5.server.api.stream.IPlaylist playlist,
                   int itemIndex);

```

**Specified by:** Method nextItem in interface IPlaylistController

Get next item to play.

## 39.3. previousItem(IPlaylist, int)

```

public int previousItem(org.red5.server.api.stream.IPlaylist playlist,
                      int itemIndex);

```

**Specified by:** Method previousItem in interface IPlaylistController

Get previous item to play.

# 40. Exception StreamNotFoundException

Throw when stream can't be found

## 40.1. Synopsis

```

public class StreamNotFoundException extends java.?lang.?Exception {
// Public Constructors

    public StreamNotFoundException(String name);
}

}

```

**Methods inherited from java.lang.Throwable:** fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, setStackTrace, toString

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait

# 41. Class StreamService

## Stream service

**41.1. Synopsis**

```

public class StreamService implements, org.?red5.?server.?api.?stream.?IStreamService {
// Public Constructors

    public StreamService();

// Public Static Methods

    public static void closeStream(org.red5.server.api.IConnection connection,
                                  int streamId);

    public static void sendNetStreamStatus(org.red5.server.api.IConnection conn,
                                         String statusCode,
                                         String description,
                                         String name,
                                         String status,
                                         int streamId);

// Public Methods

    public void closeStream();

    public int createStream();

    public void deleteStream(int streamId);

    public void deleteStream(org.red5.server.api.stream.IStreamCapableConnection conn,
                           int streamId);

    public IBroadcastScope getBroadcastScope(org.red5.server.api.IScope scope,
                                             String name);

    public void pause(boolean pausePlayback,
                     int position);

    public void pause(Boolean pausePlayback,
                     int position);

    public void play(Boolean dontStop);

    public void play(String name);

    public void play(String name,
                    int start);

    public void play(String name,
                    int start,
                    int length);

    public void play(String name,
                    int start,
                    int length,
                    boolean flushPlaylist);

    public void play(String name,
                    int start,
                    int length);

```

```

        int length,
        Object flushPlaylist);

public void publish(Boolean dontStop);

public void publish(String name);

public void publish(String name,
                   String mode);

public void receiveAudio(boolean receive);

public void receiveVideo(boolean receive);

public void releaseStream(String streamName);

public void seek(int position);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 41.2. closeStream()

```
public void closeStream();
```

**Specified by:** Method closeStream in interface IStreamService

Close the stream but not deallocate the resources.

## 41.3. closeStream(IConnection, int)

```
public static void closeStream(org.red5.server.api.IConnection connection,
                             int streamId);
```

Close stream. This method can close both IClientBroadcastStream (coming from Flash Player to Red5) and ISubscriberStream (from Red5 to Flash Player). Corresponding application handlers (streamSubscriberClose, etc.) are called as if close was initiated by Flash Player. It is recommended to remember stream id in application handlers, ex.:

```

public void streamBroadcastStart(IBroadcastStream stream) {
    super.streamBroadcastStart(stream);
    if (stream instanceof IClientBroadcastStream) {
        int publishedStreamId = ((ClientBroadcastStream)stream).getStreamId();
        Red5.getConnectionLocal().setAttribute(PUBLISHED_STREAM_ID_ATTRIBUTE, pu\
blishedStreamId);
    }
}

```

```

public void streamPlaylistItemPlay(IPlaylistSubscriberStream stream, IPlay\
Item item, boolean isLive) {
    super.streamPlaylistItemPlay(stream, item, isLive);
}

```

```
Red5.getConnectionLocal().setAttribute(WATCHED_STREAM_ID_ATTRIBUTE, strea\
m.getId());
}
```

When stream is closed, corresponding NetStream status will be sent to stream provider / consumers. Implementation is based on Red5's StreamService.close()

Parameters	
connection	client connection
streamId	stream ID (number: 1,2,...)

#### 41.4. createStream()

```
public int createStream();
```

**Specified by:** Method createStream in interface IStreamService

Create a stream and return a corresponding id.

#### 41.5. deleteStream(int)

```
public void deleteStream(int streamId);
```

**Specified by:** Method deleteStream in interface IStreamService

Close the stream if not been closed. Deallocate the related resources.

#### 41.6. deleteStream(IStreamCapableConnection, int)

```
public void deleteStream(org.red5.server.api.stream.IStreamCapableConnection conn,
int streamId);
```

**Specified by:** Method deleteStream in interface IStreamService

Delete stream

#### 41.7. getBroadcastScope(IScope, String)

```
public IBroadcastScope getBroadcastScope(org.red5.server.api.IScope scope,
String name);
```

Return broadcast scope object for given scope and child scope name

Parameters	
scope	Scope object
name	Child scope name
return	Broadcast scope

#### 41.8. pause(boolean, int)

```
public void pause(boolean pausePlayback,
```

```
int position);
```

**Specified by:** Method pause in interface IStreamService

Pauses playback

#### 41.9. pause(Boolean, int)

```
public void pause(Boolean pausePlayback,
                  int position);
```

Pause at given position. Required as "pausePlayback" can be "null" if no flag is passed by the client

##### Parameters

pausePlayback	Pause playback or not
position	Pause position

#### 41.10. play(Boolean)

```
public void play(Boolean dontStop);
```

**Specified by:** Method play in interface IStreamService

Play stream without initial stop

#### 41.11. play(String)

```
public void play(String name);
```

**Specified by:** Method play in interface IStreamService

Play stream with name

#### 41.12. play(String, int)

```
public void play(String name,
                 int start);
```

**Specified by:** Method play in interface IStreamService

Play stream with name from start position

#### 41.13. play(String, int, int)

```
public void play(String name,
                 int start,
                 int length);
```

**Specified by:** Method play in interface IStreamService

Play stream with name from start position and for given amount if time

## 41.14. play(String, int, int, boolean)

```
public void play(String name,
                 int start,
                 int length,
                 boolean flushPlaylist);
```

**Specified by:** Method play in interface IStreamService

Publishes stream from given position for given amount of time

## 41.15. play(String, int, int, Object)

```
public void play(String name,
                 int start,
                 int length,
                 Object flushPlaylist);
```

## 41.16. publish(Boolean)

```
public void publish(Boolean dontStop);
```

**Specified by:** Method publish in interface IStreamService

Publish

## 41.17. publish(String)

```
public void publish(String name);
```

**Specified by:** Method publish in interface IStreamService

Publishes stream with given name

## 41.18. publish(String, String)

```
public void publish(String name,
                    String mode);
```

**Specified by:** Method publish in interface IStreamService

Publishes stream with given name and mode

## 41.19. receiveAudio(boolean)

```
public void receiveAudio(boolean receive);
```

**Specified by:** Method receiveAudio in interface IStreamService

Can recieve audio?

## 41.20. receiveVideo(boolean)

```
public void receiveVideo(boolean receive);
```

**Specified by:** Method receiveVideo in interface IStreamService

Can receive video?

### 41.21. releaseStream(String)

```
public void releaseStream(String streamName);
```

**Specified by:** Method releaseStream in interface IStreamService

Called by FME.

### 41.22. seek(int)

```
public void seek(int position);
```

**Specified by:** Method seek in interface IStreamService

Seek to position

### 41.23. sendNetStreamStatus(IConnection, String, String, String, String, int)

```
public static void sendNetStreamStatus(org.red5.server.api.IConnection conn,
                                       String statusCode,
                                       String description,
                                       String name,
                                       String status,
                                       int streamId);
```

Send NetStream.Status to client (Flash Player)

#### Parameters

conn	
statusCode	NetStream status code
description	
name	
status	The status - error, warning, or status
streamId	

## 42. Class StreamTracker

```
public class StreamTracker implements org.?red5.?server.?net.?rtmp.?message.?Constants {
    // Public Constructors

    public StreamTracker();

    // Public Methods

    public int add(org.red5.server.net.rtmp.event.IRTMPEvent event);

    public boolean isRelative();

    public void reset();
```

```
}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 42.1. StreamTracker()

```
public StreamTracker();
```

Constructs a new StreamTracker.

## 42.2. add(IRTMPEvent)

```
public int add(org.red5.server.net.rtmp.event.IRTMPEvent event);
```

RTMP event handler

### Parameters

event	RTMP event
<i>return</i>	Timeframe since last notification (or audio or video packet sending)

## 42.3. isRelative()

```
public boolean isRelative();
```

Getter for property 'relative'.

### Parameters

<i>return</i>	Value for property 'relative'.
---------------	--------------------------------

## 42.4. reset()

```
public void reset();
```

Reset state

## 43. Class StreamingProxy

A proxy to publish stream from server to server. TODO: Use timer to monitor the connect/stream creation.

## 43.1. Synopsis

```
public class StreamingProxy implements, org.?red5.?server.?messaging.?IPushableConsumer, org.?red5.?se
// Public Constructors

public StreamingProxy();
```

```
// Public Methods

    public void init();

    public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                    org.red5.server.messaging.IPipe pipe,
                                    org.red5.server.messaging.OOBControlMessage oobCtrlMsg);

    public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);

    public synchronized void onStreamEvent(org.red5.server.net.rtmp.event.Notify notify);

    public synchronized void pushMessage(org.red5.server.messaging.IPipe pipe,
                                       org.red5.server.messaging.IMessage message)
        throws IOException;

    public synchronized void resultReceived(org.red5.server.api.service.IPendingServiceCall call);

    public void setApp(String app);

    public void setHost(String host);

    public void setPort(int port);

    public synchronized void start(String publishName);

    public synchronized void stop();

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 43.2. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler

### Parameters

event	Pipe connection event
-------	-----------------------

**Description copied from interface: onPipeConnectionEvent**

## 43.3. pushMessage(IPipe, IMessage)

```
public synchronized void pushMessage(org.red5.server.messaging.IPipe pipe,
                                   org.red5.server.messaging.IMessage message)
        throws IOException;
```

**Specified by:** Method pushMessage in interface IPushableConsumer

Pushes message through pipe

#### Parameters

pipe	Pipe
message	Message

IOException  
if message could not be written

**Description copied from interface: pushMessage**

## 43.4. resultReceived(IPendingServiceCall)

```
public synchronized void resultReceived(org.red5.server.api.service.IPendingServiceCall call);
```

**Specified by:** Method resultReceived in interface IPendingServiceCallback

Triggered when results are received

#### Parameters

call	Call object this callback is applied to
------	---

**Description copied from interface: resultReceived**

## 44. Class VideoCodecFactory

Factory for video codecs. Creates and returns video codecs

### 44.1. Synopsis

```
public class VideoCodecFactory {
// Public Static Fields

    public static final String KEY = "videoCodecFactory";

// Public Constructors

    public VideoCodecFactory();

// Public Methods

    public org.red5.server.api.stream.IVideoStreamCodec getVideoCodec(org.apache.mina.common.ByteBuffer

    public void setCodecs(java.util.List<org.red5.server.api.stream.IVideoStreamCodec> codecs);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

### 44.2. KEY

```
public static final String KEY = "videoCodecFactory";
```

Object key

### 44.3. getVideoCodec(ByteBuffer)

```
public org.red5.server.api.stream.IVideoStreamCodec getVideoCodec(org.apache.mina.common.ByteBuffer
```

Create and return new video codec applicable for byte buffer data

#### Parameters

data	Byte buffer data
<i>return</i>	Video codec

### 44.4. setCodecs(List<IVideoStreamCodec>)

```
public void setCodecs(java.util.List<org.red5.server.api.stream.IVideoStreamCodec> codecs);
```

Setter for codecs

#### Parameters

codecs	List of codecs
--------	----------------

## 45. Class VideoFrameDropper

State machine for video frame dropping in live streams.

We start sending all frame types. Disposable interframes can be dropped any time without affecting the current state. If a regular interframe is dropped, all future frames up to the next keyframes are dropped as well. Dropped keyframes result in only keyframes being sent. If two consecutive keyframes have been successfully sent, regular interframes will be sent in the next iteration as well. If these frames all went through, disposable interframes are sent again.

So from highest to lowest bandwidth and back, the states go as follows:

- all frames
- keyframes and interframes
- keyframes
- keyframes and interframes
- all frames

### 45.1. Synopsis

```
public class VideoFrameDropper implements org.?red5.?server.?stream.?IFrameDropper {
// Protected Fields
```

```

protected static org.slf4j.Logger log ;

// Public Constructors

public VideoFrameDropper();

// Public Methods

public boolean canSendPacket(org.red5.server.stream.message.RTMPMessage message,
                           long pending);

public void dropPacket(org.red5.server.stream.message.RTMPMessage message);

public void reset();

public void reset(int state);

public void sendPacket(org.red5.server.stream.message.RTMPMessage message);

}

```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,  
notify , notifyAll , toString , wait

## 45.2. VideoFrameDropper()

```
public VideoFrameDropper();
```

Constructs a new VideoFrameDropper.

## 45.3. canSendPacket(RTMPMessage, long)

```
public boolean canSendPacket(org.red5.server.stream.message.RTMPMessage message,
                           long pending);
```

**Specified by:** Method canSendPacket in interface IFrameDropper

Checks if a message may be sent to the subscriber.

## 45.4. dropPacket(RTMPMessage)

```
public void dropPacket(org.red5.server.stream.message.RTMPMessage message);
```

**Specified by:** Method dropPacket in interface IFrameDropper

Notify that a packet has been dropped.

## 45.5. reset()

```
public void reset();
```

**Specified by:** Method reset in interface IFrameDropper

Reset the frame dropper.

## 45.6. reset(int)

```
public void reset(int state);
```

**Specified by:** Method reset in interface IFrameDropper

Reset the frame dropper to a given state.

## 45.7. sendPacket(RTMPMessage)

```
public void sendPacket(org.red5.server.stream.message.RTMPMessage message);
```

**Specified by:** Method sendPacket in interface IFrameDropper

Notify that a message has been sent.

# 1. Class ScreenVideo

Red5 video codec for the screen capture format.

## 1.1. Synopsis

```
public class ScreenVideo implements org.red5.server.api.stream.IVideoStreamCodec {  
    // Public Constructors  
  
    public ScreenVideo();  
  
    // Public Methods  
  
    public boolean addData(org.apache.mina.common.ByteBuffer data);  
  
    public boolean canDropFrames();  
  
    public boolean canHandleData(org.apache.mina.common.ByteBuffer data);  
  
    public org.apache.mina.common.ByteBuffer getKeyframe();  
  
    public String getName();  
  
    public void reset();  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. ScreenVideo()

```
public ScreenVideo();
```

Constructs a new ScreenVideo.

## 1.3. addData(ByteBuffer)

```
public boolean addData(org.apache.mina.common.ByteBuffer data);
```

**Specified by:** Method addData in interface IVideoStreamCodec

Update the state of the codec with the passed data.

## 1.4. canDropFrames()

```
public boolean canDropFrames();
```

**Specified by:** Method canDropFrames in interface IVideoStreamCodec

Check if the codec supports frame dropping.

## 1.5. canHandleData(ByteBuffer)

```
public boolean canHandleData(org.apache.mina.common.ByteBuffer data);
```

**Specified by:** Method canHandleData in interface IVideoStreamCodec

Returns true if the codec knows how to handle the passed stream data.

## 1.6. getKeyframe()

```
public org.apache.mina.common.ByteBuffer getKeyframe();
```

**Specified by:** Method getKeyframe in interface IVideoStreamCodec

Return the data for a keyframe.

## 1.7. getName()

```
public String getName();
```

**Specified by:** Method getName in interface IVideoStreamCodec

Return the name of the video codec.

## 1.8. reset()

```
public void reset();
```

**Specified by:** Method reset in interface IVideoStreamCodec

Reset the codec to its initial state.

## 2. Class SorensonVideo

Red5 video codec for the sorenson video format. VERY simple implementation, just stores last keyframe.

### 2.1. Synopsis

```
public class SorensonVideo implements, org.?red5.?server.?api.?stream.?IVideoStreamCodec {
// Public Constructors

    public SorensonVideo();

// Public Methods

    public boolean addData(org.apache.mina.common.ByteBuffer data);

    public boolean canDropFrames();

    public boolean canHandleData(org.apache.mina.common.ByteBuffer data);

    public org.apache.mina.common.ByteBuffer getKeyframe();

    public String getName();
```

```
public void reset();
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. SorensonVideo()

```
public SorensonVideo();
```

Constructs a new SorensonVideo.

## 2.3. addData(ByteBuffer)

```
public boolean addData(org.apache.mina.common.ByteBuffer data);
```

**Specified by:** Method addData in interface IVideoStreamCodec

Update the state of the codec with the passed data.

## 2.4. canDropFrames()

```
public boolean canDropFrames();
```

**Specified by:** Method canDropFrames in interface IVideoStreamCodec

Check if the codec supports frame dropping.

## 2.5. canHandleData(ByteBuffer)

```
public boolean canHandleData(org.apache.mina.common.ByteBuffer data);
```

**Specified by:** Method canHandleData in interface IVideoStreamCodec

Returns true if the codec knows how to handle the passed stream data.

## 2.6. getKeyframe()

```
public org.apache.mina.common.ByteBuffer getKeyframe();
```

**Specified by:** Method getKeyframe in interface IVideoStreamCodec

Return the data for a keyframe.

## 2.7. getName()

```
public String getName();
```

**Specified by:** Method getName in interface IVideoStreamCodec

Return the name of the video codec.

## 2.8. reset()

```
public void reset();
```

**Specified by:** Method reset in interface IVideoStreamCodec

Reset the codec to its initial state.

## 3. Class StreamCodecInfo

```
public class StreamCodecInfo implements org.red5.server.api.stream.IStreamCodecInfo {
    // Public Constructors

    public StreamCodecInfo();

    // Public Methods

    public String getAudioCodecName();

    public org.red5.server.api.stream.IVideoStreamCodec getVideoCodec();

    public String getVideoCodecName();

    public boolean hasAudio();

    public boolean hasVideo();

    public void setHasAudio(boolean value);

    public void setHasVideo(boolean value);

    public void setVideoCodec(org.red5.server.api.stream.IVideoStreamCodec codec);
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

### 3.1. getAudioCodecName()

```
public String getAudioCodecName();
```

**Specified by:** Method getAudioCodecName in interface IStreamCodecInfo

Getter for audio codec name

### 3.2. getVideoCodec()

```
public org.red5.server.api.stream.IVideoStreamCodec getVideoCodec();
```

**Specified by:** Method getVideoCodec in interface IStreamCodecInfo

Return video codec

### 3.3. getVideoCodecName()

```
public String getVideoCodecName();
```

**Specified by:** Method getVideoCodecName in interface IStreamCodecInfo

Getter for video codec name

### 3.4. hasAudio()

```
public boolean hasAudio();
```

**Specified by:** Method hasAudio in interface IStreamCodecInfo

Has audio support?

### 3.5. hasVideo()

```
public boolean hasVideo();
```

**Specified by:** Method hasVideo in interface IStreamCodecInfo

Has video support?

### 3.6. setHasAudio(boolean)

```
public void setHasAudio(boolean value);
```

New value for audio support

Parameters

value	Audio support
-------	---------------

### 3.7. setHasVideo(boolean)

```
public void setHasVideo(boolean value);
```

New value for video support

Parameters

value	Video support
-------	---------------

### 3.8. setVideoCodec(IVideoStreamCodec)

```
public void setVideoCodec(org.red5.server.api.stream.IVideoStreamCodec codec);
```

Setter for video codec

Parameters

codec	Video codec
-------	-------------

# 1. Class ConnectionConsumer

RTMP connection consumer.

## 1.1. Synopsis

```
public class ConnectionConsumer implements, org.red5.server.messaging.IPushableConsumer, org.red5.server.messaging.IControlConsumer
// Public Static Fields

    public static final String KEY;

// Public Constructors

    public ConnectionConsumer(org.red5.server.net.rtmp.RTMPConnection conn,
                            int videoChannel,
                            int audioChannel,
                            int dataChannel);

// Public Methods

    public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                    org.red5.server.messaging.IPipe pipe,
                                    org.red5.server.messaging.OOBControlMessage oobCtrlMsg);

    public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);

    public void pushMessage(org.red5.server.messaging.IPipe pipe,
                           org.red5.server.messaging.IMessage message);

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. ConnectionConsumer(RTMPConnection, int, int, int)

```
public ConnectionConsumer(org.red5.server.net.rtmp.RTMPConnection conn,
                        int videoChannel,
                        int audioChannel,
                        int dataChannel);
```

Create rtmp connection consumer for given connection and channels

### Parameters

conn	RTMP connection
videoChannel	Video channel
audioChannel	Audio channel
dataChannel	Data channel

## 1.3. KEY

```
public static final String KEY;
```

Connection consumer class name

## 1.4. onOOBControlMessage(IMessageComponent, IPipe, OOBControlMessage)

```
public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                org.red5.server.messaging.IPipe pipe,
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

## 1.5. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler

## 1.6. pushMessage(IPipe, IMessage)

```
public void pushMessage(org.red5.server.messaging.IPipe pipe,
                       org.red5.server.messaging.IMessage message);
```

**Specified by:** Method pushMessage in interface IPushableConsumer

Pushes message through pipe

## 2. Class FileConsumer

Consumer that pushes messages to file. Used when recording live streams.

### 2.1. Synopsis

```
public class FileConsumer implements, org.?red5.?server.?net.?rtmp.?message.?Constants, org.?red5.?ser
// Public Constructors

    public FileConsumer(org.red5.server.api.IScope scope,
                      java.io.File file);

// Public Methods

    public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                    org.red5.server.messaging.IPipe pipe,
                                    org.red5.server.messaging.OOBControlMessage oobCtrlMsg);

    public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);

    public void pushMessage(org.red5.server.messaging.IPipe pipe,
                           org.red5.server.messaging.IMessage message)
        throws IOException;

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

## 2.2. FileConsumer(IScope, File)

```
public FileConsumer(org.red5.server.api.IScope scope,
                   java.io.File file);
```

Creates file consumer

### Parameters

scope	Scope of consumer
file	File

## 2.3. onOOBControlMessage(IMessageComponent, IPipe, OOBControlMessage)

```
public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                org.red5.server.messaging.IPipe pipe,
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

Out-of-band control message handler

### Parameters

source	Source of message
pipe	Pipe that is used to transmit OOB message
oobCtrlMsg	OOB control message

## 2.4. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler

### Parameters

event	Pipe connection event
-------	-----------------------

## 2.5. pushMessage(IPipe, IMessage)

```
public void pushMessage(org.red5.server.messaging.IPipe pipe,
                       org.red5.server.messaging.IMessage message)
                       throws IOException;
```

**Specified by:** Method pushMessage in interface IPushableConsumer

Push message through pipe Synchronize this method to avoid FLV corruption from abrupt disconnection

### Parameters

pipe	Pipe
message	Message to push

IOException

if message could not be written

# 1. Class StreamBandwidthController

Controls stream bandwidth

## 1.1. Synopsis

```
public class StreamBandwidthController implements org.red5.server.messaging.IFilter, org.red5.s  
// Public Static Fields  
  
    public static final String KEY ;  
  
// Public Constructors  
  
    public StreamBandwidthController();  
  
// Public Methods  
  
    public void close();  
  
    public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,  
                                    org.red5.server.messaging.IPipe pipe,  
                                    org.red5.server.messaging.OOBControlMessage oobCtrlMsg);  
  
    public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);  
  
    public void run();  
  
    public void start();  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. KEY

```
public static final String KEY ;
```

Class name

## 1.3. close()

```
public void close();
```

Stop pulling, close stream

## 1.4. onOOBControlMessage(IMessageComponent, IPipe, OOBControlMessage)

```
public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,  
                                org.red5.server.messaging.IPipe pipe,  
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

## 1.5. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler

## 1.6. run()

```
public void run();
```

**Specified by:** Method run in interface Runnable

## 1.7. start()

```
public void start();
```

Start pulling (streaming)

# 1. Class RTMPMessage

RTMP message

## 1.1. Synopsis

```
public class RTMPMessage extends org.red5.server.messaging.AbstractMessage {  
    // Public Constructors  
  
    public RTMPMessage();  
  
    // Public Methods  
  
    public org.red5.server.net.rtmp.event.IRTMPEvent getBody();  
  
    public void setBody(org.red5.server.net.rtmp.event.IRTMPEvent body);  
}
```

### Methods inherited from org.red5.server.messaging.AbstractMessage:

getBooleanProperty , getByteProperty , getCorrelationID , getDoubleProperty ,  
getFloatProperty , getIntProperty , getLongProperty , getMessageID , getMessageType  
, getProperty , getShortProperty , getStringProperty , setBooleanProperty  
, setByteProperty , setCorrelationID , setDoubleProperty , setFloatProperty ,  
setIntProperty , setLongProperty , setMessageID , setMessageType , setObjectProperty  
, setShortProperty , setStringProperty

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.messaging.AbstractMessage:** correlationID ,  
extraHeaders , messageID , messageType

## 1.2. getBody()

```
public org.red5.server.net.rtmp.event.IRTMPEvent getBody();
```

Return RTMP message body

### Parameters

return	Value for property 'body'.
--------	----------------------------

## 1.3. setBody(IERTMPEvent)

```
public void setBody(org.red5.server.net.rtmp.event.IRTMPEvent body);
```

Setter for RTMP message body

### Parameters

body	Value to set for property 'body'.
------	-----------------------------------

## 2. Class ResetMessage

To notify the client to reset the playing state.

### 2.1. Synopsis

```
public class ResetMessage extends, org.?red5.?server.?messaging.?AbstractMessage {
// Public Constructors

    public ResetMessage();

}
```

#### **Methods inherited from org.red5.server.messaging.AbstractMessage:**

getBooleanProperty , getByteProperty , getCorrelationID , getDoubleProperty ,  
 getFloatProperty , getIntProperty , getLongProperty , getMessageID , getMessageType  
 , getObjectProperty , getShortProperty , getStringProperty , setBooleanProperty  
 , setByteProperty , setCorrelationID , setDoubleProperty , setFloatProperty ,  
 setIntProperty , setLongProperty , setMessageID , setMessageType , setObjectProperty  
 , setShortProperty , setStringProperty

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
 , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.messaging.AbstractMessage:** correlationID ,  
 extraHeaders , messageID , messageType

## 3. Class StatusMessage

```
public class StatusMessage extends, org.?red5.?server.?messaging.?AbstractMessage {
// Public Constructors

    public StatusMessage();

// Public Methods

    public org.red5.server.net.rtmp.status.Status getBody();

    public void setBody(org.red5.server.net.rtmp.status.Status body);

}
```

#### **Methods inherited from org.red5.server.messaging.AbstractMessage:**

getBooleanProperty , getByteProperty , getCorrelationID , getDoubleProperty ,  
 getFloatProperty , getIntProperty , getLongProperty , getMessageID , getMessageType  
 , getObjectProperty , getShortProperty , getStringProperty , setBooleanProperty  
 , setByteProperty , setCorrelationID , setDoubleProperty , setFloatProperty ,  
 setIntProperty , setLongProperty , setMessageID , setMessageType , setObjectProperty  
 , setShortProperty , setStringProperty

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
 , notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.messaging.AbstractMessage:** correlationID , extraHeaders , messageID , messageType

### 3.1. **getBody()**

```
public org.red5.server.net.rtmp.status.Status getBody();
```

Getter for property 'body'.

#### Parameters

<i>return</i>	Value for property 'body'.
---------------	----------------------------

### 3.2. **setBody(Status)**

```
public void setBody(org.red5.server.net.rtmp.status.Status body);
```

Setter for property 'body'.

#### Parameters

<i>body</i>	Value to set for property 'body'.
-------------	-----------------------------------

# 1. Class ConnectionProvider

Provides connection via pipe

## 1.1. Synopsis

```
public class ConnectionProvider implements, org.red5.server.messaging.IProvider, org.red5.server.  
// Public Constructors  
  
    public ConnectionProvider();  
  
// Public Methods  
  
    public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,  
                                    org.red5.server.messaging.IPipe pipe,  
                                    org.red5.server.messaging.OOBControlMessage oobCtrlMsg);  
  
    public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. onOOBControlMessage(IMessageComponent, IPipe, OOBControlMessage)

```
public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,  
                                org.red5.server.messaging.IPipe pipe,  
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

## 1.3. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler

# 2. Class FileProvider

Pullable provider for files

## 2.1. Synopsis

```
public class FileProvider implements, org.red5.server.messaging.IPassive, org.red5.server.  
// Public Static Fields  
  
    public static final String KEY ;  
  
// Public Constructors  
  
    public FileProvider(org.red5.server.api.IScope scope,
```

```

        java.io.File file);

// Public Methods

    public boolean hasVideo();

    public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                    org.red5.server.messaging.IPipe pipe,
                                    org.red5.server.messaging.OOBControlMessage oobCtrlMsg);

    public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);

    public synchronized org.red5.server.messaging.IMessage pullMessage(org.red5.server.messaging.IPipe
        throws IOException;

    public org.red5.server.messaging.IMessage pullMessage(org.red5.server.messaging.IPipe pipe,
        long wait)
        throws IOException;

    public synchronized int seek(int ts);

    public void setStart(int start);

}

```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. FileProvider(I Scope, File)

```

public FileProvider(org.red5.server.api.IScope scope,
                    java.io.File file);

```

Create file provider for given file and scope

### Parameters

scope	Scope
file	File

## 2.3. KEY

```

public static final String KEY ;

```

Class name

## 2.4. hasVideo()

```

public boolean hasVideo();

```

**Specified by:** Method hasVideo in interface IStreamTypeAwareProvider

Check if the provider contains video tags.

## 2.5. onOOBControlMessage(IMessageComponent, IPipe, OOBControlMessage)

```
public void onOOBControlMessage(org.red5.server.messaging.IMessageComponent source,
                                org.red5.server.messaging.IPipe pipe,
                                org.red5.server.messaging.OOBControlMessage oobCtrlMsg);
```

## 2.6. onPipeConnectionEvent(PipeConnectionEvent)

```
public void onPipeConnectionEvent(org.red5.server.messaging.PipeConnectionEvent event);
```

**Specified by:** Method onPipeConnectionEvent in interface IPipeConnectionListener

Pipe connection event handler

## 2.7. pullMessage(IPipe)

```
public synchronized org.red5.server.messaging.IMessage pullMessage(org.red5.server.messaging.IPipe pipe)
throws IOException;
```

**Specified by:** Method pullMessage in interface IPullableProvider

## 2.8. pullMessage(IPipe, long)

```
public org.red5.server.messaging.IMessage pullMessage(org.red5.server.messaging.IPipe pipe,
                                                       long wait)
throws IOException;
```

**Specified by:** Method pullMessage in interface IPullableProvider

## 2.9. seek(int)

```
public synchronized int seek(int ts);
```

**Specified by:** Method seek in interface ISeekableProvider

Seek the provider to timestamp ts (in milliseconds).

## 2.10. setStart(int)

```
public void setStart(int start);
```

Setter for start position

### Parameters

start	Start position
-------	----------------

# 1. Class TomcatApplicationContext

Class that wraps a Tomcat webapp context.

## 1.1. Synopsis

```
public class TomcatApplicationContext implements org.red5.server.api.IApplicationContext {  
    // Protected Fields  
  
    protected static org.slf4j.Logger log ;  
  
    // Protected Constructors  
  
    protected TomcatApplicationContext(org.apache.catalina.Context context);  
  
    // Public Methods  
  
    public void stop();  
  
}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. TomcatApplicationContext(Context)

```
protected TomcatApplicationContext(org.apache.catalina.Context context);
```

Wrap the passed Tomcat webapp context.

### Parameters

context
---------

## 1.3. log

```
protected static org.slf4j.Logger log ;
```

Logger

## 1.4. stop()

```
public void stop();
```

**Specified by:** Method stop in interface IApplicationContext

Stop the web application.

**Description copied from interface: stop**

# 2. Class TomcatApplicationLoader

Class that can load new applications in Tomcat.

## 2.1. Synopsis

```
public class TomcatApplicationLoader implements org.?red5.?server.?api.?IApplicationLoader {
    // Protected Fields

    protected static org.slf4j.Logger log;

    // Protected Constructors

    protected TomcatApplicationLoader(org.apache.catalina.startup.Embedded embedded,
                                      org.apache.catalina.Host host,
                                      org.springframework.context.ApplicationContext rootCtx);

    // Public Methods

    public org.springframework.context.ApplicationContext getRootContext();

    public void loadApplication(String contextPath,
                               String virtualHosts,
                               String directory)
        throws Exception;

}
```

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 2.2. TomcatApplicationLoader(Embedded, Host, ApplicationContext)

```
protected TomcatApplicationLoader(org.apache.catalina.startup.Embedded embedded,
                                 org.apache.catalina.Host host,
                                 org.springframework.context.ApplicationContext rootCtx);
```

Wrap Tomcat engine and host.

### Parameters

embedded	
----------	--

host	
------	--

## 2.3. getRootContext()

```
public org.springframework.context.ApplicationContext getRootContext();
```

**Specified by:** Method getRootContext in interface IApplicationLoader

Return the root org.springframework.context.ApplicationContext.

## 2.4. loadApplication(String, String, String)

```
public void loadApplication(String contextPath,
                           String virtualHosts,
```

```
    String directory)
throws Exception;
```

**Specified by:** Method loadApplication in interface IApplicationLoader

Load a new application for the given context path from a directory.

## 3. Class TomcatLoader

Red5 loader for Tomcat.

### 3.1. Synopsis

```
public class TomcatLoader extends, org.?red5.?server.?LoaderBase
    implements, org.?springframework.?context.?ApplicationContextAware, org.?red5.?server.?LoaderMBean
// Public Static Fields

    public static final String defaultParentContextKey = "default.context";

    public static final String defaultSpringConfigLocation = "/WEB-INF/red5-* .xml";

// Protected Fields

    protected java.util.Map<java.lang.String, java.lang.String> connectionProperties;

    protected org.apache.catalina.connector.Connector connector;

    protected static org.apache.catalina.startup.Embedded embedded;

    protected static org.apache.catalina.Engine engine;

    protected org.apache.catalina.Host host;

    protected org.apache.catalina.Realm realm;

    protected java.util.List<org.apache.catalina.Valve> valves;

// Public Constructors

    public TomcatLoader();

// Public Methods

    public org.apache.catalina.Context addContext(String path,
                                                String docBase);

    public org.apache.catalina.Host getBaseHost();

    public org.apache.catalina.connector.Connector getConnector();

    public org.apache.catalina.startup.Embedded getEmbedded();

    public org.apache.catalina.Engine getEngine();

    public org.apache.catalina.Host getHost();

    public org.apache.catalina.Realm getRealm();
```

```

public void init();

public void registerJMX();

public void removeContext(String path);

public void setBaseHost(org.apache.catalina.Host baseHost);

public void setConnectionProperties(java.util.Map<java.lang.String, java.lang.String> props);

public void setConnector(org.apache.catalina.connector.Connector connector);

public void setConnectors(java.util.List<org.apache.catalina.connector.Connector> connectors);

public void setContexts(java.util.Map<java.lang.String, java.lang.String> contexts);

public void setEmbedded(org.apache.catalina.startup.Embedded embedded);

public void setHost(org.apache.catalina.Host host);

public void setHosts(java.util.List<org.apache.catalina.Host> hosts);

public void setRealm(org.apache.catalina.Realm realm);

public void setValves(java.util.List<org.apache.catalina.Valve> valves);

public void shutdown();

// Protected Methods

protected static String formatPath(String absWebappsPath,
                                    String contextDirName);

}

```

**Direct known subclasses:** org.?red5.?server.?net.?rtmpt.?TomcatRTMPTLoader , org.?red5.?server.?tomcat.?TomcatVHostLoader

**Methods inherited from org.red5.server.LoaderBase:** getApplicationContext ,  
getApplicationLoader , getRed5ApplicationContext , removeContext ,  
removeRed5ApplicationContext , setApplicationContext , setApplicationLoader ,  
setRed5ApplicationContext , setWebappFolder

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.LoaderBase:** applicationContext , loader ,  
red5AppCtx , webappFolder

## 3.2. connectionProperties

```
protected java.util.Map<java.lang.String, java.lang.String> connectionProperties ;
```

Additional connection properties to be set at init.

### 3.3. connector

```
protected org.apache.catalina.connector.Connector connector ;
```

Tomcat connector.

### 3.4. embedded

```
protected static org.apache.catalina.startup.Embedded embedded ;
```

Embedded Tomcat service (like Catalina).

### 3.5. engine

```
protected static org.apache.catalina.Engine engine ;
```

Tomcat engine.

### 3.6. host

```
protected org.apache.catalina.Host host ;
```

Base container host.

### 3.7. realm

```
protected org.apache.catalina.Realm realm ;
```

Tomcat realm.

### 3.8. valves

```
protected java.util.List<org.apache.catalina.Valve> valves ;
```

Valves

### 3.9. addContext(String, String)

```
public org.apache.catalina.Context addContext(String path,
                                              String docBase);
```

Add context for path and docbase to current host.

#### Parameters

path	Path
docBase	Document base
return	Catalina context (that is, web application)

### 3.10. formatPath(String, String)

```
protected static String formatPath(String absWebappsPath,
```

```
String contextDirName);
```

Quick-n-dirty directory formatting to support launching in windows, specifically from ant.

### 3.11. getBaseHost()

```
public org.apache.catalina.Host getBaseHost();
```

Get base host.

#### Parameters

<i>return</i>	Base host
---------------	-----------

### 3.12. getConnector()

```
public org.apache.catalina.connector.Connector getConnector();
```

Return connector.

#### Parameters

<i>return</i>	Connector
---------------	-----------

### 3.13. getEmbedded()

```
public org.apache.catalina.startup.Embedded getEmbedded();
```

Getter for embedded object.

#### Parameters

<i>return</i>	Embedded object
---------------	-----------------

### 3.14. getEngine()

```
public org.apache.catalina.Engine getEngine();
```

Return Tomcat engine.

#### Parameters

<i>return</i>	Tomcat engine
---------------	---------------

### 3.15. getHost()

```
public org.apache.catalina.Host getHost();
```

Get the host.

#### Parameters

<i>return</i>	host
---------------	------

### 3.16. getRealm()

```
public org.apache.catalina.Realm getRealm();
```

Getter for realm.

#### Parameters

<i>return</i>	Realm
---------------	-------

### 3.17. init()

```
public void init();
```

**Specified by:** Method init in interface LoaderMBean

Initialization.

### 3.18. removeContext(String)

```
public void removeContext(String path);
```

**Specified by:** Method removeContext in interface LoaderMBean

Remove context from the current host.

#### Parameters

path	Path
------	------

### 3.19. setBaseHost(Host)

```
public void setBaseHost(org.apache.catalina.Host baseHost);
```

Set base host.

#### Parameters

baseHost	Base host
----------	-----------

### 3.20. setConnectionProperties(Map<String, String>)

```
public void setConnectionProperties(java.util.Map<java.lang.String, java.lang.String> props);
```

Set connection properties for the connector

#### Parameters

mappings	
----------	--

### 3.21. setConnector(Connector)

<pre>public void setConnector(org.apache.catalina.connector.Connector connector);</pre>
---

Set connector.

Parameters	
------------	--

connector	Connector
-----------	-----------

### 3.22. setConnectors(List<Connector>)

<pre>public void setConnectors(java.util.List&lt;org.apache.catalina.connector.Connector&gt; connectors);</pre>
---

Set additional connectors.

Parameters	
------------	--

connectors	Additional connectors
------------	-----------------------

### 3.23. setContexts(Map<String, String>)

<pre>public void setContexts(java.util.Map&lt;java.lang.String, java.lang.String&gt; contexts);</pre>
---

Set additional contexts.

Parameters	
------------	--

contexts	Map of contexts
----------	-----------------

### 3.24. setEmbedded(Embedded)

<pre>public void setEmbedded(org.apache.catalina.startup.Embedded embedded);</pre>
--

Setter for embedded object.

Parameters	
------------	--

embedded	Embedded object
----------	-----------------

### 3.25. setHost(Host)

<pre>public void setHost(org.apache.catalina.Host host);</pre>
--

Set the host.

Parameters	
------------	--

host	
------	--

### 3.26. setHosts(List<Host>)

```
public void setHosts(java.util.List<org.apache.catalina.Host> hosts);
```

Set additional hosts.

#### Parameters

hosts	List of hosts added to engine
-------	-------------------------------

### 3.27. setRealm(Realm)

```
public void setRealm(org.apache.catalina.Realm realm);
```

Setter for realm.

#### Parameters

realm	Realm
-------	-------

### 3.28. setValves(List<Valve>)

```
public void setValves(java.util.List<org.apache.catalina.Valve> valves);
```

Set additional valves.

#### Parameters

valves	List of valves
--------	----------------

### 3.29. shutdown()

```
public void shutdown();
```

**Specified by:** Method shutdown in interface LoaderMBean

Shut server down.

## 4. Class TomcatLoader.DirectoryFilter

Filters directory content

### 4.1. Synopsis

```
protected class TomcatLoader.DirectoryFilter implements java.io.FilenameFilter {
// Protected Constructors

    protected TomcatLoader.DirectoryFilter();

// Public Methods

    public boolean accept(java.io.File dir,
                         String name);

}
```

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait

## 4.2. accept(File, String)

```
public boolean accept(java.io.File dir,
                      String name);
```

**Specified by:** Method accept in interface `FilenameFilter`

Check whether file matches filter rules

Parameters	
dir	Directory
name	File name
<i>return</i>	true If file does match filter rules, false otherwise

## 5. Class TomcatVHostLoader

Red5 loader for Tomcat virtual hosts.

### 5.1. Synopsis

```
public class TomcatVHostLoader extends, org.?red5.?server.?tomcat.?TomcatLoader
    implements, org.?red5.?server.?tomcat.?TomcatVHostLoaderMBean {
// Protected Fields

    protected boolean autoDeploy ;

    protected String domain ;

    protected boolean liveDeploy ;

    protected String name ;

    protected boolean startChildren ;

    protected boolean unpackWARS ;

    protected String webappRoot ;

// Public Constructors

    public TomcatVHostLoader();

// Public Methods

    public void addAlias(String alias);

    public void addValve(org.apache.catalina.Valve valve);

    public org.apache.catalina.Host createHost();
```

```

public boolean getAutoDeploy();

public String getDefaultApplicationContextId();

public String getDomain();

public org.apache.catalina.Host getHost();

public boolean getLiveDeploy();

public String getName();

public boolean getStartChildren();

public boolean getUnpackWARs();

public String getWebappRoot();

public void init();

public void registerJMX();

public void removeAlias(String alias);

public void removeValve(String valveInfo);

public void setAutoDeploy(boolean autoDeploy);

public void setContexts(java.util.Map<java.lang.String, java.lang.String> contexts);

public void setDefaultApplicationContextId(String defaultApplicationContextId);

public void setDomain(String domain);

public void setLiveDeploy(boolean liveDeploy);

public void setName(String name);

public void setStartChildren(boolean startChildren);

public void setUnpackWARs(boolean unpackWARs);

public void setWebappRoot(String webappRoot);

public boolean startWebApplication(String applicationName);

public void uninit();

public void unregisterJMX();

}

```

**Methods inherited from org.red5.server.tomcat.TomcatLoader:** addContext , formatPath , getBaseHost , getConnector , getEmbedded , getEngine , getHost , getRealm , init , registerJMX , removeContext , setBaseHost ,

```
setConnectionProperties , setConnector , setConnectors , setContexts , setEmbedded ,
setHost , setHosts , setRealm , setValves , shutdown
```

**Methods inherited from org.red5.server.LoaderBase:** getApplicationContext ,  
getApplicationLoader , getRed5ApplicationContext , removeRed5ApplicationContext  
, setApplicationContext , setApplicationLoader , setRed5ApplicationContext ,  
setWebappFolder

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode  
, notify , notifyAll , toString , wait

**Fields inherited from org.red5.server.tomcat.TomcatLoader:** connectionProperties  
, connector , defaultParentContextKey , defaultSpringConfigLocation , embedded ,  
engine , host , realm , valves

**Fields inherited from org.red5.server.LoaderBase:** applicationContext , loader ,  
red5AppCtx , webappFolder

## 5.2. webappRoot

```
protected String webappRoot ;
```

Base web applications directory

## 5.3. addAlias(String)

```
public void addAlias(String alias);
```

**Specified by:** Method addAlias in interface TomcatVHostLoaderMBean

Adds an alias to the current host.

Parameters

alias
-------

## 5.4. addValve(Valve)

```
public void addValve(org.apache.catalina.Valve valve);
```

**Specified by:** Method addValve in interface TomcatVHostLoaderMBean

Adds a valve to the current host.

Parameters

valve
-------

## 5.5. createHost()

```
public org.apache.catalina.Host createHost();
```

Create a standard host.

#### Parameters

<i>return</i>
---------------

## 5.6. getHost()

public org.apache.catalina.Host getHost();
--

**Specified by:** Method getHost in interface TomcatVHostLoaderMBean

Returns the current host.

#### Parameters

<i>return</i>
---------------

## 5.7. init()

public void init();
---------------------

**Specified by:** Method init in interface TomcatVHostLoaderMBean

Initialization.

## 5.8. removeAlias(String)

public void removeAlias(String alias);
--

**Specified by:** Method removeAlias in interface TomcatVHostLoaderMBean

Removes an alias from the current host.

#### Parameters

alias
-------

## 5.9. removeValve(String)

public void removeValve(String valveInfo);
--

**Specified by:** Method removeValve in interface TomcatVHostLoaderMBean

Removes a valve from the current host.

#### Parameters

valveInfo
-----------

## 5.10. setContexts(Map<String, String>)

public void setContexts(java.util.Map<java.lang.String, java.lang.String> contexts);
--

Set additional contexts.

#### Parameters

contexts	Map of contexts
----------	-----------------

## 5.11. startWebApplication(String)

```
public boolean startWebApplication(String applicationName);
```

**Specified by:** Method startWebApplication in interface TomcatVHostLoaderMBean

Starts a web application and its red5 (spring) component. This is basically a stripped down version of init().

#### Parameters

return
--------

## 5.12. uninit()

```
public void uninit();
```

**Specified by:** Method uninit in interface TomcatVHostLoaderMBean

Un-initialization.

# 6. Interface TomcatVHostLoaderMBean

Simple mbean interface for Tomcat container virtual host loaders.

## 6.1. Synopsis

```
public interface TomcatVHostLoaderMBean {
// Public Methods

    public void addAlias(String alias);

    public org.apache.catalina.Context addContext(String path,
                                                String docBase);

    public void addValve(org.apache.catalina.Valve valve);

    public boolean getAutoDeploy();

    public String getDomain();

    public org.apache.catalina.Host getHost();

    public boolean getLiveDeploy();

    public String getName();

    public boolean getStartChildren();
```

```

    public boolean getUnpackWARs();

    public String getWebappRoot();

    public void init();

    public void registerJMX();

    public void removeAlias(String alias);

    public void removeContext(String path);

    public void removeValve(String valveInfo);

    public void setAutoDeploy(boolean autoDeploy);

    public void setDomain(String domain);

    public void setLiveDeploy(boolean liveDeploy);

    public void setName(String name);

    public void setStartChildren(boolean startChildren);

    public void setUnpackWARs(boolean unpackWARs);

    public void setWebappRoot(String webappRoot);

    public void shutdown();

    public boolean startWebApplication(String applicationName);

    public void uninit();

    public void unregisterJMX();
}

```

## 7. Class WebappClassLoader

This class loader provides a means to by-pass the class loader used to launch an application as the system class loader.

### 7.1. Synopsis

```

public class WebappClassLoader extends org.apache.catalina.loader.WebappClassLoader {
// Public Constructors

    public WebappClassLoader();

    public WebappClassLoader(ClassLoader parent);

}

```

**Methods inherited from org.apache.catalina.loader.WebappClassLoader:**

```
addLifecycleListener , addPermission , addRepository , addURL , clearReferences ,
closeJARs , deleteDir , filter , findClass , findClassInternal , findLifecycleListeners ,
findLoadedClass0 , findLoadedResource , findRepositories , findResource ,
findResourceInternal , findResources , getAntiJARLocking , getDelegate , getJarPath ,
getPermissions , getResource , getResourceAsStream , getResources , getURI , getURL ,
getURLs , isPackageSealed , loadClass , loadedByThisOrChild , modified , nullInstance ,
openJARs , refreshPolicy , removeLifecycleListener , setAntiJARLocking , setDelegate ,
setJarPath , setParentClassLoader , setResources , setWorkDir , start , stop , toString ,
validate , validateJarFile
```

**Methods inherited from java.net.URLClassLoader:** definePackage , newInstance

**Methods inherited from java.security.SecureClassLoader:** defineClass

**Methods inherited from java.lang.ClassLoader:** clearAssertionStatus ,
findLibrary , findLoadedClass , findSystemClass , getPackage , getPackages ,
getParent , getSystemClassLoader , getSystemResource , getSystemResourceAsStream ,
getSystemResources , resolveClass , setClassAssertionStatus , setDefaultAssertionStatus ,
setPackageAssertionStatus , setSigners

**Methods inherited from java.lang.Object:** clone , equals , finalize , getClass , hashCode ,
notify , notifyAll , wait

**Fields inherited from org.apache.catalina.loader.WebappClassLoader:** allPermission ,
delegate , ENABLE\_CLEAR\_REFERENCES , files , hasExternalRepositories , jarFiles ,
jarNames , jarPath , jarRealFiles , lastJarAccessed , lastModifiedDates , loaderDir ,
loaderPC , log , needConvert , notFoundResources , packageTriggers , parent , paths ,
permissionList , repositories , repositoryURLs , resourceEntries , resources ,
securityManager , sm , started , system , triggers

## 7.2. WebappClassLoader()

```
public WebappClassLoader();
```

Construct a new ClassLoader with no defined repositories and no parent ClassLoader

## 7.3. WebappClassLoader(ClassLoader)

```
public WebappClassLoader(ClassLoader parent);
```

Construct a new ClassLoader with no defined repositories

# 1. Class WarLoaderServlet

Entry point from which the server config file is loaded while running within a J2EE application container. This listener should be registered after Log4jConfigListener in web.xml, if the latter is used.

## 1.1. Synopsis

```
public class WarLoaderServlet extends org.springframework.web.context.ContextLoaderListener {  
    // Public Static Fields  
  
    public static org.slf4j.Logger logger;  
  
    // Public Constructors  
  
    public WarLoaderServlet();  
  
    // Public Methods  
  
    public void contextDestroyed(javax.servlet.ServletContextEvent sce);  
  
    public void contextInitialized(javax.servlet.ServletContextEvent sce);  
  
    public org.springframework.web.context.ContextLoader getContextLoader();  
  
    public void registerSubContext(String webAppKey);  
}
```

**Methods inherited from org.springframework.web.context.ContextLoaderListener:**

contextDestroyed, contextInitialized, createContextLoader, getContextLoader

**Methods inherited from java.lang.Object:** clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait

## 1.2. contextDestroyed(ServletContextEvent)

```
public void contextDestroyed(javax.servlet.ServletContextEvent sce);
```

Clearing the in-memory configuration parameters, we will receive notification that the servlet context is about to be shut down

## 1.3. contextInitialized(ServletContextEvent)

```
public void contextInitialized(javax.servlet.ServletContextEvent sce);
```

Main entry point for the Red5 Server as a war

# Appendix A. Constant field values

## 1. Package org.red5.server.api.remoting.\*

Table A.1.

APPEND_TO_GATEWAY_URL	"AppendToGatewayUrl"
CREDENTIALS	"Credentials"
DEBUG_SERVER	"amf_server_debug"
PERSISTENT_HEADER	"RequestPersistentHeader"
REPLACE_GATEWAY_URL	"ReplaceGatewayUrl"

## 2. Package org.red5.server.api.scheduling.\*

Table A.2.

BEAN_NAME	"schedulingService"
-----------	---------------------

## 3. Package org.red5.server.api.so.\*

Table A.3.

BEAN_NAME	"sharedObjectService"
-----------	-----------------------

## 4. Package org.red5.server.net.proxy.\*

Table A.4.

FORWARD_KEY	"proxy_forward_key"
-------------	---------------------

## 5. Package org.red5.server.net.udp.\*

Table A.5.

PORT	5150
------	------

# Index

## Symbols

\_context, 594

## A

AbstractClientStream, 1000  
AbstractMessage, 10, 10, 601  
AbstractPipe, 606  
AbstractScopeAdapter, 347  
AbstractStream, 1002  
AbstractStream.State, 1004  
accept, 1110  
AccessDeniedException, 583  
AcknowledgeMessage, 11  
acquire, 488, 969, 986  
acquireCount, 967  
acquireToken, 1035  
acquireTokenBestEffort, 1035  
acquireTokenNonblocking, 1036  
add, 1078  
addAlias, 1112  
addChildScope, 281, 307, 348, 419, 425, 437  
addClient, 225, 265  
addContext, 1105  
addData, 556, 1084, 1086  
addEvent, 961, 961, 980, 980  
addEventListener, 254, 454, 986  
addHeader, 464, 464, 673, 679, 679  
addItem, 526, 526, 1050, 1050, 1061, 1061  
addListener, 335, 335, 361, 429, 429  
addMapping, 335, 430  
addParameters, 4, 5, 10, 12, 13, 17  
addPipeConnectionListener, 608, 623, 1007  
addScheduledJob, 361, 469, 918  
addScheduledJobAfterDelay, 361, 469, 919  
addScheduledOnceJob, 362, 362, 469, 470, 919, 919  
addSharedObjectListener, 489, 953, 986  
addStreamListener, 516, 1012  
addTask, 632  
addValve, 1112  
allocate, 286  
AllocationDebugger, 793  
AMF, 39  
AMF3, 57  
AMFGatewayServlet, 892  
AMFTunnelServlet, 895

analyzeKeyFrames, 115, 122, 169  
Annotations  
    Anonymous, 2  
    DeclarePrivate, 2  
    DeclareProtected, 2  
    DontSerialize, 2  
    Anonymous, 2  
APP\_GC, 853  
APP\_RESOURCE\_LOWMEMORY, 853  
APP\_SCRIPT\_ERROR, 853  
APP\_SCRIPT\_WARNING, 853  
APP\_SHUTDOWN, 853  
appConnect, 353, 362  
appContext, 343  
appCtx, 707, 746, 882  
appDisconnect, 354, 363  
APPEND\_TO\_GATEWAY\_URL, 465  
appJoin, 354  
appLeave, 354, 363  
APPLICATION\_AMF, 893, 898  
ApplicationAdapter, 350  
applicationContext, 277, 291, 335, 879  
ApplicationMBean, 352  
appLoader, 343  
appStart, 354, 363  
appStop, 354, 364  
arguments, 932  
ArrayList, 6  
associate, 643, 650  
asStatus, 859  
AsyncMessage, 11  
attributeNames, 93  
attributes, 235  
AttributeStore, 234, 235, 235, 235  
AttributeStoreMBean, 386  
AUDIO\_ADPCM, 806  
AUDIO\_MP3, 806  
AUDIO\_NELLYMOOSER, 806  
AUDIO\_NELLYMOOSER\_8KHZ, 806  
AUDIO\_UNCOMPRESSED, 807  
AudioData, 794, 795  
audioOnly, 115  
available, 1037, 1043

## B

BaseConnection, 240, 243  
BaseEvent, 795, 797, 797  
BaseInput, 173  
BaseInput.ReferenceMode, 174

BaseMRTMPConnection, 639  
BaseOutput, 175, 176  
BaseProtocolEncoder, 655  
BaseRTMPClientHandler, 698  
BaseRTMPHandler, 705  
BaseRTMPTConnection, 867  
BaseStreamableFileService, 18  
BasicHandler, 904  
BasicHandler.TimeoutTask, 905  
BasicScope, 252, 253  
BasicScope.EmptyBasicScopelterator, 256  
basicScopes, 243  
BEAN\_NAME, 428, 497, 541, 548  
beginUpdate, 490, 490, 953, 953, 969, 969, 986, 986  
Bootstrap, 257  
BroadcastScope, 1005, 1006  
buf, 52, 52  
buffer, 869  
bufferDecoding, 658  
BufferType, 19  
BufferUtils, 201  
ByteArray, 60, 63, 63  
byteArrayToBinaryString, 206  
byteArrayToHexString, 206, 207  
ByteBufferUtil, 202  
bytesAvailable, 63  
BytesRead, 799, 800

**C**

CacheableImpl, 572  
cachedStatusObjects, 862  
CacheImpl, 569  
cacheStatusObjects, 863  
CachingFileKeyFrameMetaCache, 19  
call, 816  
Call, 930, 931, 931, 932  
calls, 696  
cancelGhostConnectionsCleanup, 364  
canContinueDecoding, 658  
canDropFrames, 557, 1084, 1086  
canHandle, 18, 24  
canHandleData, 557, 1085, 1086  
canSendPacket, 1027, 1082  
canStartDecoding, 658  
CaptureViewerServlet, 896  
changeStats, 967  
Channel, 710, 711  
CHARSET, 40, 58, 213

checkRelease, 970  
checkRemoveEmptyDirectories, 907  
chunkBuffer, 764  
ChunkSize, 801, 802  
Classes  
    AbstractClientStream, 1000  
    AbstractMessage, 10, 601  
    AbstractPipe, 606  
    AbstractScopeAdapter, 347  
    AbstractStream, 1002  
    AbstractStream.State, 1004  
    AcknowledgeMessage, 11  
    AllocationDebugger, 793  
    AMF, 39  
    AMF3, 57  
    AMFGatewayServlet, 892  
    AMFTunnelServlet, 895  
    ApplicationAdapter, 350  
    ArrayList, 6  
    AsyncMessage, 11  
    AttributeStore, 234  
    AudioData, 794  
    BaseConnection, 240  
    BaseEvent, 795  
    BaseInput, 173  
    BaseInput.ReferenceMode, 174  
    BaseMRTMPConnection, 639  
    BaseOutput, 175  
    BaseProtocolEncoder, 655  
    BaseRTMPClientHandler, 698  
    BaseRTMPHandler, 705  
    BaseRTMPTConnection, 867  
    BaseStreamableFileService, 18  
    BasicHandler, 904  
    BasicHandler.TimeoutTask, 905  
    BasicScope, 252  
    BasicScope.EmptyBasicScopelterator, 256  
    Bootstrap, 257  
    BroadcastScope, 1005  
    BufferType, 19  
    BufferUtils, 201  
    ByteArray, 60  
    ByteBufferUtil, 202  
    BytesRead, 799  
    CacheableImpl, 572  
    CacheImpl, 569  
    CachingFileKeyFrameMetaCache, 20  
    Call, 930  
    CaptureViewerServlet, 896

Channel, 710  
ChunkSize, 801  
Client, 257  
ClientBroadcastStream, 1010  
ClientBW, 803  
ClientList, 263  
ClientManager, 225  
ClientRegistry, 264  
ClientSharedObject, 951  
CommandMessage, 12  
ConnectionConsumer, 1089  
ConnectionProvider, 1098  
Constants, 13  
ConsumerService, 1016  
Context, 268  
ContextLoader, 276  
ContextLoggingListener, 219  
ContextServiceResolver, 935  
ConversionUtils, 936  
CoreHandler, 280  
DataInput, 69  
DataMessage, 4  
DataOutput, 73  
DataTypes, 177  
DebugPooledByteBufferAllocator, 284  
DebugProxyHandler, 662  
DefaultStreamFilenameGenerator, 1017  
DeferredResult, 712  
DenyAllStreamAccess, 560  
Deserializer, 181  
DOM2Writer, 202  
DummyBWControlService, 1018  
EchoService, 227  
EchoService.SampleObject, 231  
EdgeMRTMPHandler, 640  
EdgeRTMPHandler, 714  
EdgeRTMPMinConnection, 715  
EdgeRTMPMinaloHandler, 716  
EhCacheImpl, 573  
ErrorMessage, 15  
ExecutorFilter, 100  
ExecutorFilter.Event, 103  
ExecutorFilter.EventType, 104  
FileConsumer, 1090  
FileKeyFrameMetaCache, 20  
FilePersistence, 906  
FilePersistenceThread, 908  
FileProvider, 1098  
FileStreamSource, 1021  
Flag, 182  
FlexMessage, 809  
FlexMessagingService, 667  
FlexSharedObjectMessage, 957  
FlexStreamSend, 810  
FLV, 117  
FLVData, 805  
FLVHeader, 106  
FLVReader, 120  
FLVService, 125  
FLVWriter, 127  
GlobalScope, 288  
GroovyScriptFactory, 922  
HandlerServiceResolver, 940  
Header, 835  
HexCharset, 203  
HexCharset.Decoder, 204  
HexCharset.Encoder, 204  
HexDump, 205  
IConnection.Encoding, 411  
IEvent.Type, 452  
IKeyFrameDataAnalyzer.KeyFrameMeta, 115  
InMemoryPullPullPipe, 625  
InMemoryPushPushPipe, 626  
Input, 42, 86, 155  
Input.ClassReference, 92  
Input.PendingObject, 93  
Input.RefStorage, 93  
Invoke, 812  
IOUtils, 213  
IProviderService.INPUT\_TYPE, 1031  
ISharedObjectEvent.Type, 959  
IStreamFilenameGenerator.GenerationType, 542  
JettyApplicationContext, 590  
JettyApplicationLoader, 590  
JettyLoader, 592  
JMXAgent, 596  
JMXFactory, 598  
JMXUtil, 599  
JythonScriptFactory, 925  
LoaderBase, 290  
LoggerContextFilter, 219  
LoggingContextSelector, 220  
MappingStrategy, 294  
MetaCue, 144  
MetaData, 146  
MetaService, 150  
MidiPlayer, 635

Mock, 158  
MP3, 165  
MP3Header, 166  
MP3Reader, 168  
MP3Service, 171  
MRTMPClient, 644  
MRTMPCodecFactory, 652  
MRTMPEdgeConnection, 644  
MRTMPMinaTransport, 644  
MRTMPOriginConnection, 645  
MRTMPPacket, 646  
MRTMPPacket.Body, 646  
MRTMPPacket.Header, 647  
MRTMPPacket.RTMPBody, 647  
MRTMPPacket.RTMPHeader, 648  
MRTMPProtocolDecoder, 652  
MRTMPProtocolEncoder, 653  
MulticastEventProcessor, 764  
MultiThreadedApplicationAdapter, 356  
NetworkDumpFilter, 663  
NoCacheImpl, 578  
Notify, 814  
ObjectMap, 215  
ObjectProxy, 7  
OOBControlMessage, 628  
OriginMRTMPHandler, 648  
Output, 50, 94, 159  
OutputStream, 1038  
Packet, 838  
PendingCall, 943  
PersistableAttributeStore, 295  
PersistenceUtils, 462  
Ping, 818  
PipeConnectionEvent, 630  
PipeUtils, 633  
PlayBuffer, 1040  
PlayEngine, 1042  
PlayEngine.Builder, 1047  
PlaylistSubscriberStream, 1048  
ProtocolState, 657  
ProviderService, 1055  
ProxyFilter, 665  
QuartzSchedulingService, 918  
QuartzSchedulingServiceJob, 920  
RamPersistence, 910  
RandomGUID, 216  
RecordSet, 191  
RecordSetPage, 194  
Red5, 432  
Red5WebPropertiesConfiguration, 593  
RedirectHTTPSServlet, 897  
RemotingCall, 693  
RemotingClient, 671  
RemotingClient.RemotingWorker, 675  
RemotingCodecFactory, 687  
RemotingConnection, 675  
RemotingHeader, 684  
RemotingMessage, 16  
RemotingPacket, 695  
RemotingProtocolDecoder, 688  
RemotingProtocolEncoder, 690  
RequestDumpServlet, 898  
ResetMessage, 1096  
Resolver, 153  
RhinoScriptFactory, 927  
RhinoScriptUtils, 929  
RTMP, 765  
RTMPClient, 719  
RTMPClientConnManager, 720  
RTMPCodecFactory, 773  
RTMPConnection, 721  
RTMPConnManager, 720  
RTMPHandler, 738  
RTMPHandshake, 740  
RTMPMessage, 1095  
RTMPMinaCodecFactory, 775  
RTMPMinaConnection, 741  
RTMPMinaloHandler, 745  
RTMPMinaProtocolDecoder, 777  
RTMPMinaProtocolEncoder, 778  
RTMPMinaTransport, 749  
RTMPOriginConnection, 751  
RTMPProtocolDecoder, 779  
RTMPProtocolEncoder, 786  
RTMPTClient, 871  
RTMPTClientConnection, 872  
RTMPTCodecFactory, 889  
RTMPTConnection, 873  
RTMPTHandler, 877  
RTMPTLoader, 878  
RTMPTProtocolDecoder, 890  
RTMPTProtocolEncoder, 891  
RTMPTServlet, 880  
RTMPUtils, 753  
RuntimeStatusObject, 843  
Scope, 302  
ScopeResolver, 331  
ScopeServiceResolver, 945

ScopeUtils, 441  
ScreenVideo, 1084  
SequencedMessage, 4  
Serializer, 195  
SerializeUtils, 823  
Server, 333  
ServerBW, 823  
ServerStream, 1058  
ServiceInvoker, 946  
ServiceUtils, 480, 948  
ServletUtils, 899  
SharedMidiObject, 635  
SharedMidiObject.MidiReceiver, 636  
SharedObject, 962  
SharedObjectEvent, 977  
SharedObjectMessage, 978  
SharedObjectScope, 982  
SharedObjectService, 991  
SharedObjectTypeMapping, 841  
Shutdown, 339  
SimpleBandwidthConfigure, 560  
SimpleBWControlService, 1065  
SimpleBWControlService.BWContext, 1069  
SimpleBWControlService.TokenRequest, 1069  
SimpleBWControlService.TokenRequestContext, 1070  
SimpleBWControlService.TokenRequestType, 1070  
SimpleClient, 223  
SimpleConnectionBWConfig, 562  
SimpleMRTMPEdgeManager, 649  
SimpleMRTMPOriginManager, 650  
SimplePlayItem, 563  
SimplePlaylistController, 1070  
SocketPolicyHandler, 654  
SorensonVideo, 1085  
Standalone, 340, 905  
StatefulScopeWrappingAdapter, 379  
StatisticsCounter, 514  
StatisticsService, 994  
StatisticsServlet, 901  
Status, 845  
StatusMessage, 1096  
StatusObject, 858  
StatusObjectService, 862  
StreamableFileFactory, 37  
StreamBandwidthController, 1093  
StreamCodecInfo, 1087  
StreamingProxy, 1078  
StreamService, 1072  
StreamTracker, 1077  
StreamUtils, 566  
Tag, 130  
Test, 636  
Test.MyReceiver, 637  
ThreadPool, 914  
TomcatApplicationContext, 1101  
TomcatApplicationLoader, 1102  
TomcatLoader, 1103  
TomcatLoader.DirectoryFilter, 1109  
TomcatRTMPSLoader, 865  
TomcatRTMPTLoader, 886  
TomcatVHostLoader, 1110  
Unknown, 825  
VideoCodecFactory, 1080  
VideoData, 827  
VideoData.FrameType, 829  
VideoFrameDropper, 1081  
W3CAppender, 221  
WarLoaderServlet, 1117  
WebappClassLoader, 1115  
WebScope, 341  
Worker, 914  
XmlRpcScopeStatistics, 996  
XMLUtils, 216  
ZAMFGatewayServlet, 902  
ZAMFGatewayServlet.Handler, 902  
className, 93  
clear, 490, 953, 961, 970, 980, 986, 1040  
clearReferences, 174, 176, 184, 188  
clearSharedObjects, 364, 499, 991  
client, 243  
Client, 257, 259  
CLIENT\_BUFFER, 820  
ClientBroadcastStream, 1010  
ClientBroadcastStreamMBean, 1016  
ClientBW, 803  
clientCallback, 694  
ClientDetailsException, 583, 584, 584  
ClientExceptionHandler, 712  
clientid, 844, 847  
ClientList, 263  
ClientManager, 225, 225  
ClientMBean, 264, 386  
ClientNotFoundException, 584, 585  
ClientRegistry, 264  
ClientRegistryMBean, 267, 387

ClientRejectedException, 585, 586, 586  
ClientSharedObject, 951, 953  
clientStats, 306  
clone, 836  
close, 28, 31, 123, 128, 169, 245, 405, 490, 532, 636, 637, 679, 711, 727, 743, 870, 953, 970, 986, 1012, 1022, 1033, 1039, 1044, 1050, 1061, 1093  
closeChannel, 727  
CLOSED, 105  
closed, 243  
closeStream, 550, 1073, 1073  
closing, 869  
code, 847  
codecFactory, 747, 877, 893  
CommandMessage, 12  
compareTo, 145  
compress, 63  
configureClassLoader, 594  
configureDefaults, 594  
configureWebApp, 594  
conn, 433  
connect, 245, 245, 281, 282, 307, 307, 348, 351, 365, 405, 406, 419, 420, 425, 437, 437, 487, 634, 639, 641, 679, 679, 701, 701, 701, 702, 702, 953  
connectArguments, 700  
connectCallback, 700  
connectionClosed, 223, 707, 718  
ConnectionConsumer, 1089, 1089  
ConnectionMBean, 387  
connectionOpened, 223, 702, 707, 718  
connectionParams, 700  
connectionProperties, 1104  
ConnectionProvider, 1098  
connectionStats, 306  
connector, 1105  
connToScope, 259  
Constants, 13, 830  
CONSUMER\_CONNECT\_PULL, 631  
CONSUMER\_CONNECT\_PUSH, 631  
CONSUMER\_DISCONNECT, 631  
consumers, 607  
ConsumerService, 1016  
containsKey, 8  
Context, 268, 269, 269  
context, 887  
contextDestroyed, 1117  
contextInitialized, 1117  
ContextLoader, 276  
ContextLoaderMBean, 279  
ContextLoggingListener, 219  
contextMap, 277  
ContextMBean, 279  
contextPath, 343  
contextsConfig, 277  
ContextServiceResolver, 935  
continueDecoding, 658  
controller, 1060  
ConversionUtils, 936  
convert, 937  
convertArrayToList, 937  
convertArrayToSet, 938  
convertBeanToMap, 938  
convertMapToBean, 938  
convertNumberToWrapper, 938  
convertParams, 939  
convertStringToWrapper, 939  
convertToArray, 939  
convertToWrappedPrimitive, 940  
copy, 899, 900  
copyThenClose, 900  
CORE\_ARRAY, 178  
CORE\_BOOLEAN, 178  
CORE\_BYTEARRAY, 178  
CORE\_DATE, 178  
CORE\_MAP, 178  
CORE\_NULL, 178  
CORE\_NUMBER, 178  
CORE\_OBJECT, 179  
CORE\_SKIP, 179  
CORE\_STRING, 179  
CORE\_XML, 179  
CoreHandler, 280  
CoreHandlerMBean, 283  
correlationId, 12  
correlationID, 602  
create, 793  
createChildScope, 308, 323, 380, 420, 437  
createHost, 1112  
createOutputStream, 727  
createRhinoObject, 929  
createServerStream, 567  
createSharedObject, 365, 499, 992  
createStream, 550, 1074  
createStreamName, 727  
createTokenBucket, 1038  
creationTime, 259

CREDENTIALS, 466  
currentItem, 1060  
CUSTOM\_AMF\_MASK, 179  
CUSTOM\_JSON\_MASK, 179  
CUSTOM\_MOCK\_MASK, 179  
CUSTOM\_RTMP\_MASK, 179  
CUSTOM\_XML\_MASK, 179

**D**

data, 63, 685, 696, 816, 826, 828, 840  
DATA\_OPERATION\_SET, 14  
DATA\_OPERATION\_UPDATE, 14  
DATA\_OPERATION\_UPDATE\_ATTRIBUTES, 14  
dataInput, 63  
DataInput, 69, 70  
DataMessage, 4  
dataOffset, 107  
dataOutput, 63  
DataOutput, 73, 74  
dataType, 826  
DataTypes, 177  
debug, 214  
DEBUG\_SERVER, 466  
DebugPooledByteBufferAllocator, 284, 285, 285, 286, 286  
DebugProxyHandler, 662  
DeclarePrivate, 2  
DeclareProtected, 2  
decode, 660, 689, 778, 782, 870  
decodeAudioData, 758, 782  
decodeBuffer, 660, 689, 782  
decodeBytesRead, 758, 782  
decodeCalls, 690  
decodeChannelId, 754  
decodeChunkSize, 759, 782  
decodeFlexMessage, 759, 782  
decodeFlexSharedObject, 759, 783  
decodeHandshake, 783  
decodeHeader, 28, 123, 169, 783  
decodeHeaderSize, 754  
decodeInvoke, 759, 783  
decodeMessage, 784  
decodeNotify, 760, 784  
decodeNotifyOrInvoke, 784  
decodePacket, 784  
decodePing, 760, 785  
decoder, 687, 774, 776, 869, 889  
DECODER\_BUFFER, 657  
DECODER\_CONTINUE, 658  
DECODER\_OK, 658  
decodeRequest, 893  
decodeSharedObject, 760, 785  
decodeUnknown, 760, 785  
decodeVideoData, 761, 785  
deconfigureWebApp, 595  
decrement, 514  
DEFAULT\_BUFFER\_SIZE, 899  
DEFAULT\_CHUNK\_SIZE, 767  
DEFAULT\_HOST, 332  
DEFAULT\_TIMEOUT, 672  
defaultController, 1060  
DefaultStreamFilenameGenerator, 1017  
defaultWebConfig, 592  
DeferredResult, 712  
deferredResults, 725  
deleteStats, 967  
deleteStream, 550, 550, 1074, 1074  
deleteStreamById, 537, 728  
DenyAllStreamAccess, 560  
dequeue, 1022, 1034  
description, 847  
deserialize, 181, 298, 458, 970  
Deserializer, 181  
deserializer, 687, 774, 889  
destroy, 308, 323, 446, 570, 575, 579, 882  
details, 844, 848  
disconnect, 259, 282, 308, 348, 351, 365, 400, 420, 425, 438, 487, 634, 639, 641, 703, 954  
dispatchEvent, 246, 254, 308, 453, 680, 954, 986, 1012  
dispose, 287, 690, 691, 778, 779  
disposeCached, 765  
dissociate, 643, 650  
docToString, 217  
docToString1, 217  
docToString2, 217  
doDecodeSharedObject, 785  
doEncodeSharedObject, 788  
DOM2Writer, 202  
DontSerialize, 2  
doPost, 902  
doRelease, 800, 802, 816  
dropPacket, 1027, 1082  
droppedMessages, 243  
DummyBWControlService, 1018  
dump, 793  
duration, 116

**E**

echoArray, 227, 232  
 echoBoolean, 228, 232  
 echoDate, 228, 232  
 echoList, 228, 232  
 echoNumber, 229, 233  
 echoObject, 229, 233  
 EchoService, 227  
 EchoService.SampleObject, 231  
 echoString, 229, 233  
 echoXML, 229, 233  
 EdgeMRTMPHandler, 640  
 EdgeRTMPHandler, 714  
 EdgeRTMPMinaConnection, 715  
 EdgeRTMPMinaloHandler, 716  
 EhCacheImpl, 573  
 Elements  
     permission, 2  
 embedded, 1105  
 EMPTY, 335  
 encode, 661, 692, 779, 788  
 encodeAudioData, 762, 788  
 encodeBytesRead, 762, 788  
 encodeChunkSize, 762, 789  
 encodeFlexMessage, 789  
 encodeFlexSharedObject, 762, 789  
 encodeHeader, 789, 789  
 encodeHeaderByte, 754  
 encodeInvoke, 762, 790  
 encodeMessage, 790  
 encodeNotify, 763, 790  
 encodeNotifyOrInvoke, 790, 791  
 encodePacket, 791  
 encodePing, 763, 791  
 encoder, 687, 774, 776, 869, 889  
 encodeSharedObject, 763, 791  
 encodeString, 52, 96  
 encodeUnknown, 763, 791  
 encodeVideoData, 764, 792  
 encoding, 725  
 endpoints, 667  
 endUpdate, 490, 954, 970, 986  
 enforceAMF3, 88, 96  
 engine, 1105  
 equals, 260, 802, 814, 816, 836  
 ERROR, 848  
 ErrorMessage, 15  
 errorReceived, 670

EVENT, 136  
 EXCEPTION, 105  
 exception, 932  
 exceptionCaught, 102, 662, 747  
 Exceptions  
     AccessDeniedException, 583  
     ClientDetailsException, 583  
     ClientNotFoundException, 584  
     ClientRejectedException, 585  
     HandshakeFailedException, 655  
     MethodNotFoundException, 942  
     NotAllowedException, 942  
     OperationNotSupportedException, 557  
     ProtocolException, 656  
     ResourceExistException, 558  
     ResourceNotFoundException, 558  
     ScopeHandlerNotFoundException, 586  
     ScopeNotFoundException, 587  
     ScopeShuttingDownException, 587  
     ServiceNotFoundException, 588, 947  
     SharedObjectException, 588  
     StreamControlException, 589  
     StreamDataException, 589  
     StreamNotFoundException, 1071  
 execute, 468, 915, 920  
 executeScript, 923  
 executeTask, 675  
 ExecutorFilter, 100, 101, 101, 101  
 ExecutorFilter.Event, 103  
 ExecutorFilter.EventType, 104  
 extraHeaders, 602

**F**

FCPublish, 366  
 FCUnpublish, 366  
 Fields  
     \_context, 594  
     acquireCount, 967  
     APP\_GC, 853  
     APP\_RESOURCE\_LOWMEMORY, 853  
     APP\_SCRIPT\_ERROR, 853  
     APP\_SCRIPT\_WARNING, 853  
     APP\_SHUTDOWN, 853  
     appContext, 343  
     appCtx, 707, 747, 882  
     APPEND\_TO\_GATEWAY\_URL, 465  
     APPLICATION\_AMF, 893, 898  
     applicationContext, 277, 291, 335, 879  
     appLoader, 343

arguments, 932  
attributeNames, 93  
attributes, 235  
AUDIO\_ADPCM, 806  
AUDIO\_MP3, 806  
AUDIO\_NELLYMOOSER, 806  
AUDIO\_NELLYMOOSER\_8KHZ, 806  
AUDIO\_UNCOMPRESSED, 807  
audioOnly, 115  
basicScopes, 243  
BEAN\_NAME, 428, 497, 541, 548  
buf, 52  
buffer, 869  
cachedStatusObjects, 862  
call, 816  
calls, 696  
changeStats, 967  
CHARSET, 40, 58, 213  
className, 93  
client, 243  
CLIENT\_BUFFER, 820  
clientCallback, 694  
clientid, 844, 847  
clientStats, 306  
CLOSED, 105  
closed, 243  
closing, 869  
code, 847  
codecFactory, 747, 877, 893  
conn, 433  
connectArguments, 700  
connectCallback, 700  
connectionParams, 700  
connectionProperties, 1104  
connectionStats, 306  
connector, 1105  
connToScope, 259  
CONSUMER\_CONNECT\_PULL, 631  
CONSUMER\_CONNECT\_PUSH, 631  
CONSUMER\_DISCONNECT, 631  
consumers, 608  
context, 887  
contextMap, 277  
contextPath, 343  
contextsConfig, 277  
controller, 1060  
CORE\_ARRAY, 178  
CORE\_BOOLEAN, 178  
CORE\_BYTEARRAY, 178  
CORE\_DATE, 178  
CORE\_MAP, 178  
CORE\_NULL, 178  
CORE\_NUMBER, 178  
CORE\_OBJECT, 179  
CORE\_SKIP, 179  
CORE\_STRING, 179  
CORE\_XML, 179  
correlationId, 12  
correlationID, 602  
creationTime, 259  
CREDENTIALS, 466  
currentItem, 1060  
CUSTOM\_AMF\_MASK, 179  
CUSTOM\_JSON\_MASK, 179  
CUSTOM\_MOCK\_MASK, 179  
CUSTOM\_RTMP\_MASK, 179  
CUSTOM\_XML\_MASK, 179  
data, 63, 685, 696, 816, 826, 828, 840  
DATA\_OPERATION\_SET, 14  
DATA\_OPERATION\_UPDATE, 14  
DATA\_OPERATION\_UPDATE\_ATTRIBUTES, 14  
dataInput, 63  
dataOffset, 107  
dataOutput, 63  
dataType, 826  
DEBUG\_SERVER, 466  
decoder, 687, 774, 776, 869, 889  
DECODER\_BUFFER, 657  
DECODER\_CONTINUE, 658  
DECODER\_OK, 658  
DEFAULT\_BUFFER\_SIZE, 899  
DEFAULT\_CHUNK\_SIZE, 767  
DEFAULT\_HOST, 332  
DEFAULT\_TIMEOUT, 672  
defaultController, 1060  
defaultWebConfig, 592  
deferredResults, 725  
deleteStats, 967  
description, 847  
deserializer, 687, 774, 889  
details, 844, 848  
droppedMessages, 243  
duration, 116  
embedded, 1105  
EMPTY, 335  
encoder, 687, 774, 776, 869, 889  
encoding, 725

endpoints, 667  
engine, 1105  
ERROR, 848  
EVENT, 136  
EXCEPTION, 105  
exception, 932  
extraHeaders, 602  
FLAG\_CODEC\_H263, 34  
FLAG\_CODEC\_SCREEN, 34  
FLAG\_CODEC\_VP6, 34  
FLAG\_FORMAT\_ADPCM, 34  
FLAG\_FORMAT\_MP3, 34  
FLAG\_FORMAT\_NELLYMOSER, 34  
FLAG\_FORMAT\_NELLYMOSER\_8\_KHZ, 34  
FLAG\_FORMAT\_RAW, 34  
FLAG\_FRAMETYPE\_DISPOSABLE, 34  
FLAG\_FRAMETYPE\_INTERFRAME, 35  
FLAG\_FRAMETYPE\_KEYFRAME, 35  
FLAG\_RATE\_11\_KHZ, 35  
FLAG\_RATE\_22\_KHZ, 35  
FLAG\_RATE\_44\_KHZ, 35  
FLAG\_RATE\_5\_5\_KHZ, 35  
FLAG\_SIZE\_16\_BIT, 35  
FLAG\_SIZE\_8\_BIT, 35  
FLAG\_TYPE\_MONO, 35  
FLAG\_TYPE\_STEREO, 36  
flagAudio, 107  
flagReserved01, 107  
flagReserved02, 107  
flagVideo, 107  
FORWARD\_KEY, 666  
FRAMETYPE\_DISPOSABLE, 807  
FRAMETYPE\_INTERFRAME, 807  
FRAMETYPE\_KEYFRAME, 807  
globals, 335  
globalScope, 332  
handler, 747, 869  
HANDLER\_ATTRIBUTE, 878  
HANDLER\_ERROR, 694  
HANDLER\_SUCCESS, 694  
HANDSHAKE\_SIZE, 831  
header, 797, 840  
HEADER\_CONTINUE, 831  
HEADER\_ENDPOINT, 14  
HEADER\_NEW, 832  
HEADER\_SAME\_SOURCE, 832  
HEADER\_TIMER\_CHANGE, 832  
headers, 664, 673, 678, 696  
host, 244, 1105  
hostnames, 343  
id, 259  
ID, 400, 419, 429, 436  
IDLE, 105  
INCREASE\_POLLING\_DELAY\_COUNT, 875  
INITIAL\_POLLING\_DELAY, 875  
invokeld, 726  
ioLog, 689, 691, 781, 788  
ioSession, 743  
items, 1060  
jetty, 593  
jettyConfig, 593  
keepAliveJobName, 726  
keepOnDisconnect, 253  
KEY, 1081, 1089, 1093, 1099  
lastModified, 297, 967  
lastPingSent, 726  
lastPingTime, 726  
lastPongReceived, 726  
level, 848  
listeners, 253, 608, 967  
listenerStats, 968  
loader, 291  
local, 286  
log, 88, 169, 197, 217, 244, 259, 277, 286, 306, 340, 344, 351, 360, 590, 591, 593, 594, 664, 666, 668, 673, 689, 691, 707, 711, 713, 726, 739, 743, 747, 781, 788, 842, 862, 879, 882, 893, 898, 953, 968, 1022, 1039, 1101  
LONG\_STRING\_LENGTH, 40, 58  
mapping, 335  
MASK\_SOUND\_FORMAT, 36  
MASK\_SOUND\_RATE, 36  
MASK\_SOUND\_SIZE, 36  
MASK\_SOUND\_TYPE, 36  
MASK\_VIDEO\_CODEC, 36  
MASK\_VIDEO\_FRAMETYPE, 36  
MAX\_INTEGER\_VALUE, 58  
MAX\_POLLING\_DELAY, 875  
maxInactivity, 726  
MEDIUM\_INT\_MAX, 832  
message, 840  
messageID, 602  
messageType, 602  
MIN\_INTEGER\_VALUE, 58  
mode, 747  
MODE\_CLIENT, 767  
MODE\_SERVER, 767  
modified, 968

name, 298, 666, 685, 968  
NAVIGATION, 136  
NC\_CALL\_BADVERSION, 853  
NC\_CALL\_FAILED, 854  
NC\_CONNECT\_APPSHUTDOWN, 854  
NC\_CONNECT\_CLOSED, 854  
NC\_CONNECT\_FAILED, 854  
NC\_CONNECT\_INVALID\_APPLICATION, 854  
NC\_CONNECT\_REJECTED, 854  
NC\_CONNECT\_SUCCESS, 854  
noPendingMessages, 875  
notifyMessages, 869  
NS\_CLEAR\_FAILED, 854  
NS\_CLEAR\_SUCCESS, 854  
NS\_DATA\_START, 855  
NS\_FAILED, 855  
NS\_INVALID\_ARGUMENT, 855  
NS\_PAUSE\_NOTIFY, 855  
NS\_PLAY\_COMPLETE, 855  
NS\_PLAY\_FAILED, 855  
NS\_PLAY\_FILE\_STRUCTURE\_INVALID, 855  
NS\_PLAY\_INSUFFICIENT\_BW, 855  
NS\_PLAY\_NO\_SUPPORTED\_TRACK\_FOUND, 855  
NS\_PLAY\_PUBLISHNOTIFY, 856  
NS\_PLAY\_RESET, 856  
NS\_PLAY\_START, 856  
NS\_PLAY\_STOP, 856  
NS\_PLAY\_STREAMNOTFOUND, 856  
NS\_PLAY\_SWITCH, 856  
NS\_PLAY\_UNPUBLISHNOTIFY, 856  
NS\_PUBLISH\_BADNAME, 856  
NS\_PUBLISH\_START, 857  
NS\_RECORD\_FAILED, 857  
NS\_RECORD\_NOACCESS, 857  
NS\_RECORD\_START, 857  
NS\_RECORD\_STOP, 857  
NS\_SEEK\_FAILED, 857  
NS\_SEEK\_NOTIFY, 857  
NS\_UNPAUSE\_NOTIFY, 857  
NS\_UNPUBLISHED\_SUCCESS, 857  
object, 797  
objects, 911  
oName, 306, 726, 750  
OPENED, 105  
operation, 13, 16  
OPERATION\_AUTHENTICATION, 14  
OPERATION\_PING, 14  
OPERATION\_POLL, 15  
OPERATION\_REGISTER, 15  
OPT\_REFERENCE, 180  
ownerMessage, 968  
packet, 678  
params, 244  
parent, 254  
parentContext, 277  
path, 244, 298, 968  
pendingCalls, 726  
pendingMessages, 869  
PERMISSIONS, 259  
PERSISTENCE\_NO\_NAME, 911  
persistenceClass, 254  
persistent, 298, 968  
PERSISTENT, 405  
PERSISTENT\_HEADER, 466  
persistentSO, 968  
PING\_CLIENT, 820  
pingInterval, 727  
POLLING, 405  
pollingDelay, 876  
PONG\_SERVER, 820  
positions, 116  
PROVIDER\_CONNECT\_PULL, 631  
PROVIDER\_CONNECT\_PUSH, 631  
PROVIDER\_DISCONNECT, 631  
providers, 608  
publishedName, 1061  
raw, 665  
READ, 105  
readBytes, 869  
readMessages, 244  
realm, 1105  
RECEIVED, 105  
recordingFilename, 1061  
red5AppCtx, 292  
refcount, 797  
referenceMode, 173  
refId, 173, 176  
refMap, 174, 176  
registered, 344  
registry, 259  
remoteAddress, 244  
remoteAddresses, 244  
remotePort, 244  
REPLACE\_GATEWAY\_URL, 466  
request, 679, 696  
required, 685  
resources, 911

rtmpsEngine, 866  
rtmptConfig, 879  
rtmptEngine, 887  
rtmptServer, 879  
saveStacks, 286  
SCHEDULED\_JOB, 920  
SCHEDULING\_SERVICE, 920  
schedulingService, 361  
scope, 244, 380, 679  
scopePath, 696  
SEND\_ALL, 1027  
SEND\_INTERFRAMES, 1027  
SEND\_KEYFRAMES, 1027  
SEND\_KEYFRAMES\_CHECK, 1027  
sendStats, 968  
SENT, 105  
SEPARATOR, 419, 436  
serializer, 688, 774, 863, 890  
server, 344, 739, 879, 887, 893  
SERVICE\_NAME, 668, 947  
serviceInvoker, 668, 701  
serviceMethodName, 932  
serviceName, 932  
serviceProvider, 701  
servlet, 876  
servletContext, 344  
servletMappings, 888  
session, 679  
SESSION\_KEY, 658  
sessionId, 245  
sharedObjects, 701  
shuttingDown, 344  
signature, 107  
SLASH, 335  
so, 985  
SO\_CLIENT\_CLEAR\_DATA, 832  
SO\_CLIENT\_DELETE\_DATA, 832  
SO\_CLIENT\_INITIAL\_DATA, 832  
SO\_CLIENT\_SEND\_MESSAGE, 832  
SO\_CLIENT\_STATUS, 832  
SO\_CLIENT\_UPDATE\_ATTRIBUTE, 833  
SO\_CLIENT\_UPDATE\_DATA, 833  
SO\_CONNECT, 833  
SO\_CREATION\_FAILED, 858  
SO\_DELETE\_ATTRIBUTE, 833  
SO\_DISCONNECT, 833  
SO\_NO\_READ\_ACCESS, 858  
SO\_NO\_WRITE\_ACCESS, 858  
SO\_PERSISTENCE\_MISMATCH, 858  
SO\_SEND\_MESSAGE, 833  
SO\_SET\_ATTRIBUTE, 833  
SOUND\_RATE\_11\_KHZ, 807  
SOUND\_RATE\_22\_KHZ, 807  
SOUND\_RATE\_44\_KHZ, 807  
SOUND\_RATE\_5\_5\_KHZ, 807  
SOUND\_SIZE\_16\_BIT, 807  
SOUND\_SIZE\_8\_BIT, 808  
source, 16, 797, 969  
stacks, 286  
state, 1003  
STATE\_CONNECT, 767  
STATE\_CONNECTED, 767  
STATE\_DISCONNECTED, 768  
STATE\_EDGE\_CONNECT\_ORIGIN\_SENT, 768  
STATE\_EDGE\_DISCONNECTING, 768  
STATE\_ERROR, 768  
STATE\_HANDSHAKE, 768  
STATE\_ORIGIN\_CONNECT\_FORWARDED, 768  
STATUS, 848  
status, 932  
STATUS\_ACCESS\_DENIED, 932  
STATUS\_APP\_SHUTTING\_DOWN, 933  
STATUS\_GENERAL\_EXCEPTION, 933  
STATUS\_INVOCATION\_EXCEPTION, 933  
STATUS\_METHOD\_NOT\_FOUND, 933  
STATUS\_PENDING, 933  
STATUS\_SERVICE\_NOT\_FOUND, 933  
STATUS\_SUCCESS\_NULL, 933  
STATUS\_SUCCESS\_RESULT, 933  
STATUS\_SUCCESS\_VOID, 933  
statusObjects, 863  
statusObjectService, 739  
storage, 969  
store, 298  
STREAM\_CLEAR, 820  
STREAM\_PLAYBUFFER\_CLEAR, 820  
STREAM\_RESET, 820  
streamBuffers, 727  
stringCache, 52  
subscopeStats, 307  
syncEvents, 969  
threadPool, 673  
timestamp, 798  
timestamps, 116  
TRANSIENT, 405  
TRANSIENT\_PREFIX, 457

type, 93, 245, 298  
 TYPE, 419, 437  
 TYPE\_AMF3\_OBJECT, 40  
 TYPE\_ARRAY, 40, 58  
 TYPE\_AUDIO, 36, 808  
 TYPE\_AUDIO\_DATA, 833  
 TYPE\_BOOLEAN, 40  
 TYPE\_BOOLEAN\_FALSE, 58  
 TYPE\_BOOLEAN\_TRUE, 58  
 TYPE\_BYTEARRAY, 59  
 TYPE\_BYTES\_READ, 833  
 TYPE\_CHUNK\_SIZE, 834  
 TYPE\_CLASS\_OBJECT, 40  
 TYPE\_CLIENT\_BANDWIDTH, 834  
 TYPE\_DATE, 40, 59  
 TYPE\_END\_OF\_OBJECT, 41  
 TYPE\_FLEX\_MESSAGE, 834  
 TYPE\_FLEX\_SHARED\_OBJECT, 834  
 TYPE\_FLEX\_STREAM\_SEND, 834  
 TYPE\_INTEGER, 59  
 TYPE\_INVOKE, 834  
 TYPE\_LONG\_STRING, 41  
 TYPE\_METADATA, 36, 808  
 TYPE\_MIXED\_ARRAY, 41  
 TYPE\_MOVIECLIP, 41  
 TYPE\_NOTIFY, 834  
 TYPE\_NULL, 41, 59  
 TYPE\_NUMBER, 41, 59  
 TYPE\_OBJECT, 41, 59  
 TYPE\_OBJECT\_EXTERNALIZABLE, 59  
 TYPE\_OBJECT\_PROPERTY, 59  
 TYPE\_OBJECT\_PROXY, 60  
 TYPE\_OBJECT\_VALUE, 60  
 TYPE\_PING, 834  
 TYPE\_RECORDSET, 41  
 TYPE\_REFERENCE, 41  
 TYPE\_SERVER\_BANDWIDTH, 834  
 TYPE\_SHARED\_OBJECT, 835  
 TYPE\_STREAM\_METADATA, 835  
 TYPE\_STRING, 42, 60  
 TYPE\_UNDEFINED, 42, 60  
 TYPE\_UNSUPPORTED, 42  
 TYPE\_VIDEO, 37, 808  
 TYPE\_VIDEO\_DATA, 835  
 TYPE\_XML, 42, 60  
 TYPE\_XML\_DOCUMENT, 60  
 typeMap, 842  
 UNDEFINED, 820  
 UNKNOWN\_2, 820  
 UNKNOWN\_5, 821  
 UNKNOWN\_8, 821  
 updateCounter, 969  
 VALUE\_FALSE, 42  
 VALUE\_TRUE, 42  
 valves, 1105  
 version, 108, 969  
 VERSION, 433  
 VIDEO\_ON2\_VP6, 808  
 VIDEO\_SCREEN\_VIDEO, 808  
 VIDEO\_SORENSEN\_H263, 808  
 virtualHosts, 344  
 WARNING, 848  
 webAppCtx, 893  
 webappFolder, 292  
 webappRoot, 1112  
 WRITTEN, 105  
 writtenBytes, 869  
 writtenMessages, 245  
 FileConsumer, 1090, 1091  
 FileKeyFrameMetaCache, 20  
 FilePersistence, 906, 907, 907  
 FilePersistenceThread, 908  
 FileProvider, 1098, 1099  
 FileStreamSource, 1021, 1021  
 filterClose, 102  
 filterNull, 235  
 filterWrite, 102  
 findApplication, 442  
 findMethodsByNameAndNumParams, 940  
 findMethodWithExactParameters, 949, 949  
 findMethodWithListParameters, 949, 949  
 findRoot, 442  
 finishDecode, 778  
 fireConsumerConnectionEvent, 608  
 firePipeConnectionEvent, 608  
 fireProviderConnectionEvent, 609  
 Flag, 182  
 FLAG\_CODEC\_H263, 34  
 FLAG\_CODEC\_SCREEN, 34  
 FLAG\_CODEC\_VP6, 34  
 FLAG\_FORMAT\_ADPCM, 34  
 FLAG\_FORMAT\_MP3, 34  
 FLAG\_FORMAT\_NELLYMOSER, 34  
 FLAG\_FORMAT\_NELLYMOSER\_8\_KHZ, 34  
 FLAG\_FORMAT\_RAW, 34  
 FLAG\_FRAMETYPE\_DISPOSABLE, 34  
 FLAG\_FRAMETYPE\_INTERFRAME, 35  
 FLAG\_FRAMETYPE\_KEYFRAME, 35

FLAG\_RATE\_11\_KHZ, 35  
 FLAG\_RATE\_22\_KHZ, 35  
 FLAG\_RATE\_44\_KHZ, 35  
 FLAG\_RATE\_5\_5\_KHZ, 35  
 FLAG\_SIZE\_16\_BIT, 35  
 FLAG\_SIZE\_8\_BIT, 35  
 FLAG\_TYPE\_MONO, 35  
 FLAG\_TYPE\_STEREO, 36  
 flagAudio, 107  
 flagReserved01, 107  
 flagReserved02, 107  
 flagVideo, 107  
 FlexMessage, 809  
 FlexMessagingService, 667  
 FlexSharedObjectMessage, 957, 957, 958  
 FlexStreamSend, 810, 810  
 flushHeaders, 111, 118  
 FLV, 117, 118, 118, 118  
 FLVData, 805  
 FLVHeader, 106  
 FLVReader, 120, 122, 122, 122  
 FLVService, 125  
 FLVWriter, 127, 128, 128  
 formatHTML, 896  
 formatPath, 1105  
 FORWARD\_KEY, 666  
 frameDuration, 166  
 frameSize, 167  
 FRAMETYPE\_DISPOSABLE, 807  
 FRAMETYPE\_INTERFRAME, 807  
 FRAMETYPE\_KEYFRAME, 807

**G**

generateErrorResult, 655  
 generateFilename, 541, 541, 1017, 1018  
 get, 8, 446, 570, 575, 580  
 getActiveClients, 308, 323, 505  
 getActiveConnections, 308, 323, 505  
 getActiveListeners, 508, 970  
 getActiveSubscopes, 309, 323, 505  
 getActiveSubscribers, 502, 1012  
 getAppendWriter, 22, 118, 165  
 getApplication, 860  
 getApplicationContext, 269, 292, 413, 570, 575, 580  
 getApplicationLoader, 292, 344  
 getArguments, 473, 934  
 getAttribute, 236, 236, 298, 381, 381, 390, 390, 680, 680, 954, 970, 986, 986  
 getAttributeNames, 236, 381, 391, 680, 987  
 getAttributes, 236, 381, 391, 680, 987  
 getAudio, 1039  
 getAudioBucket, 1019, 1023, 1066  
 getAudioCodecName, 539, 1087  
 getBandwidth, 804, 824  
 getBandwidthConfigure, 260, 393, 728, 1000  
 getBaseHost, 1106  
 getBasicScope, 309, 323, 420, 438  
 getBasicScopeNames, 309, 324  
 getBasicScopes, 246, 406, 680  
 getBean, 270, 413  
 getBitflags, 132  
 getBitRate, 167  
 getBody, 25, 132, 1095, 1097  
 getBodySize, 26, 132  
 getBoolAttribute, 236, 397, 680  
 getBooleanProperty, 603, 613  
 getBroadcastScope, 1074  
 getBroadcastStream, 366, 518  
 getBroadcastStreamNames, 366, 519, 1029, 1056  
 getBuffer, 88, 96  
 getBufferSize, 123  
 getBufferType, 123  
 getBWControllable, 1023, 1069  
 getByteAttribute, 237, 397, 680  
 getByteBuffer, 449, 573  
 getByteProperty, 603, 613  
 getBytes, 449, 573, 900  
 getBytesRead, 28, 123, 169, 800  
 getBytesReceived, 502, 1012  
 getBytesSent, 504, 1050  
 getBytesWritten, 31, 128  
 getCachedStatusObjectAsByteArray, 863  
 getCacheHit, 570, 575, 580  
 getCacheId, 765  
 getCacheManagerEventListener, 575  
 getCacheMiss, 570, 576, 580  
 getCall, 814, 816  
 getCallbacks, 471, 945  
 getCalls, 696  
 getCanSeekToEnd, 139, 147  
 getCapacity, 1036, 1041  
 getChannel, 728  
 getChannelBandwidth, 394, 561  
 getChannelId, 836  
 getChannelInitialBurst, 394, 561  
 getChildScope, 381

getChildScopeNames, 382  
getClassLoader, 270, 309  
getClassName, 916  
getClient, 246, 406, 433, 680, 882  
getClientBufferDuration, 504, 1001  
getClientBytesRead, 246, 406, 680, 728  
getClientid, 845, 848  
getClientId, 882  
getClientList, 265  
getClientRegistry, 270, 414  
getClientResponse, 694  
getClientResult, 695  
 getClients, 265, 309, 324, 382, 421, 438  
getClientTTL, 366  
getCode, 848, 860  
getCodec, 808  
getCodecFactory, 703, 878  
getCodecInfo, 532, 1003  
getCodeSection, 287  
getColumnNames, 192  
getConnection, 433, 520, 711, 1001  
getConnectionLocal, 434  
getConnectionParams, 816  
getConnections, 260, 260, 310, 324, 400, 400, 421, 438  
getConnectionsIter, 382  
getConnector, 1106  
getConnectParams, 246, 407, 680  
getConsumer, 632  
getConsumerOutput, 1017, 1026  
getConsumers, 609, 619, 1007  
getContext, 277, 310, 324, 382, 421, 434, 439  
getContextPath, 310, 324, 421, 439  
getCoreService, 270, 414  
getCorrelationID, 603, 614  
getCount, 221  
getCreationTime, 260, 310, 401, 510, 971, 1012, 1051  
getCurrent, 514  
getCurrentItem, 526, 1051, 1061  
getCurrentItemIndex, 527, 1051, 1061  
getCurrentTimestamp, 512, 1012, 1051  
getCursor, 195  
getData, 26, 64, 103, 132, 167, 195, 490, 545, 795, 816, 826, 828, 840, 971, 987, 1033, 1039  
getDataBucket, 1019, 1024, 1067  
getDataOffset, 108  
getDataType, 26, 132, 545, 795, 798, 801, 802, 804, 809, 810, 811, 814, 817, 821, 825, 827, 828, 836, 958, 981  
getDebug, 821  
getDecoder, 774, 776  
getDecoderBufferAmount, 659  
getDepth, 254, 310, 325, 382, 395, 505  
getDescription, 849, 860  
getDeserializer, 126, 151  
getDetails, 845, 849  
getDiskExpiryThreadIntervalSeconds, 576  
getDiskStore, 576  
getDoubleAttribute, 237, 397, 680  
getDoubleProperty, 603, 614  
getDownstreamBandwidth, 412, 562  
getDroppedMessages, 247, 407, 680  
getDuration, 28, 123, 139, 147, 170  
getEmbedded, 1106  
getEnabled, 311, 325  
getEncoder, 774, 776  
getEncoding, 407, 681, 697, 728, 768  
getEndian, 64, 71, 75, 78, 82  
getEngine, 1106  
getEstimatedBufferFill, 504, 1051  
getEventListeners, 254, 454  
getEvents, 961, 981  
getException, 473, 934  
getExecutor, 102, 1051  
getExtension, 18, 24, 126, 172  
getFile, 29, 31, 123, 129, 151, 170  
getFileData, 124  
getFlagAudio, 108  
getFlagReserved01, 108  
getFlagReserved02, 108  
getFlagVideo, 108  
getFloatProperty, 603, 614  
getFrameRate, 139, 148  
getFrameType, 828  
getGhostConnsCleanupPeriod, 367  
getGlobal, 336, 430  
getGlobalNames, 336, 430  
getGlobalScope, 271, 332, 414, 427, 894  
getGlobalScopes, 336, 430  
getHandler, 311, 325, 421, 439  
getHeader, 798, 811, 840  
getHeaderLength, 755  
getHeaders, 465, 681, 697  
getHeight, 139, 148  
getHost, 247, 407, 681, 1106, 1113

getHostname, 707  
getId, 261, 401, 711  
getInstance, 570, 580, 793, 909  
getIntAttribute, 237, 398, 681  
getIntProperty, 603, 614  
getInvokedId, 728, 817  
getIoSession, 743  
getItem, 527, 1051, 1061  
getItemAt, 193  
getItemSize, 527, 1051, 1061  
getJobName, 919, 921  
getKey, 337, 959, 977  
getKeyframe, 557, 1085, 1086  
getKeyFrameData, 112, 118  
getLastModified, 298, 458, 971  
getLastPingTime, 408, 681, 729  
getLastReadChannel, 768  
getLastReadHeader, 769  
getLastReadPacket, 769  
getLastWriteChannel, 769  
getLastWriteHeader, 769  
getLastWritePacket, 769  
getLength, 193, 524, 564  
getLevel, 849, 860  
getListAttribute, 238, 398, 681  
getListeners, 367, 609, 971  
getLiveProviderInput, 1029, 1056  
getLongAttribute, 238, 398, 681  
getLongProperty, 603, 614  
getMapAttribute, 238, 398, 681  
getMappingStrategy, 271, 414  
getMappingTable, 337, 430  
getMax, 514  
getMaxClients, 311, 325, 506  
getMaxConnections, 311, 325, 506  
getMaxListeners, 508, 971  
getMaxSubscopes, 311, 326, 506  
getMaxSubscribers, 502, 1012  
getMemoryStoreEvictionPolicy, 576  
getMessage, 840  
getMessageCount, 1041  
getMessageID, 604, 615  
getMessageInput, 524, 564  
getMessageSize, 1041  
getMessageType, 604, 615  
getMetaCue, 139, 148  
getMetaData, 112, 119  
getMethodName, 916  
getMethodParams, 916  
getMinaDecoder, 776  
getMinaEncoder, 776  
getMode, 770  
getMsgInput, 564  
getMustUnderstand, 466, 685  
getName, 137, 145, 299, 383, 395, 449, 458, 466, 506, 508, 524, 533, 557, 565, 573, 685, 962, 971, 981, 987, 1003, 1085, 1086  
getNext, 156  
getNextAvailableChannelId, 729  
getNextFilter, 104  
getNextRTMPMessage, 1061  
getNumberAvailable, 193  
getObject, 451, 798, 981  
getObjectId, 911  
getObjectName, 911  
getObjectNames, 447, 460, 571, 576, 580, 912  
getObjectPath, 907, 912  
getObjectProperty, 604, 615  
getObjects, 447, 461, 571, 576, 581, 912  
getOffset, 29, 31, 124, 129, 170  
getOnDemandStream, 367, 523  
getParameters, 584  
getParamMap, 632  
getParamTypes, 916  
getParent, 255, 311, 326, 383, 396  
getParentBWControllable, 261, 393, 729, 1001  
getParentContext, 277  
getPath, 247, 255, 299, 312, 326, 383, 396, 408, 458, 506, 681, 972, 987  
getPendingCall, 729  
getPendingMessages, 247, 408, 681, 743, 870, 870, 876  
getPendingVideoMessages, 248, 537, 681, 729  
getPermissions, 261, 401  
getPersistanceStore, 271, 414  
getPersistenceStore, 463  
getPollingDelay, 876  
getPrefix, 18, 24, 126, 172  
getPreviousTagSize, 26, 133  
getPreviosTagSize, 133  
getProvider, 516, 632, 1013, 1061  
getProviderInput, 1029, 1056  
getProviders, 609, 621, 1007  
getPublishedName, 503, 517, 1013, 1062  
getReadBytes, 248, 408, 682, 729, 743, 870  
getReadChunkSize, 770  
getReader, 22, 119, 165  
getReadMessages, 248, 408, 682

getRealm, 1107  
getReason, 586  
getRed5ApplicationContext, 292  
getReference, 174  
getReferenceld, 176  
getRemoteAddress, 248, 409, 682  
getRemoteAddresses, 249, 409, 682, 901  
getRemotePort, 249, 409, 682  
getResolver, 151  
getResource, 271, 312, 326, 383  
getResources, 272, 312, 326, 383  
getResult, 471, 628, 916, 945  
getRootContext, 389, 591, 1102  
getRunning, 312, 327  
getSampleRate, 167  
getSaveFilename, 503, 517, 1013, 1062  
getScheduledJobNames, 367, 470, 919  
getScope, 249, 272, 313, 327, 384, 409, 422, 434, 439, 533, 682, 1003  
getScopeAttributes, 997, 998  
getScopeNames, 313, 327, 422, 439  
getScopePath, 697  
getScopeResolver, 272  
getScopes, 261, 401, 510, 511, 994, 994, 998, 998  
getScopeService, 442, 442, 443, 443  
getScopeStatisticsSO, 511, 995  
getScriptedObject, 924, 925, 928  
getScriptInterfaces, 924, 925, 928  
getScriptSourceLocator, 926, 928  
getSerializer, 126, 151  
getServer, 289, 313, 344, 416  
getServerStream, 567  
getService, 37  
getServiceHandler, 313, 327, 478, 954, 987  
getServiceHandlerNames, 314, 327, 478, 954, 987  
getServiceHandlerProvider, 479  
getServiceHandlers, 314, 314  
getServiceInvoker, 272, 415  
getServiceMethodName, 473, 934  
getServiceName, 474, 628, 934, 948  
getServiceParamMap, 629  
getServices, 23, 37  
getSessionId, 249, 410, 682  
getSetAttribute, 238, 399, 682  
getSharedObject, 368, 368, 500, 500, 703, 992, 992  
getSharedObjectName, 368, 500, 993  
getSharedObjects, 511, 995, 998  
getSharedObjectSecurity, 368, 497, 987  
getSharedObjectStatisticsSO, 511, 995  
getShortAttribute, 239, 399, 682  
getShortProperty, 604, 615  
getSignature, 109  
getSimpleDecoder, 659, 688, 774, 777, 890  
getSimpleEncoder, 660, 688, 775, 777, 890  
getSize, 525, 565, 802, 836  
getSource, 451, 798  
getSpeed, 1036  
getStart, 525, 565  
getState, 770  
getStatistics, 314, 422, 488, 519, 531, 987, 1013, 1051  
getStatus, 474, 934  
getStatusObject, 863  
getStore, 289, 299, 458, 972, 987  
getStreamableFile, 19, 24, 126, 172  
getStreamAwareHandler, 1004  
getStreamByChannelId, 729  
getStreamById, 538, 730  
getStreamId, 520, 707, 837, 1001  
getStreamIdForChannel, 730  
getStreamLength, 369  
getStreamListeners, 517, 1013  
getStreamPlaybackSecurity, 369, 548  
getStreamPublishSecurity, 369, 548  
getStreams, 730  
getString, 44, 44, 89, 156, 184  
getStringAttribute, 239, 399, 682  
getStringProperty, 604, 615  
getSubscriberStream, 369, 556  
getTarget, 629  
getTime, 137, 145  
getTimeout, 287  
getTimeoutMillis, 287  
getTimer, 837  
getTimestamp, 26, 133, 546, 798, 811  
getTotal, 515  
getTotalBytes, 29, 124, 170  
getTotalChanges, 508, 972  
getTotalClients, 314, 328, 507  
getTotalConnections, 315, 328, 507  
getTotalDeletes, 508, 972  
getTotalListeners, 509, 972  
getTotalSends, 509, 972  
getTotalSubscopes, 315, 328, 507  
getTotalSubscribers, 503, 1013

getType, 104, 133, 137, 145, 250, 299, 396, 410, 451, 459, 632, 682, 798, 959, 972, 978, 981, 987  
 getUpstreamBandwidth, 412, 563  
 getUpTime, 434  
 getUsedStreamCount, 730  
 getValue, 466, 686, 959, 978  
 getValue1, 821  
 getValue2, 804, 821  
 getValue3, 821  
 getValue4, 822  
 getVersion, 109, 434, 491, 509, 962, 973, 981, 987  
 getVideo, 1039  
 getVideoBucket, 1019, 1024, 1067  
 getVideoCodec, 539, 1081, 1087  
 getVideoCodecFactory, 730  
 getVideoCodeclId, 140, 148  
 getVideoCodecName, 540, 1088  
 getVideoDataRate, 140, 148  
 getVODProviderFile, 1030, 1057  
 getVODProviderInput, 1030, 1057  
 getWebAppContext, 595  
 getWidth, 140, 148  
 getWriteChunkSize, 770  
 getWriter, 22, 119, 165  
 getWrittenBytes, 250, 410, 682, 731, 744, 870  
 getWrittenMessages, 250, 410, 683  
 globals, 335  
 GlobalScope, 288  
 globalScope, 332  
 GroovyScriptFactory, 922, 923, 923

hasAttribute, 239, 384, 391, 683, 988  
 hasAudio, 540, 1088  
 hasBroadcastStream, 369, 519  
 hasChildScope, 315, 315, 328, 328, 384, 422, 422, 440, 440  
 hasClient, 265, 403  
 hasClients, 266  
 hasContext, 316, 329  
 hasDecodedObject, 659  
 hasEventsWaiting, 638  
 hasHandler, 316, 329, 423, 440  
 hashCode, 262, 803  
 hasKeyFrameData, 112, 119  
 hasMetaData, 112, 119  
 hasMore, 1022, 1034  
 hasMoreItems, 527, 1052, 1062  
 hasMoreProperties, 45  
 hasMoreTags, 29, 124, 170  
 hasNext, 256  
 hasOnDemandStream, 370, 523  
 hasParent, 255, 316, 329, 384, 396  
 hasPermission, 262, 402  
 hasReference, 176  
 hasSharedObject, 370, 500, 993  
 hasSource, 451, 798  
 hasVideo, 29, 124, 170, 540, 1034, 1088, 1099  
 header, 797, 840  
 Header, 835  
 HEADER\_CONTINUE, 831  
 HEADER\_ENDPOINT, 14  
 HEADER\_NEW, 832  
 HEADER\_SAME\_SOURCE, 832  
 HEADER\_TIMER\_CHANGE, 832  
 headers, 664, 673, 678, 696  
 HexCharset, 203  
 HexCharset.Decoder, 204  
 HexCharset.Encoder, 204  
 HexDump, 205  
 hexStringToByteArray, 207  
 host, 244, 1105  
 hostnames, 343

**H**

handleBadRequest, 882  
 handleClose, 883  
 handleEvent, 250, 255, 282, 315, 348, 453, 683  
 handleIdle, 883  
 handleOpen, 883  
 handlePendingCallResult, 707  
 handler, 747, 869  
 HANDLER\_ATTRIBUTE, 878  
 HANDLER\_ERROR, 694  
 HANDLER\_SUCCESS, 694  
 handleRemotingPacket, 894  
 handleRequest, 668, 668, 668, 669  
 HandlerServiceResolver, 940  
 handleSend, 884  
 HANDSHAKE\_SIZE, 831  
 HandshakeFailedException, 655, 656

hasAttribute, 239, 384, 391, 683, 988  
 hasAudio, 540, 1088  
 hasBroadcastStream, 369, 519  
 hasChildScope, 315, 315, 328, 328, 384, 422, 422, 440, 440  
 hasClient, 265, 403  
 hasClients, 266  
 hasContext, 316, 329  
 hasDecodedObject, 659  
 hasEventsWaiting, 638  
 hasHandler, 316, 329, 423, 440  
 hashCode, 262, 803  
 hasKeyFrameData, 112, 119  
 hasMetaData, 112, 119  
 hasMore, 1022, 1034  
 hasMoreItems, 527, 1052, 1062  
 hasMoreProperties, 45  
 hasMoreTags, 29, 124, 170  
 hasNext, 256  
 hasOnDemandStream, 370, 523  
 hasParent, 255, 316, 329, 384, 396  
 hasPermission, 262, 402  
 hasReference, 176  
 hasSharedObject, 370, 500, 993  
 hasSource, 451, 798  
 hasVideo, 29, 124, 170, 540, 1034, 1088, 1099  
 header, 797, 840  
 Header, 835  
 HEADER\_CONTINUE, 831  
 HEADER\_ENDPOINT, 14  
 HEADER\_NEW, 832  
 HEADER\_SAME\_SOURCE, 832  
 HEADER\_TIMER\_CHANGE, 832  
 headers, 664, 673, 678, 696  
 HexCharset, 203  
 HexCharset.Decoder, 204  
 HexCharset.Encoder, 204  
 HexDump, 205  
 hexStringToByteArray, 207  
 host, 244, 1105  
 hostnames, 343

**I**

Application, 353  
 ApplicationContext, 388  
 ApplicationLoader, 389  
 AttributeStore, 389  
 BandwidthConfigure, 394  
 BasicScope, 395

IBroadcastScope, 1025  
IBroadcastStream, 516  
IBroadcastStreamService, 518  
IBWControlContext, 1022  
IBWControllable, 392  
IBWControlService, 1023  
ICacheable, 448  
ICacheStore, 446  
ICastingAttributeStore, 396  
IClient, 399  
IClientBroadcastStream, 519  
IClientBroadcastStreamStatistics, 502  
IClientRegistry, 402  
IClientSharedObject, 487  
IClientStream, 520  
IConnection, 403  
IConnection.Encoding, 411  
IConnectionBWConfig, 412  
IConnectionEventQueue, 638  
IConnectionListener, 455  
IConsumer, 611  
IConsumerService, 1025  
IContext, 412  
ICoreObject, 416  
ICueType, 136  
ICustomSerializable, 182  
id, 259  
ID, 400, 419, 429, 436  
IDataInput, 77  
IDataOutput, 81  
IDLE, 105  
IEchoService, 231  
IEvent, 451  
IEvent.Type, 452  
IEventDecoder, 758  
IEventDispatcher, 452  
IEventEncoder, 761  
IEventHandler, 453  
IEventListener, 453  
IEventObservable, 454  
IExternalizable, 86  
IFilter, 612  
IFLV, 111  
IFLVService, 114  
IFrameDropper, 1026  
IGlobalScope, 416  
IKeyFrameDataAnalyzer, 114  
IKeyFrameDataAnalyzer.KeyFrameMeta, 115  
IKeyFrameMetaCache, 21  
IMappingStrategy, 417  
IMessage, 612  
IMessageComponent, 618  
IMessageInput, 619  
IMessageOutput, 621  
IMeta, 136  
IMetaCue, 136  
IMetaData, 138  
IMetaService, 142  
IMP3, 164  
IMP3Service, 164  
IMRTMPConnection, 641  
IMRTMPEdgeManager, 642  
IMRTMPManager, 642  
IMRTMPOriginManager, 643  
includeStacktrace, 584  
INCREASE\_POLLING\_DELAY\_COUNT, 875  
increment, 515  
INetStreamEventHandler, 717  
init, 278, 316, 329, 688, 775, 777, 866, 879, 884, 888, 890, 894, 897, 902, 1107, 1113  
INITIAL\_POLLING\_DELAY, 875  
initialize, 251, 410, 683, 863  
InMemoryPullPullPipe, 625  
InMemoryPushPushPipe, 626  
Input, 42, 44, 86, 88, 88, 155, 183  
Input.ClassReference, 92, 92  
Input.PendingObject, 93  
Input.RefStorage, 93  
Interfaces  
    ApplicationMBean, 352  
    AttributeStoreMBean, 386  
    ClientBroadcastStreamMBean, 1016  
    ClientExceptionHandler, 712  
    ClientMBean, 264, 386  
    ClientRegistryMBean, 267, 387  
    ConnectionMBean, 387  
    Constants, 830  
    ContextLoaderMBean, 279  
    ContextMBean, 279  
    CoreHandlerMBean, 283  
    IApplication, 353  
    IApplicationContext, 388  
    IApplicationLoader, 389  
    IAttributeStore, 389  
    IBandwidthConfigure, 394  
    IBasicScope, 395  
    IBroadcastScope, 1025  
    IBroadcastStream, 516

- IBroadcastStreamService, 518  
IBWControlContext, 1022  
IBWControllable, 393  
IBWControlService, 1023  
ICacheable, 448  
ICacheStore, 446  
ICastingAttributeStore, 396  
IClient, 399  
IClientBroadcastStream, 519  
IClientBroadcastStreamStatistics, 502  
IClientRegistry, 402  
IClientSharedObject, 487  
IClientStream, 520  
IConnection, 403  
IConnectionBWConfig, 412  
IConnectionEventQueue, 638  
IConnectionListener, 455  
IConsumer, 611  
IConsumerService, 1026  
IContext, 412  
ICoreObject, 416  
ICueType, 136  
ICustomSerializable, 182  
IDataInput, 77  
IDataOutput, 81  
IEchoService, 231  
IEvent, 451  
IEventDecoder, 758  
IEventDispatcher, 452  
IEventEncoder, 761  
IEventHandler, 453  
IEventListener, 453  
IEventObservable, 454  
IExternalizable, 86  
IFilter, 612  
IFLV, 111  
IFLVService, 114  
IFrameDropper, 1026  
IGlobalScope, 416  
IKeyFrameDataAnalyzer, 114  
IKeyFrameMetaCache, 21  
IMappingStrategy, 417  
IMessage, 612  
IMessageComponent, 618  
IMessageInput, 619  
IMessageOutput, 621  
IMeta, 136  
IMetaCue, 136  
IMetaData, 138  
IMetaService, 142  
IMP3, 164  
IMP3Service, 164  
IMRTMPConnection, 641  
IMRTMPEdgeManager, 642  
IMRTMPManager, 642  
IMRTMPOriginManager, 643  
INetStreamEventHandler, 717  
Input, 183  
IoConstants, 32  
IOnDemandStream, 521  
IOnDemandStreamService, 523  
IPassive, 622  
IPendingServiceCall, 471  
IPendingServiceCallback, 472  
IPersistable, 457  
IPersistenceStore, 460  
IPipe, 623  
IPipeConnectionListener, 623  
IPlayItem, 524  
IPlaylist, 525  
IPlaylistController, 529  
IPlaylistSubscriberStream, 530  
IPlaylistSubscriberStreamStatistics, 503  
IProvider, 624  
IProviderService, 1028  
IPullableProvider, 624  
IPushableConsumer, 624  
IRemotingCallback, 670  
IRemotingConnection, 464  
IRemotingHeader, 465  
IResolver, 143  
IRTMPConnManager, 717  
IRTMPEvent, 810  
IRTMPHandler, 717  
IScheduledJob, 468  
ISchedulingService, 468  
IScope, 418  
IScopeAware, 423  
IScopeHandler, 424  
IScopeListener, 455  
IScopeResolver, 427  
IScopeService, 428  
IScopeStatistics, 504  
ISeekableProvider, 1031  
ISeekableStreamSource, 1032  
IServer, 428  
IServerStream, 531  
IServiceCall, 472

IServiceCapableConnection, 474  
IServiceHandlerProvider, 477  
IServiceHandlerProviderAware, 478  
IServiceInvoker, 479  
IServiceResolver, 941  
ISharedObject, 487  
ISharedObjectBase, 489  
ISharedObjectEvent, 958  
ISharedObjectHandlerProvider, 492  
ISharedObjectListener, 492  
ISharedObjectMessage, 960  
ISharedObjectSecurity, 495  
ISharedObjectSecurityService, 497  
ISharedObjectService, 498  
ISharedObjectStatistics, 507  
ISingleItemSubscriberStream, 531  
IStatisticsBase, 509  
IStatisticsService, 510  
IStream, 532  
IStreamableFile, 22  
IStreamableFileFactory, 23  
IStreamableFileService, 23  
IStreamAwareScopeHandler, 533  
IStreamCapableConnection, 537  
IStreamCodecInfo, 539  
IStreamControl, 1032  
IStreamData, 1033  
IStreamFilenameGenerator, 540  
IStreamHandler, 542  
IStreamListener, 544  
IStreamPacket, 545  
IStreamPlaybackSecurity, 546  
IStreamPublishSecurity, 546  
IStreamSecurityService, 547  
IStreamService, 549  
IStreamSource, 1033  
IStreamStatistics, 512  
IStreamTypeAwareProvider, 1034  
ISubscriberStream, 553  
ISubscriberStreamService, 555  
ITag, 25  
ITagReader, 28  
ITagWriter, 30  
ITokenBucket, 1034  
ITokenBucket.ITokenBucketCallback, 1036  
ITokenBucketService, 1037  
IVideoStreamCodec, 556  
ListMBean, 290  
LoaderMBean, 294  
Output, 187  
QuartzSchedulingServiceMBean, 920  
Red5MBean, 435  
RTMPMinaConnectionMBean, 744  
RTMPMinaTransportMBean, 751  
ScopeMBean, 321, 435  
SimpleProtocolCodecFactory, 659  
SimpleProtocolDecoder, 660  
SimpleProtocolEncoder, 661  
StatusCodes, 851  
ThreadPoolMBean, 914  
TomcatVHostLoaderMBean, 1114  
invoke, 475, 475, 476, 476, 476, 476, 479, 480, 703, 703, 731, 731, 731, 731, 731, 731, 947, 947  
Invoke, 812, 813, 813, 813  
invokeCall, 739  
invokedId, 726  
invokeMethod, 673, 673  
invokeOnAllConnections, 481, 481, 482, 482  
invokeOnClient, 482, 483  
invokeOnConnection, 483, 483, 484, 484  
IoConstants, 32  
ioLog, 689, 691, 781, 788  
IOnDemandStream, 521  
IOnDemandStreamService, 523  
ioSession, 743  
IOUtils, 213  
IPassive, 622  
IPendingServiceCall, 471  
IPendingServiceCallback, 472  
IPersistable, 457  
IPersistenceStore, 460  
IPipe, 623  
IPipeConnectionListener, 623  
IPlayItem, 524  
IPlaylist, 525  
IPlaylistController, 529  
IPlaylistSubscriberStream, 530  
IPlaylistSubscriberStreamStatistics, 503  
IProvider, 624  
IProviderService, 1028  
IProviderService.INPUT\_TYPE, 1031  
IPullableProvider, 624  
IPushableConsumer, 624  
IRemotingCallback, 670  
IRemotingConnection, 464  
IRemotingHeader, 465  
IResolver, 143  
IRTMPConnManager, 717

IRTMPEvent, 810  
IRTMPHandler, 717  
isAcquired, 488, 973, 988  
isAncestor, 443  
isApp, 444  
isBasicType, 180  
isCached, 449, 573  
isChannelUsed, 732  
IScheduledJob, 468  
ISchedulingService, 468  
isClosing, 870  
isComplexType, 180  
isConnected, 251, 411, 487, 683, 744, 954  
isConnectionAllowed, 495, 988  
IScope, 418  
IScopeAware, 423  
IScopeHandler, 424  
IScopeListener, 455  
IScopeResolver, 427  
IScopeService, 428  
IScopeStatistics, 504  
isCreationAllowed, 496  
isCustom, 52, 160, 188  
isCustomType, 180  
isDebug, 770  
isDeleteAllowed, 496, 988  
isDisposable, 809  
ISeekableProvider, 1031  
ISeekableStreamSource, 1032  
isEmpty, 962, 981  
isEnabled, 316  
IServer, 428  
IServerStream, 531  
IServiceCall, 472  
IServiceCapableConnection, 474  
IServiceHandlerProvider, 477  
IServiceHandlerProviderAware, 478  
IServicelInvoker, 479  
IServiceResolver, 941  
isFullyPopulated, 193  
isGlobal, 444  
ISharedObject, 487  
ISharedObjectBase, 488  
ISharedObjectEvent, 958  
ISharedObjectEvent.Type, 959  
ISharedObjectHandlerProvider, 492  
ISharedObjectListener, 492  
ISharedObjectMessage, 960  
ISharedObjectSecurity, 495  
ISharedObjectSecurityService, 497  
ISharedObjectService, 498  
ISharedObjectStatistics, 507  
ISingleItemSubscriberStream, 531  
isLocked, 491, 954, 988  
isPaused, 521, 554, 1052  
isPersistent, 300, 459, 962, 973, 981  
isPersistentObject, 491, 509, 973, 988  
isPlaybackAllowed, 546, 560  
isPlaying, 521  
isProtected, 167  
isPublishAllowed, 547, 560  
isPullMode, 1044  
isRandom, 527, 1052, 1062  
isRelative, 1078  
isRepeat, 528, 1052, 1062  
isRewind, 528, 1052, 1062  
isRoom, 444  
isRoot, 444  
isRunning, 317  
isSendAllowed, 496, 988  
isShuttingDown, 344  
isStereo, 168  
isStopped, 522  
isSuccess, 474, 934  
IStatisticsBase, 509  
IStatisticsService, 510  
isTimerRelative, 837  
IStream, 532  
IStreamableFile, 21  
IStreamableFileFactory, 23  
IStreamableFileService, 23  
IStreamAwareScopeHandler, 533  
IStreamCapableConnection, 537  
IStreamCodeclInfo, 539  
IStreamControl, 1032  
IStreamData, 1033  
IStreamFilenameGenerator, 540  
IStreamFilenameGenerator.GenerationType, 542  
IStreamHandler, 542  
IStreamListener, 544  
IStreamPacket, 545  
IStreamPlaybackSecurity, 546  
IStreamPublishSecurity, 546  
IStreamSecurityService, 547  
IStreamService, 549  
IStreamSource, 1033  
IStreamStatistics, 512  
IStreamTypeAwareProvider, 1034

- ISubscriberStream, 553  
 ISubscriberStreamService, 555  
 isWriteAllowed, 497, 989  
**I**  
 ITag, 25  
 ITagReader, 27  
 ITagWriter, 30  
 items, 1060  
 iterateScopeNameList, 262  
 iterator, 255, 317  
 ITOKENBucket, 1034  
 ITOKENBucket.ITOKENBucketCallback, 1036  
 ITOKENBucketService, 1037  
 IVideoStreamCodec, 556
- J**  
 jetty, 593  
 JettyApplicationContext, 590, 590  
 JettyApplicationLoader, 590, 591  
 jettyConfig, 593  
 JettyLoader, 592  
 JMXAgent, 596  
 JMXFactory, 598  
 JMXUtil, 599  
 join, 282, 348, 351, 370, 425  
 JythonScriptFactory, 925
- K**  
 keepAliveJobName, 726  
 keepOnDisconnect, 253  
 KEY, 1080, 1089, 1093, 1099  
 killGhostConnections, 370
- L**  
 lastModified, 297, 967  
 lastPingSent, 726  
 lastPingTime, 726  
 lastPongReceived, 726  
 leave, 282, 348, 352, 371, 426  
 length, 64  
 level, 848  
 listeners, 253, 608, 967  
 listenerStats, 967  
 ListMBean, 290  
 load, 461, 461, 907, 907, 912, 912  
 loadApplication, 389, 591, 1102  
 loadContext, 278  
 loader, 291  
 LoaderBase, 290  
 LoaderMBean, 294
- loadKeyFrameMeta, 20, 21  
 loadStatusObjects, 863  
 local, 286  
 lock, 491, 955, 989  
 log, 88, 169, 197, 217, 244, 259, 277, 286, 306, 340, 344, 351, 360, 590, 591, 593, 594, 664, 666, 668, 673, 689, 691, 707, 711, 713, 726, 739, 743, 747, 781, 788, 841, 862, 879, 882, 893, 898, 953, 968, 1022, 1039, 1101  
 LoggerContextFilter, 219  
 LoggingContextSelector, 220  
 LONG\_STRING\_LENGTH, 40, 58  
 lookupClient, 266, 403  
 lookupConnections, 317, 423, 440  
 lookupContext, 1019, 1024, 1067  
 lookupGlobal, 337, 431  
 lookupMRTMPConnection, 642  
 lookupProviderInput, 1030, 1057  
 lookupScopeHandler, 273, 415  
 lookupService, 273, 415
- M**  
 main, 207, 340, 341  
 makeDefaultConnectionParams, 704  
 mapping, 335  
 MappingStrategy, 294  
 mapResourcePrefix, 294, 417  
 mapScopeHandlerName, 295, 417  
 mapServiceName, 295, 417  
 MASK\_SOUND\_FORMAT, 36  
 MASK\_SOUND\_RATE, 36  
 MASK\_SOUND\_SIZE, 36  
 MASK\_SOUND\_TYPE, 36  
 MASK\_VIDEO\_CODEC, 36  
 MASK\_VIDEO\_FRAMETYPE, 36  
 MAX\_INTEGER\_VALUE, 58  
 MAX\_POLLING\_DELAY, 875  
 maxInactivity, 726  
 measureBandwidth, 371, 371  
 MEDIUM\_INT\_MAX, 832  
 message, 840  
 messageDropped, 732  
 messageID, 602  
 messageReceived, 102, 224, 662, 665, 666, 708, 718, 732, 747, 872, 878, 903  
 messageSent, 102, 224, 708, 719, 732, 747  
 messageType, 602  
 MetaCue, 144, 145  
 MetaData, 146, 147

MetaService, 150, 151  
MethodNotFoundException, 941, 942, 942  
Methods  
    AbstractMessage, 10  
    accept, 1110  
    acquire, 488, 969, 986  
    acquireToken, 1035  
    acquireTokenBestEffort, 1035  
    acquireTokenNonblocking, 1036  
    add, 1078  
    addAlias, 1112  
    addChildScope, 281, 307, 348, 419, 425, 437  
    addClient, 225, 265  
    addContext, 1105  
    addData, 556, 1084, 1086  
    addEvent, 961, 961, 980, 980  
    addEventListener, 254, 454, 986  
    addHeader, 464, 464, 673, 679, 679  
    addItem, 526, 526, 1050, 1050, 1061, 1061  
    addListener, 335, 335, 361, 429, 429  
    addMapping, 335, 430  
    addParameters, 4, 5, 10, 12, 13, 17  
    addPipeConnectionListener, 608, 623, 1007  
    addScheduledJob, 361, 469, 918  
    addScheduledJobAfterDelay, 361, 469, 919  
    addScheduledOnceJob, 362, 362, 469, 470, 919, 919  
    addSharedObjectListener, 489, 953, 986  
    addStreamListener, 516, 1012  
    addTask, 632  
    addValve, 1112  
    allocate, 286  
    analyzeKeyFrames, 115, 122, 169  
    appConnect, 353, 362  
    appDisconnect, 354, 363  
    appJoin, 354  
    appLeave, 354, 363  
    appStart, 354, 363  
    appStop, 354, 364  
    associate, 643, 650  
    asStatus, 859  
    AttributeStore, 235, 235, 235  
    AudioData, 795  
    available, 1037, 1043  
    BaseConnection, 243  
    BaseEvent, 797, 797  
    BaseOutput, 176  
    BasicScope, 253  
beginUpdate, 490, 490, 953, 953, 969, 969, 986, 986  
BroadcastScope, 1006  
buf, 52  
bufferDecoding, 658  
ByteArray, 63, 63  
byteArrayToBinaryString, 206  
byteArrayToHexString, 206, 207  
bytesAvailable, 63  
BytesRead, 800  
cacheStatusObjects, 863  
Call, 931, 931, 932  
cancelGhostConnectionsCleanup, 364  
canContinueDecoding, 658  
canDropFrames, 557, 1084, 1086  
canHandle, 18, 24  
canHandleData, 557, 1085, 1086  
canSendPacket, 1027, 1082  
canStartDecoding, 658  
Channel, 711  
checkRelease, 970  
checkRemoveEmptyDirectories, 907  
chunkBuffer, 764  
ChunkSize, 802  
clear, 490, 953, 961, 970, 980, 986, 1040  
clearReferences, 174, 176, 184, 188  
clearSharedObjects, 364, 499, 991  
Client, 259  
ClientDetailsException, 584, 584  
ClientManager, 225  
ClientNotFoundException, 585  
ClientRejectedException, 586, 586  
ClientSharedObject, 953  
clone, 836  
close, 28, 31, 123, 128, 169, 245, 405, 490, 532, 636, 637, 679, 711, 727, 743, 870, 953, 970, 986, 1012, 1022, 1033, 1039, 1044, 1050, 1061, 1093  
closeChannel, 727  
closeStream, 550, 1073, 1073  
compareTo, 145  
compress, 63  
configureClassLoader, 594  
configureDefaults, 594  
configureWebApp, 594  
connect, 245, 245, 281, 282, 307, 307, 348, 351, 365, 405, 406, 419, 420, 425, 437, 437, 487, 634, 639, 641, 679, 679, 701, 701, 701, 702, 702, 953

connectionClosed, 223, 707, 718  
ConnectionConsumer, 1089  
connectionOpened, 223, 702, 707, 718  
containsKey, 8  
Context, 269, 269  
contextDestroyed, 1117  
contextInitialized, 1117  
continueDecoding, 658  
convert, 937  
convertArrayToList, 937  
convertArrayToSet, 938  
convertBeanToMap, 938  
convertMapToBean, 938  
convertNumberToWrapper, 938  
convertParams, 939  
convertStringToWrapper, 939  
convertToArray, 939  
convertToWrappedPrimitive, 940  
copy, 899, 900  
copyThenClose, 900  
create, 793  
createChildScope, 308, 323, 380, 420, 437  
createHost, 1112  
createOutputStream, 727  
createRhinoObject, 929  
createServerStream, 567  
createSharedObject, 365, 499, 992  
createStream, 550, 1074  
createStreamName, 727  
createTokenBucket, 1038  
DataInput, 70  
DataOutput, 74  
debug, 214  
DebugPooledByteBufferAllocator, 285, 285, 286, 286  
decode, 660, 689, 778, 782, 870  
decodeAudioData, 758, 782  
decodeBuffer, 660, 689, 782  
decodeBytesRead, 758, 782  
decodeCalls, 690  
decodeChannelId, 754  
decodeChunkSize, 759, 782  
decodeFlexMessage, 759, 782  
decodeFlexSharedObject, 759, 783  
decodeHandshake, 783  
decodeHeader, 28, 123, 169, 783  
decodeHeaderSize, 754  
decodeInvoke, 759, 783  
decodeMessage, 784  
decodeNotify, 760, 784  
decodeNotifyOrInvoke, 784  
decodePacket, 784  
decodePing, 760, 785  
decodeRequest, 893  
decodeSharedObject, 760, 785  
decodeUnknown, 760, 785  
decodeVideoData, 761, 785  
deconfigureWebApp, 595  
decrement, 514  
deleteStream, 550, 550, 1074, 1074  
deleteStreamById, 537, 728  
dequeue, 1022, 1034  
deserialize, 181, 298, 458, 970  
destroy, 308, 323, 446, 570, 575, 579, 882  
disconnect, 259, 282, 308, 348, 351, 365, 400, 420, 425, 438, 487, 634, 639, 641, 703, 954  
dispatchEvent, 246, 254, 308, 453, 680, 954, 986, 1012  
dispose, 287, 690, 691, 778, 779  
disposeCached, 765  
dissociate, 643, 650  
docToString, 217  
docToString1, 217  
docToString2, 217  
doDecodeSharedObject, 785  
doEncodeSharedObject, 788  
doPost, 902  
doRelease, 800, 802, 816  
dropPacket, 1027, 1082  
dump, 793  
echoArray, 227, 232  
echoBoolean, 228, 232  
echoDate, 228, 232  
echoList, 228, 232  
echoNumber, 229, 233  
echoObject, 229, 233  
echoString, 229, 233  
echoXML, 229, 233  
encode, 661, 692, 779, 788  
encodeAudioData, 762, 788  
encodeBytesRead, 762, 788  
encodeChunkSize, 762, 789  
encodeFlexMessage, 789  
encodeFlexSharedObject, 762, 789  
encodeHeader, 789, 789  
encodeHeaderByte, 754  
encodeInvoke, 762, 790  
encodeMessage, 790

encodeNotify, 763, 790  
encodeNotifyOrInvoke, 790, 791  
encodePacket, 791  
encodePing, 763, 791  
encodeSharedObject, 763, 791  
encodeString, 52, 96  
encodeUnknown, 763, 791  
encodeVideoData, 764, 792  
endUpdate, 490, 954, 970, 986  
enforceAMF3, 88, 96  
equals, 260, 802, 814, 816, 836  
errorReceived, 670  
exceptionCaught, 102, 662, 747  
execute, 468, 915, 920  
executeScript, 923  
executeTask, 675  
ExecutorFilter, 101, 101, 101  
FCPublish, 366  
FCUnpublish, 366  
FileConsumer, 1091  
FilePersistence, 907, 907  
FileProvider, 1099  
FileStreamSource, 1021  
filterClose, 102  
filterNull, 235  
filterWrite, 102  
findApplication, 442  
findMethodsByNameAndNumParams, 940  
findMethodWithExactParameters, 949, 949  
findMethodWithListParameters, 949, 949  
findRoot, 442  
finishDecode, 778  
fireConsumerConnectionEvent, 608  
firePipeConnectionEvent, 608  
fireProviderConnectionEvent, 609  
FlexSharedObjectMessage, 957, 958  
FlexStreamSend, 810  
flushHeaders, 111, 118  
FLV, 118, 118, 118  
FLVReader, 122, 122, 122  
FLVWriter, 128, 128  
formatHTML, 896  
formatPath, 1105  
frameDuration, 166  
frameSize, 167  
generateErrorResult, 655  
generateFilename, 541, 541, 1017, 1018  
get, 8, 446, 570, 575, 580  
getActiveClients, 308, 323, 505  
getActiveConnections, 308, 323, 505  
getActiveListeners, 508, 970  
getActiveSubscopes, 309, 323, 505  
getActiveSubscribers, 502, 1012  
getAppendWriter, 22, 118, 165  
getApplication, 860  
getApplicationContext, 269, 292, 413, 570, 575, 580  
getApplicationLoader, 292, 344  
getArguments, 473, 934  
getAttribute, 236, 236, 298, 381, 381, 390, 390, 680, 680, 954, 970, 986, 986  
getAttributeNames, 236, 381, 391, 680, 987  
getAttributes, 236, 381, 391, 680, 987  
getAudio, 1039  
getAudioBucket, 1019, 1023, 1066  
getAudioCodecName, 539, 1087  
getBandwidth, 804, 824  
getBandwidthConfigure, 260, 393, 728, 1000  
getBaseHost, 1106  
getBasicScope, 309, 323, 420, 438  
getBasicScopeNames, 309, 324  
getBasicScopes, 246, 406, 680  
getBean, 270, 413  
getBitflags, 132  
getBitRate, 167  
getBody, 25, 132, 1095, 1097  
getBodySize, 26, 132  
getBoolAttribute, 236, 397, 680  
getBooleanProperty, 603, 613  
getBroadcastScope, 1074  
getBroadcastStream, 366, 518  
getBroadcastStreamNames, 366, 519, 1029, 1056  
getBuffer, 88, 96  
getBufferSize, 123  
getBufferType, 123  
getBWControllable, 1023, 1069  
getByteAttribute, 237, 397, 680  
getByteBuffer, 449, 573  
getByteProperty, 603, 613  
getBytes, 449, 573, 900  
getBytesRead, 28, 123, 169, 800  
getBytesReceived, 502, 1012  
getBytesSent, 504, 1050  
getBytesWritten, 31, 128  
getCachedStatusObjectAsByteArray, 863  
getCacheHit, 570, 575, 580  
getCacheId, 765

getCacheManagerEventListener, 575  
getCacheMiss, 570, 576, 580  
getCall, 814, 816  
getCallbacks, 471, 945  
getCalls, 696  
getCanSeekToEnd, 139, 147  
getCapacity, 1036, 1041  
getChannel, 728  
getChannelBandwidth, 394, 561  
getChannelId, 836  
getChannelInitialBurst, 394, 561  
getChildScope, 381  
getChildScopeNames, 382  
getClassLoader, 270, 309  
getClassName, 916  
getClient, 246, 406, 433, 680, 882  
getClientBufferDuration, 504, 1001  
getClientBytesRead, 246, 406, 680, 728  
getClientid, 845, 848  
getClientId, 882  
getClientList, 265  
getClientRegistry, 270, 414  
getClientResponse, 694  
getClientResult, 695  
 getClients, 265, 309, 324, 382, 421, 438  
getClientTTL, 366  
getCode, 848, 860  
getCodec, 808  
getCodecFactory, 703, 878  
getCodecInfo, 532, 1003  
getCodeSection, 287  
getColumnNames, 192  
getConnection, 433, 520, 711, 1001  
getConnectionLocal, 434  
getConnectionParams, 816  
getConnections, 260, 260, 310, 324, 400, 400, 421, 438  
getConnectionsIter, 382  
getConnector, 1106  
getConnectParams, 246, 407, 680  
getConsumer, 632  
getConsumerOutput, 1017, 1026  
getConsumers, 609, 619, 1007  
getContext, 277, 310, 324, 382, 421, 434, 439  
getContextPath, 310, 324, 421, 439  
getCoreService, 270, 414  
getCorrelationID, 603, 614  
getCount, 221  
getCreationTime, 260, 310, 401, 510, 971, 1012, 1051  
getCurrent, 514  
getCurrentItem, 526, 1051, 1061  
getCurrentItemIndex, 527, 1051, 1061  
getCurrentTimestamp, 512, 1012, 1051  
getCursor, 195  
getData, 26, 64, 103, 132, 167, 195, 490, 545, 795, 816, 826, 828, 840, 971, 987, 1033, 1039  
getDataBucket, 1019, 1024, 1067  
getDataOffset, 108  
getDataType, 26, 132, 545, 795, 798, 801, 802, 804, 809, 810, 811, 814, 817, 821, 825, 827, 828, 836, 958, 981  
getDebug, 821  
getDecoder, 774, 776  
getDecoderBufferAmount, 659  
getDepth, 254, 310, 325, 382, 395, 505  
getDescription, 849, 860  
getDeserializer, 126, 151  
getDetails, 845, 849  
getDiskExpiryThreadIntervalSeconds, 576  
getDiskStore, 576  
getDoubleAttribute, 237, 397, 680  
getDoubleProperty, 603, 614  
getDownstreamBandwidth, 412, 562  
getDroppedMessages, 247, 407, 680  
getDuration, 28, 123, 139, 147, 170  
getEmbedded, 1106  
getEnabled, 311, 325  
getEncoder, 774, 776  
getEncoding, 407, 681, 697, 728, 768  
getEndian, 64, 71, 75, 78, 82  
getEngine, 1106  
getEstimatedBufferFill, 504, 1051  
getEventListeners, 254, 454  
getEvents, 961, 981  
getException, 473, 934  
getExecutor, 102, 1051  
getExtension, 18, 24, 126, 172  
getFile, 29, 31, 123, 129, 151, 170  
getFileData, 124  
getFlagAudio, 108  
getFlagReserved01, 108  
getFlagReserved02, 108  
getFlagVideo, 108  
getFloatProperty, 603, 614  
getFrameRate, 139, 148  
getFrameType, 828

getGhostConnsCleanupPeriod, 367  
getGlobal, 336, 430  
getGlobalNames, 336, 430  
getGlobalScope, 271, 332, 414, 427, 894  
getGlobalScopes, 336, 430  
getHandler, 311, 325, 421, 439  
getHeader, 798, 811, 840  
getHeaderLength, 755  
getHeaders, 465, 681, 697  
getHeight, 139, 148  
getHost, 247, 407, 681, 1106, 1113  
getHostname, 707  
getId, 261, 401, 711  
getInstance, 570, 580, 793, 909  
getIntAttribute, 237, 398, 681  
getIntProperty, 603, 614  
getInvokedId, 728, 817  
getIoSession, 743  
getItem, 527, 1051, 1061  
getItemAt, 193  
getItemSize, 527, 1051, 1061  
getJobName, 919, 921  
getKey, 337, 959, 977  
getKeyframe, 557, 1085, 1086  
getKeyFrameData, 112, 118  
getLastModified, 298, 458, 971  
getLastPingTime, 408, 681, 729  
getLastReadChannel, 768  
getLastReadHeader, 769  
getLastReadPacket, 769  
getLastWriteChannel, 769  
getLastWriteHeader, 769  
getLastWritePacket, 769  
getLength, 193, 524, 564  
getLevel, 849, 860  
getListAttribute, 238, 398, 681  
getListeners, 367, 609, 971  
getLiveProviderInput, 1029, 1056  
getLongAttribute, 238, 398, 681  
getLongProperty, 603, 614  
getMapAttribute, 238, 398, 681  
getMappingStrategy, 271, 414  
getMappingTable, 337, 430  
getMax, 514  
getMaxClients, 311, 325, 506  
getMaxConnections, 311, 325, 506  
getMaxListeners, 508, 971  
getMaxSubscopes, 311, 326, 506  
getMaxSubscribers, 502, 1012  
getMemoryStoreEvictionPolicy, 576  
getMessage, 840  
getMessageCount, 1041  
getMessageID, 604, 615  
getMessageInput, 524, 564  
getMessageSize, 1041  
getMessageType, 604, 615  
getMetaCue, 139, 148  
getMetaData, 112, 119  
getMethodName, 916  
getMethodParams, 916  
getMinaDecoder, 776  
getMinaEncoder, 776  
getMode, 770  
getMsgInput, 564  
getMustUnderstand, 466, 685  
getName, 137, 145, 299, 383, 395, 449, 458, 466, 506, 508, 524, 533, 557, 565, 573, 685, 962, 971, 981, 987, 1003, 1085, 1086  
getNext, 156  
getNextAvailableChannelId, 729  
getNextFilter, 104  
getNextRTMPMessage, 1061  
getNumberAvailable, 193  
getObject, 451, 798, 981  
getObjectId, 911  
getObjectName, 911  
getObjectNames, 447, 460, 571, 576, 580, 912  
getObjectPath, 907, 912  
getObjectProperty, 604, 615  
getObjects, 447, 461, 571, 576, 581, 912  
getOffset, 29, 31, 124, 129, 170  
getOnDemandStream, 367, 523  
getParameters, 584  
getParamMap, 632  
getParamTypes, 916  
getParent, 255, 311, 326, 383, 396  
getParentBWControllable, 261, 393, 729, 1001  
getParentContext, 277  
getPath, 247, 255, 299, 312, 326, 383, 396, 408, 458, 506, 681, 972, 987  
getPendingCall, 729  
getPendingMessages, 247, 408, 681, 743, 870, 870, 876  
getPendingVideoMessages, 248, 537, 681, 729  
getPermissions, 261, 401  
getPersistanceStore, 271, 414

getPersistenceStore, 463  
getPollingDelay, 876  
getPrefix, 18, 24, 126, 172  
getPreviousTagSize, 26, 133  
getPreviosTagSize, 133  
getProvider, 516, 632, 1013, 1061  
getProviderInput, 1029, 1056  
getProviders, 609, 621, 1007  
getPublishedName, 503, 517, 1013, 1062  
getReadBytes, 248, 408, 682, 729, 743, 870  
getReadChunkSize, 770  
getReader, 22, 119, 165  
getReadMessages, 248, 408, 682  
getRealm, 1107  
getReason, 586  
getRed5ApplicationContext, 292  
getReference, 174  
getReferenceld, 176  
getRemoteAddress, 248, 409, 682  
getRemoteAddresses, 249, 409, 682, 901  
getRemotePort, 249, 409, 682  
getResolver, 151  
getResource, 271, 312, 326, 383  
getResources, 272, 312, 326, 383  
getResult, 471, 628, 916, 945  
getRootContext, 389, 591, 1102  
getRunning, 312, 327  
getSampleRate, 167  
getSaveFilename, 503, 517, 1013, 1062  
getScheduledJobNames, 367, 470, 919  
getScope, 249, 272, 313, 327, 384, 409, 422, 434, 439, 533, 682, 1003  
getScopeAttributes, 997, 998  
getScopeNames, 313, 327, 422, 439  
getScopePath, 697  
getScopeResolver, 272  
getScopes, 261, 401, 510, 511, 994, 994, 998, 998  
getScopeService, 442, 442, 443, 443  
getScopeStatisticsSO, 511, 995  
getScriptedObject, 924, 925, 928  
getScriptInterfaces, 924, 925, 928  
getScriptSourceLocator, 926, 928  
getSerializer, 126, 151  
getServer, 289, 313, 344, 416  
getServerStream, 567  
getService, 37  
getServiceHandler, 313, 327, 478, 954, 987  
getServiceHandlerNames, 314, 327, 478, 954, 987  
getServiceHandlerProvider, 479  
getServiceHandlers, 314, 314  
getServiceInvoker, 272, 415  
getServiceMethodName, 473, 934  
getServiceName, 474, 628, 934, 948  
getServiceParamMap, 629  
getServices, 23, 37  
getSessionId, 249, 410, 682  
getSetAttribute, 238, 399, 682  
getSharedObject, 368, 368, 500, 500, 703, 992, 992  
getSharedObjectName, 368, 500, 993  
getSharedObjects, 511, 995, 998  
getSharedObjectSecurity, 368, 497, 987  
getSharedObjectStatisticsSO, 511, 995  
getShortAttribute, 239, 399, 682  
getShortProperty, 604, 615  
getSignature, 109  
getSimpleDecoder, 659, 688, 774, 777, 890  
getSimpleEncoder, 660, 688, 775, 777, 890  
getSize, 525, 565, 802, 836  
getSource, 451, 798  
getSpeed, 1036  
getStart, 525, 565  
getState, 770  
getStatistics, 314, 422, 488, 519, 531, 987, 1013, 1051  
getStatus, 474, 934  
getStatusObject, 863  
getStore, 289, 299, 458, 972, 987  
getStreamableFile, 19, 24, 126, 172  
getStreamAwareHandler, 1004  
getStreamByChannelId, 729  
getStreamById, 538, 730  
getStreamId, 520, 707, 837, 1001  
getStreamIdForChannel, 730  
getStreamLength, 369  
getStreamListeners, 517, 1013  
getStreamPlaybackSecurity, 369, 548  
getStreamPublishSecurity, 369, 548  
getStreams, 730  
getString, 44, 44, 89, 156, 184  
getStringAttribute, 239, 399, 682  
getStringProperty, 604, 615  
getSubscriberStream, 369, 556  
getTarget, 629  
getTime, 137, 145

getTimeout, 287  
getTimeoutMillis, 287  
getTimer, 837  
getTimestamp, 26, 133, 546, 798, 811  
getTotal, 515  
getTotalBytes, 29, 124, 170  
getTotalChanges, 508, 972  
getTotalClients, 314, 328, 507  
getTotalConnections, 315, 328, 507  
getTotalDeletes, 508, 972  
getTotalListeners, 509, 972  
getTotalSends, 509, 972  
getTotalSubscopes, 315, 328, 507  
getTotalSubscribers, 503, 1013  
getType, 104, 133, 137, 145, 250, 299, 396, 410, 451, 459, 632, 682, 798, 959, 972, 978, 981, 987  
getUpstreamBandwidth, 412, 563  
getUpTime, 434  
getUsedStreamCount, 730  
getValue, 466, 686, 959, 978  
getValue1, 821  
getValue2, 804, 821  
getValue3, 821  
getValue4, 822  
getVersion, 109, 434, 491, 509, 962, 973, 981, 987  
getVideo, 1039  
getVideoBucket, 1019, 1024, 1067  
getVideoCodec, 539, 1081, 1087  
getVideoCodecFactory, 730  
getVideoCodecId, 140, 148  
getVideoCodecName, 540, 1088  
getVideoDataRate, 140, 148  
getVODProviderFile, 1030, 1057  
getVODProviderInput, 1030, 1057  
getWebAppContext, 595  
getWidth, 140, 148  
getWriteChunkSize, 770  
getWriter, 22, 119, 165  
getWrittenBytes, 250, 410, 682, 731, 744, 870  
getWrittenMessages, 250, 410, 683  
GroovyScriptFactory, 923, 923  
handleBadRequest, 882  
handleClose, 883  
handleEvent, 250, 255, 282, 315, 348, 453, 683  
handleIdle, 883  
handleOpen, 883  
handlePendingCallResult, 707  
handleRemotingPacket, 894  
handleRequest, 668, 668, 668, 669  
handleSend, 884  
HandshakeFailedException, 656  
hasAttribute, 239, 384, 391, 683, 988  
hasAudio, 540, 1088  
hasBroadcastStream, 369, 519  
hasChildScope, 315, 315, 328, 328, 384, 422, 422, 440, 440  
hasClient, 265, 403  
hasClients, 266  
hasContext, 316, 329  
hasDecodedObject, 659  
hasEventsWaiting, 638  
hasHandler, 316, 329, 423, 440  
hashCode, 262, 803  
hasKeyFrameData, 112, 119  
hasMetaData, 112, 119  
hasMore, 1022, 1034  
hasMoreItems, 527, 1052, 1062  
hasMoreProperties, 45  
hasMoreTags, 29, 124, 170  
hasNext, 256  
hasOnDemandStream, 370, 523  
hasParent, 255, 316, 329, 384, 396  
hasPermission, 262, 402  
hasReference, 176  
hasSharedObject, 370, 500, 993  
hasSource, 451, 798  
hasVideo, 29, 124, 170, 540, 1034, 1088, 1099  
hexStringToByteArray, 207  
includeStacktrace, 584  
increment, 515  
init, 278, 316, 329, 688, 775, 777, 866, 879, 884, 888, 890, 894, 897, 902, 1107, 1113  
initialize, 251, 410, 683, 863  
Input, 44, 88, 88  
Input.ClassReference, 92  
invoke, 475, 475, 476, 476, 476, 476, 479, 480, 703, 703, 731, 731, 731, 731, 731, 731, 947, 947  
Invoke, 813, 813, 813  
invokeCall, 739  
invokeMethod, 673, 673  
invokeOnAllConnections, 481, 481, 482, 482  
invokeOnClient, 482, 483  
invokeOnConnection, 483, 483, 484, 484

isAcquired, 488, 973, 988  
isAncestor, 443  
isApp, 444  
isBasicType, 180  
isCached, 449, 573  
isChannelUsed, 732  
isClosing, 870  
isComplexType, 180  
isConnected, 251, 411, 487, 683, 744, 954  
isConnectionAllowed, 495, 988  
isCreationAllowed, 496  
isCustom, 52, 160, 188  
isCustomType, 180  
isDebug, 770  
isDeleteAllowed, 496, 988  
isDisposable, 809  
isEmpty, 962, 981  
isEnabled, 316  
isFullyPopulated, 193  
isGlobal, 444  
isLocked, 491, 954, 988  
isPaused, 521, 554, 1052  
isPersistent, 300, 459, 962, 973, 981  
isPersistentObject, 491, 509, 973, 988  
isPlaybackAllowed, 546, 560  
isPlaying, 521  
isProtected, 167  
isPublishAllowed, 547, 560  
isPullMode, 1044  
isRandom, 527, 1052, 1062  
isRelative, 1078  
isRepeat, 528, 1052, 1062  
isRewind, 528, 1052, 1062  
isRoom, 444  
isRoot, 444  
isRunning, 317  
isSendAllowed, 496, 988  
isShuttingDown, 344  
isStereo, 168  
isStopped, 522  
isSuccess, 474, 934  
isTimerRelative, 837  
isWriteAllowed, 497, 989  
iterateScopeNameList, 262  
iterator, 255, 317  
JettyApplicationContext, 590  
JettyApplicationLoader, 591  
join, 282, 348, 351, 370, 425  
killGhostConnections, 370  
leave, 282, 348, 352, 371, 426  
length, 64  
load, 461, 461, 907, 907, 912, 912  
loadApplication, 389, 591, 1102  
loadContext, 278  
loadKeyFrameMeta, 20, 21  
loadStatusObjects, 863  
lock, 491, 955, 989  
lookupClient, 266, 403  
lookupConnections, 317, 423, 440  
lookupContext, 1019, 1024, 1067  
lookupGlobal, 337, 431  
lookupMRTMPConnection, 642  
lookupProviderInput, 1030, 1057  
lookupScopeHandler, 273, 415  
lookupService, 273, 415  
main, 207, 340, 341  
makeDefaultConnectionParams, 704  
mapResourcePrefix, 294, 417  
mapScopeHandlerName, 295, 417  
mapServiceName, 295, 417  
measureBandwidth, 371, 371  
messageDropped, 732  
messageReceived, 102, 224, 662, 665, 666, 708, 718, 732, 747, 872, 878, 903  
messageSent, 102, 224, 708, 719, 732, 747  
MetaCue, 145  
MetaData, 147  
MetaService, 151  
MethodNotFoundException, 942, 942  
modified, 300, 909  
moveToNext, 1062  
moveToPrevious, 1062  
MP3, 165  
MP3Header, 166  
MP3Reader, 169  
NetworkDumpFilter, 664  
newBroadcastStream, 538, 732  
newClient, 266, 403  
newInstance, 45, 89, 940  
newPlaylistSubscriberStream, 538, 732  
newSingleItemSubscriberStream, 538, 732  
next, 256  
nextId, 267  
nextItem, 528, 530, 1052, 1062, 1071  
notify, 476, 477, 477, 477, 733, 733, 733, 733  
Notify, 815, 815, 816  
notifyBroadcastClose, 1062  
notifyBroadcastStart, 1062

notifyClear, 955  
notifyClose, 461, 907, 909, 913  
notifyClosed, 884  
notifyConnect, 955  
notifyConnected, 337, 455  
notifyDelete, 955  
notifyDisconnect, 955  
notifyDisconnected, 338, 455  
notifyEvent, 251, 255, 453, 683  
notifyItemPause, 1052  
notifyItemPlay, 1052  
notifyItemResume, 1052  
notifyItemSeek, 1053  
notifyItemStop, 1053  
notifyModified, 973  
notifyOnAllConnections, 484, 485  
notifyOnClient, 485  
notifyOnConnection, 485, 485  
notifyScopeCreated, 338, 456  
notifyScopeRemoved, 338, 456  
notifySendMessage, 955  
notifySubscriberClose, 1053  
notifySubscriberStart, 1053  
notifyUpdate, 955, 956  
ObjectProxy, 8, 8  
offer, 447, 571, 577, 581  
onBroadcastStreamStart, 543  
onBroadcastStreamSubscribe, 543  
onBroadcastStreamUnsubscribe, 543  
onChunkSize, 704, 708, 739  
onInactive, 733, 744, 876  
onInvoke, 704, 708, 739  
onItemEnd, 1053, 1063  
onOnDemandStreamConnect, 543  
onOnDemandStreamDisconnect, 543  
onOOBControlMessage, 619, 1014, 1044, 1063, 1090, 1091, 1093, 1098, 1100  
onPing, 704, 709, 715, 740  
onPipeConnectionEvent, 624, 1007, 1014, 1044, 1063, 1079, 1090, 1091, 1094, 1098, 1100  
onRecordStreamStart, 544  
onRecordStreamStop, 544  
onSharedObject, 704, 709, 740  
onSharedObjectClear, 493  
onSharedObjectConnect, 493  
onSharedObjectDelete, 493  
onSharedObjectDisconnect, 494  
onSharedObjectSend, 494  
onSharedObjectUpdate, 494, 494, 495  
onStreamBytesRead, 709  
onStreamPublishStart, 544  
onStreamPublishStop, 544  
Output, 52, 95  
OutputStream, 1039  
Packet, 839, 839  
packetReceived, 545  
pause, 522, 531, 551, 554, 1044, 1053, 1063, 1074, 1075  
peekMessage, 1041  
PendingCall, 944, 944, 944  
PersistableAttributeStore, 297  
pickupEvents, 638  
ping, 411, 683, 733, 733  
Ping, 820  
pingReceived, 733  
PipeConnectionEvent, 631  
play, 522, 522, 551, 551, 551, 552, 554, 1044, 1045, 1053, 1063, 1075, 1075, 1075, 1076, 1076  
PlayBuffer, 1040  
PlaylistSubscriberStream, 1050  
position, 30, 64, 64, 124, 171  
postProcessExtension, 181  
prepareFilename, 19, 25  
prepareIO, 64  
preProcessExtension, 197  
prettyPrintHex, 207, 208, 208  
previousId, 267  
previousItem, 528, 530, 1054, 1063, 1071  
processEvent, 102  
processHeaders, 674  
ProtocolException, 656, 656  
ProxyFilter, 666  
publish, 552, 552, 552, 1076, 1076, 1076  
pullAndPush, 1045  
pullMessage, 620, 620, 626, 626, 627, 627, 1007, 1008, 1100, 1100  
pushMessage, 621, 625, 626, 627, 1008, 1014, 1045, 1063, 1079, 1090, 1091  
put, 8, 201, 447, 571, 577, 581  
putInteger, 96  
putMessage, 1041  
putServerStream, 567  
putString, 53, 53, 96, 96, 160  
QuartzSchedulingService, 918  
raiseOriginalException, 341  
RamPersistence, 911, 911

rawBufferReceived, 747  
rawWrite, 734, 744, 871  
readArray, 45, 89, 156, 184  
readBean, 45  
readBoolean, 46, 65, 71, 78, 89, 156, 184  
readByte, 65, 71, 78  
readByteArray, 46, 89, 156, 184  
readBytes, 65, 65, 65, 71, 71, 71, 78, 79, 79  
readCustom, 46, 90, 156, 185  
readDataType, 46, 46, 90, 157, 185  
readDate, 47, 90, 157, 185  
readDouble, 65, 72, 79  
readerFromNearestKeyFrame, 112, 119  
readExternal, 6, 9, 86  
readFloat, 65, 72, 79  
readHeaders, 690  
readInt, 66, 72, 80  
readKeyValues, 47, 47, 157, 185  
readMap, 47, 90, 157, 185  
readMediumInt, 201, 214, 755  
readMediumInt2, 214  
readMediumIntOld, 755  
readMetaCue, 142, 151  
readMetaData, 142, 152  
readMultiByte, 66, 72, 80  
readNull, 48, 91, 157, 185  
readNumber, 48, 91, 157, 186  
readObject, 48, 66, 72, 80, 91, 157, 186  
readPropertyName, 48  
readReference, 48, 91, 158, 186  
readReverseInt, 214, 755  
readReverseIntOld, 755  
readShort, 66, 72, 80  
readSimpleObject, 49  
readString, 49, 91, 158, 186  
readTag, 30, 125, 171  
readUnsignedByte, 66, 72, 80  
readUnsignedInt, 66, 73, 81  
readUnsignedMediumInt, 202, 214, 756  
readUnsignedMediumIntOld, 756  
readUnsignedShort, 66, 73, 81  
readUTF, 67, 73, 81  
readUTFBytes, 67, 73, 81  
readXML, 49, 92, 158, 186  
realClose, 871, 876  
receiveAudio, 553, 555, 1045, 1046, 1054, 1076  
receivedBytesRead, 734  
receiveVideo, 553, 555, 1046, 1046, 1054, 1076  
RecordSet, 192  
RecordSetPage, 195  
Red5, 433, 433  
refreshHeaders, 113, 119  
register, 262, 290, 345, 417, 974  
registerBasicScope, 251  
registerBroadcastStream, 1030, 1057  
registerBWControllable, 1020, 1024, 1067  
registerCallback, 471, 945  
registerConnection, 642  
registerDeferredResult, 734  
registerGlobal, 338, 431  
registerPendingCall, 734  
registerServiceHandler, 317, 329, 478, 492, 956, 956, 989, 989  
registerSharedObjectSecurity, 371, 498, 989  
registerStream, 734  
registerStreamPlaybackSecurity, 371, 548  
registerStreamPublishSecurity, 371, 548  
rejectClient, 372, 372  
release, 488, 794, 798, 811, 974, 989  
releaseInternal, 795, 799, 801, 803, 804, 817, 822, 825, 827, 829, 982  
releaseStream, 553, 1077  
RemotingCall, 694  
RemotingClient, 672, 672, 672  
RemotingConnection, 678  
RemotingHeader, 685  
RemotingPacket, 696  
remove, 9, 256, 448, 448, 461, 462, 571, 571, 577, 577, 581, 581, 907, 907, 913, 913  
removeAlias, 1113  
removeAllItems, 528, 1054, 1064  
removeAttribute, 239, 300, 384, 391, 683, 956, 974, 990  
removeAttributes, 240, 300, 385, 392, 683, 956, 974, 990  
removeChildScope, 283, 318, 348, 423, 426, 440  
removeClient, 226, 267  
removeContext, 292, 593, 1107  
removeEventListener, 256, 454, 990  
removeHeader, 465, 674, 683  
removeItem, 528, 1054, 1064  
removeListener, 338, 338, 372, 431, 431  
removeMapping, 339, 431

removePipeConnectionListener, 609, 623, 1008  
removeRed5ApplicationContext, 293  
removeScheduledJob, 372, 470, 919  
removeServerStream, 567  
removeSharedObjectListener, 491, 956, 990  
removeStreamListener, 517, 1014  
removeTokenBucket, 1038  
removeValve, 1113  
requiresConfigInterface, 924, 926, 928  
reserveStreamId, 538, 735  
reset, 49, 49, 92, 557, 916, 1028, 1028, 1036, 1037, 1046, 1078, 1082, 1083, 1085, 1087  
resetBuckets, 1020, 1025, 1068  
resetCredentials, 674  
resolve, 144, 154  
Resolver, 153  
resolveScope, 273, 274, 274, 332, 333, 415, 416, 427, 428, 445  
resolveService, 936, 941, 941, 946  
resolvesToAbsolutePath, 541, 1018  
ResourceExistException, 558  
ResourceNotFoundException, 559  
resultReceived, 472, 671, 1080  
resume, 522, 555, 1046, 1054  
retain, 794, 799, 812  
retrievePendingCall, 735  
returnAttributeValue, 974  
returnConnection, 230  
returnDistinctObjects, 230  
returnError, 669, 669, 974  
returnMessage, 884, 885, 885  
returnPendingMessages, 885  
returnSameObjects, 230  
RhinoScriptFactory, 928  
rollbackRequest, 1068  
roomConnect, 355, 373  
roomDisconnect, 355, 373  
roomJoin, 355  
roomLeave, 355, 373  
roomStart, 356, 373  
roomStop, 356, 374  
RTMP, 767  
RTMPClient, 720  
RTMPConnection, 725  
RTMPMinaConnection, 743  
RTMPProtocolDecoder, 781  
run, 909, 916, 1094  
RuntimeStatusObject, 844, 844, 844  
save, 462, 908, 913  
saveAs, 517, 1014, 1064  
saveKeyFrameMeta, 21, 21  
saveObject, 908  
scheduleGhostConnectionsCleanup, 374  
scheduleNextMessage, 1064  
Scope, 306, 306  
ScopeHandlerNotFoundException, 586  
ScopeNotFoundException, 587  
ScopeShuttingDownException, 588  
ScreenVideo, 1084  
searchNextFrame, 171  
seek, 522, 531, 553, 555, 1022, 1032, 1032, 1047, 1054, 1064, 1077, 1100  
send, 636, 637  
sendMessage, 491, 956, 975, 990  
sendNetStreamStatus, 1077  
sendOOBControlMessage, 610, 610, 620, 622, 1008, 1009  
sendPacket, 1028, 1083  
sendResponse, 894  
sendStatus, 711  
sendUpdates, 975  
serialize, 183, 193, 197, 300, 459, 849, 860, 975  
serializeAsXML, 203  
serializeStatusObject, 864  
ServerBW, 824  
ServerStream, 1060  
service, 886, 894, 896, 897, 897, 898, 902  
serviceAMF, 895  
serviceCall, 283, 349, 426  
ServiceNotFoundException, 948  
sessionClosed, 103, 665, 666, 748  
sessionCreated, 103, 663, 748  
sessionId, 103  
sessionOpened, 103, 663, 748  
setApplication, 861  
setApplicationContext, 274, 278, 293, 339, 572, 577, 581, 710, 748, 880  
setApplicationLoader, 293  
setArguments, 934  
setAttribute, 240, 301, 385, 392, 683, 956, 975, 990  
setAttributes, 240, 240, 301, 301, 385, 385, 392, 392, 683, 684, 956, 957, 975, 976, 990, 990  
setAutoStart, 318, 330  
setBandwidth, 805, 825

setBandwidthConfigure, 262, 393, 735, 1001, 1054  
setBaseHost, 1107  
setBitDigits, 208, 208  
setBitflags, 133  
setBody, 27, 134, 1095, 1097  
setBodySize, 27, 134  
setBooleanProperty, 604, 616  
setBufferCheckInterval, 1054  
setBufferSize, 125  
setBufferType, 125  
setByteProperty, 604, 616  
setByteSeparator, 209  
setBytesRead, 801  
setCache, 113, 120  
setCacheConfigs, 577  
setCached, 449, 573  
setCacheManagerEventListener, 578  
setCall, 817  
setCanCallService, 349  
setCanConnect, 349  
setCanSeekToEnd, 140, 148  
setCanStart, 349  
setCapacity, 1042  
setChannel, 713  
setChannelId, 837  
setChildLoadPath, 318, 330  
setClassName, 917  
setClientBufferDuration, 521, 1002  
setClientCallback, 695  
setClientid, 845, 849  
setClientRegistry, 274  
setClientTTL, 374  
setCode, 850, 861  
setCodecFactory, 663, 704, 748, 878  
setCodecInfo, 1004  
setCodecs, 1081  
setCodeSection, 287  
setConnection, 1002  
setConnectionLocal, 435  
setConnectionParams, 817  
setConnectionProperties, 1107  
setConnector, 1108  
setConnectors, 1108  
setConsumer, 633  
setContext, 318  
setContextPath, 275, 345  
setContexts, 1108, 1113  
setContextsConfig, 278  
setCorrelationID, 605, 616  
setCredentials, 674  
setData, 134, 817, 841  
setDataOffset, 109  
setDataType, 27, 134, 837  
setDebug, 771, 822  
setDefaultApp, 295  
setDeliveryMode, 194, 194, 194  
setDepth, 318, 330  
setDescription, 850  
setDescription, 850, 861  
setDeserializer, 114, 127, 152, 688, 690, 775, 786, 890  
setDetails, 845, 850  
setDiskExpiryThreadIntervalSeconds, 578  
setDiskStore, 578  
setDoubleProperty, 605, 616  
setDumpTo, 663  
setDuration, 140, 149  
setEmbedded, 1108  
setEnabled, 319, 330  
setEncoding, 771  
setEndian, 67, 73, 75, 81, 83  
setEndpoints, 670  
setException, 474, 935  
setExecutor, 1054  
setExtension, 908  
setFile, 152  
setFlagAudio, 109  
setFlagReserved01, 109  
setFlagReserved02, 110  
setFlagVideo, 110  
setFloatProperty, 605, 617  
setFLV, 129  
setForward, 663  
setFrameRate, 141, 149  
setGenerateMetadata, 127  
setGhostConnsCleanupPeriod, 374  
setGlobalScope, 333, 345, 998  
setHandler, 319, 748, 886  
setHandshake, 771  
setHasAudio, 1088  
setHasVideo, 1088  
setHeader, 799, 812  
setHeight, 141, 149  
setHost, 1108  
setHosts, 1109  
setInStream, 143, 152  
setIntProperty, 605, 617

setInvokedId, 713, 818  
setIoSession, 744  
setIsPersistent, 982  
setItem, 528, 1055, 1064  
setJoin, 349  
setKeyFrameData, 113, 120  
setLastReadHeader, 771  
setLastReadPacket, 771  
setLastWriteHeader, 772  
setLastWritePacket, 772  
setLength, 565  
setLevel, 850, 861  
setListeners, 610  
setLocalContext, 221  
setLongProperty, 605, 617  
setMappings, 888  
setMappingStrategy, 275  
setMaxEntries, 448, 572, 578, 581  
setMaxHandshakeTimeout, 735  
setMaxInactivity, 735  
setMemoryStoreEvictionPolicy, 578  
setMessage, 841  
setMessageID, 605, 617  
setMessageType, 606, 617  
setMetaCue, 141, 149  
setMetaData, 113, 120  
setMetaService, 113, 120  
setMethodName, 917  
setMethodParams, 917  
setMinaDecoder, 777  
setMinaEncoder, 777  
setMode, 748  
setMsgInput, 565  
setName, 137, 146, 301, 319, 330, 345, 450, 459, 565, 573, 976, 982, 990, 1004  
setObjectProperty, 606, 618  
setOffset, 129  
setOutStream, 143, 152  
setPacket, 684  
setParamMap, 633  
setParamTypes, 917  
setParent, 319, 345  
setParentContext, 279  
setPath, 301, 459, 908, 976, 990  
setPermissions, 263, 402  
setPersistanceStore, 275  
setPersistenceClass, 290, 320, 331  
setPersistenceClassName, 993  
setPersistent, 302, 460, 976  
setPingInterval, 735  
setPlayItem, 532  
setPlaylistController, 529, 1055, 1064  
setPreviousTagSize, 27, 134  
setPrevioosTagSize, 134  
setProvider, 633  
setPublishedName, 518, 1015, 1064  
setRandom, 529, 1055, 1064  
setReadChunkSize, 772  
setRealm, 1109  
setRed5ApplicationContext, 293  
setRemotingClient, 194  
setRepeat, 529, 1055, 1064  
setResolver, 152  
setResourceLoader, 663  
setResult, 472, 629, 713, 917, 945  
setRewind, 529, 1055, 1065  
setSchedulingService, 736  
setScope, 385, 424, 1004  
setScopePath, 697  
setScopeResolver, 275  
setSerializer, 114, 127, 152, 688, 692, 775, 792, 864, 890  
setServer, 290, 345, 740, 866, 880, 888  
setServiceCall, 713  
setServiceInvoker, 275, 670  
setServiceMethodName, 935  
setServiceName, 629, 935  
setServiceParamMap, 629  
setServiceProvider, 705  
setServiceResolvers, 947  
setServices, 37  
setServlet, 876  
setServletContext, 346  
setServletRequest, 876  
setShortProperty, 606, 618  
setSignature, 110  
setSize, 566, 803, 838  
setSource, 799, 812  
setStart, 566, 1100  
setState, 772  
setStatus, 474, 935  
setStatusObjectService, 740  
setStore, 302, 460, 976  
setStreamEventDispatcher, 705  
setStreamId, 838, 1002  
setStringProperty, 606, 618  
setTarget, 630  
setThreadPool, 674

setTime, 137, 146  
setTimeout, 287  
setTimer, 838  
setTimerRelative, 838  
setTimestamp, 27, 134, 799, 812  
setType, 135, 138, 146, 633  
setTypeFlags, 110  
setUnderrunTrigger, 1055  
setup, 736  
setupClassLoader, 786  
setUpstreamBandwidth, 412, 563  
setValue1, 822  
setValue2, 805, 822  
setValue3, 822  
setValue4, 823  
setValves, 1109  
setVersion, 110, 982  
setVideoCodec, 1088  
setVideoCodecId, 141, 149  
setVideoDataRate, 141, 149  
setVirtualHosts, 346  
setWebAppContext, 595  
setWebappFolder, 293  
setWidth, 141, 150  
setWithByteSeparator, 209  
setWriteChunkSize, 773  
SharedObject, 966, 966, 966, 967  
SharedObjectEvent, 977  
SharedObjectMessage, 980, 980  
SharedObjectScope, 985  
shutdown, 593, 597, 909, 1109  
skipData, 886  
skipEndObject, 50  
skipPropertySeparator, 50  
SorensonVideo, 1086  
start, 283, 320, 331, 349, 352, 374, 426, 533, 1015, 1047, 1055, 1065, 1094  
startBroadcastVOD, 1065  
startConnector, 705  
startDecoding, 659  
startPublishing, 520, 1015  
startRoundTripMeasurement, 736  
startUp, 230, 233  
startWaitForHandshake, 736  
startWebApplication, 1114  
Status, 847, 847, 847  
StatusObject, 859  
stop, 283, 320, 331, 350, 352, 375, 388, 427, 523, 533, 555, 590, 1015, 1047, 1055, 1065, 1101  
stopRecording, 1015  
storeReference, 174, 174, 176  
streamBroadcastClose, 375, 534  
streamBroadcastStart, 375, 534  
streamPlaylistItemPlay, 375, 534  
streamPlaylistItemStop, 376, 535  
streamPlaylistVODItemPause, 376, 535  
streamPlaylistVODItemResume, 376, 535  
streamPlaylistVODItemSeek, 377, 536  
streamPublishStart, 377, 536  
streamRecordStart, 377, 536  
streamSubscriberClose, 378, 536  
streamSubscriberStart, 378, 536  
StreamTracker, 1078  
stringToDoc, 218  
stringToHexString, 209  
subscribe, 610, 611, 620, 622, 626, 626, 627, 627, 1009, 1009  
supportsDataType, 53, 96, 160  
Tag, 131, 131  
takeMessage, 1042  
Test, 637  
toBinaryString, 209, 209, 210, 210, 210  
toByte, 842  
toByteArray, 210, 210, 211  
toHexString, 211, 211, 211, 212, 212, 212  
TomcatApplicationContext, 1101  
TomcatApplicationLoader, 1102  
toString, 9, 11, 67, 105, 110, 135, 146, 150, 212, 212, 215, 263, 320, 339, 684, 736, 795, 801, 803, 805, 814, 818, 823, 825, 827, 829, 838, 842, 851, 861, 935, 978, 982, 990  
toStringValue, 180  
toType, 842  
trimClassName, 597  
uncompress, 67  
uninit, 320, 1114  
Unknown, 826  
unloadContext, 279  
unlock, 492, 957, 990  
unregister, 263, 346, 976  
unregisterBasicScope, 251  
unregisterBroadcastStream, 1031, 1057  
unregisterBWControllable, 1020, 1025, 1068  
unregisterCallback, 472, 945  
unregisterConnection, 643

unregisterDeferredResult, 737  
unregisterMBean, 597  
unregisterServiceHandler, 320, 331, 478, 492, 957, 991  
unregisterSharedObjectSecurity, 378, 498, 991  
unregisterStream, 737  
unregisterStreamPlaybackSecurity, 378, 548  
unregisterStreamPublishSecurity, 379, 549  
unreserveStreamId, 539, 737  
unsubscribe, 611, 611, 621, 622, 1009, 1010  
updateBandwidthConfigure, 1047  
updateBWConfigure, 1021, 1025, 1068  
updateBytesRead, 737  
updateMBeanAttribute, 598, 598  
updateScopeStatistics, 512, 996  
updateSharedObjectStatistics, 512, 996  
validateHandshakeReply, 773  
VideoData, 828, 828  
VideoFrameDropper, 1082  
wasSent, 713  
WebappClassLoader, 1116, 1116  
wrap, 288  
write, 143, 153, 640, 641, 712, 737, 744, 871  
writeArbitraryObject, 53, 96  
writeArray, 53, 53, 54, 97, 97, 97, 160, 161, 161, 188, 188, 188  
writeArrayType, 197  
writeBasic, 197  
writeBoolean, 54, 67, 75, 83, 97, 161, 189  
writeByte, 67, 75, 83  
writeByteArray, 54, 97, 161, 189  
writeBytes, 68, 68, 68, 75, 75, 75, 83, 83, 84  
writeComplex, 198  
writeCustom, 54, 161, 189  
writeCustomType, 198  
writeDate, 54, 97, 161, 189  
writeDocument, 198  
writeDouble, 68, 76, 84  
writeExternal, 6, 9, 86  
writeFloat, 68, 76, 84  
writeHeader, 31, 129  
writeInt, 68, 76, 84  
writeIterator, 198  
writeList, 199  
writeListType, 199  
writeMap, 54, 54, 97, 98, 161, 162, 189, 190  
writeMediumInt, 202, 215, 756  
writeMetaCue, 143, 153  
writeMetaData, 143, 153  
writeMultiByte, 68, 76, 84  
writeNull, 55, 98, 162  
writeNumber, 55, 98, 162, 190  
writeObject, 55, 55, 69, 76, 85, 98, 98, 162, 162, 190, 190  
writeObjectType, 199  
writeRecordSet, 55, 98, 162, 190  
writeReference, 55, 162, 191  
writeReverseInt, 215, 756  
writeReverseIntOld, 756  
writerFromNearestKeyFrame, 114, 120  
writeShort, 69, 76, 85  
writeStream, 31, 129  
writeString, 55, 99, 163, 191  
writeTag, 32, 32, 130, 130  
writeUnsignedInt, 69, 76, 85  
writeUTF, 69, 77, 85  
writeUTFBytes, 69, 77, 85  
writeXML, 56, 99, 163, 191  
writeXMLType, 199  
writingMessage, 737  
written, 1055  
XmlRpcScopeStatistics, 997, 997  
MidiPlayer, 635  
MIN\_INTEGER\_VALUE, 58  
Mock, 158  
mode, 747  
MODE\_CLIENT, 767  
MODE\_SERVER, 767  
modified, 300, 909, 968  
moveToNext, 1062  
moveToPrevious, 1062  
MP3, 165, 165  
MP3Header, 165, 166  
MP3Reader, 168, 169  
MP3Service, 171  
MRTMPClient, 643  
MRTMPCodecFactory, 652  
MRTMPEdgeConnection, 644  
MRTMPMinTransport, 644  
MRTMPOriginConnection, 645  
MRTMPPacket, 646  
MRTMPPacket.Body, 646  
MRTMPPacket.Header, 647  
MRTMPPacket.RTMPBody, 647  
MRTMPPacket.RTMPHeader, 648  
MRTMPProtocolDecoder, 652  
MRTMPProtocolEncoder, 652

MulticastEventProcessor, 764  
 MultiThreadedApplicationAdapter, 356

**N**

name, 297, 666, 685, 968  
 NAVIGATION, 136  
 NC\_CALL\_BADVERSION, 853  
 NC\_CALL\_FAILED, 853  
 NC\_CONNECT\_APPSHUTDOWN, 854  
 NC\_CONNECT\_CLOSED, 854  
 NC\_CONNECT\_FAILED, 854  
 NC\_CONNECT\_INVALID\_APPLICATION, 854  
 NC\_CONNECT\_REJECTED, 854  
 NC\_CONNECT\_SUCCESS, 854  
 NetworkDumpFilter, 663, 664  
 newBroadcastStream, 538, 732  
 newClient, 266, 403  
 newInstance, 45, 89, 940  
 newPlaylistSubscriberStream, 538, 732  
 newSingleItemSubscriberStream, 538, 732  
 next, 256  
 nextId, 267  
 nextItem, 528, 530, 1052, 1062, 1071  
 NoCacheImpl, 578  
 noPendingMessages, 875  
 NotAllowedException, 942  
 notify, 476, 477, 477, 477, 733, 733, 733, 733  
 Notify, 814, 815, 815, 816  
 notifyBroadcastClose, 1062  
 notifyBroadcastStart, 1062  
 notifyClear, 955  
 notifyClose, 461, 907, 909, 913  
 notifyClosed, 884  
 notifyConnect, 955  
 notifyConnected, 337, 455  
 notifyDelete, 955  
 notifyDisconnect, 955  
 notifyDisconnected, 338, 455  
 notifyEvent, 251, 255, 453, 683  
 notifyItemPause, 1052  
 notifyItemPlay, 1052  
 notifyItemResume, 1052  
 notifyItemSeek, 1053  
 notifyItemStop, 1053  
 notifyMessages, 869  
 notifyModified, 973  
 notifyOnAllConnections, 484, 485  
 notifyOnClient, 485  
 notifyOnConnection, 485, 485

notifyScopeCreated, 338, 456  
 notifyScopeRemoved, 338, 456  
 notifySendMessage, 955  
 notifySubscriberClose, 1053  
 notifySubscriberStart, 1053  
 notifyUpdate, 955, 956  
 NS\_CLEAR\_FAILED, 854  
 NS\_CLEAR\_SUCCESS, 854  
 NS\_DATA\_START, 855  
 NS\_FAILED, 855  
 NS\_INVALID\_ARGUMENT, 855  
 NS\_PAUSE\_NOTIFY, 855  
 NS\_PLAY\_COMPLETE, 855  
 NS\_PLAY\_FAILED, 855  
 NS\_PLAY\_FILE\_STRUCTURE\_INVALID, 855  
 NS\_PLAY\_INSUFFICIENT\_BW, 855  
 NS\_PLAY\_NO\_SUPPORTED\_TRACK\_FOUND, 855  
 NS\_PLAY\_PUBLISHNOTIFY, 856  
 NS\_PLAY\_RESET, 856  
 NS\_PLAY\_START, 856  
 NS\_PLAY\_STOP, 856  
 NS\_PLAY\_STREAMNOTFOUND, 856  
 NS\_PLAY\_SWITCH, 856  
 NS\_PLAY\_UNPUBLISHNOTIFY, 856  
 NS\_PUBLISH\_BADNAME, 856  
 NS\_PUBLISH\_START, 857  
 NS\_RECORD\_FAILED, 857  
 NS\_RECORD\_NOACCESS, 857  
 NS\_RECORD\_START, 857  
 NS\_RECORD\_STOP, 857  
 NS\_SEEK\_FAILED, 857  
 NS\_SEEK\_NOTIFY, 857  
 NS\_UNPAUSE\_NOTIFY, 857  
 NS\_UNPUBLISHED\_SUCCESS, 857

**O**

object, 797  
 ObjectMap, 215  
 ObjectProxy, 7, 8, 8  
 objects, 911  
 offer, 447, 571, 577, 581  
 oName, 306, 726, 750  
 onBroadcastStreamStart, 543  
 onBroadcastStreamSubscribe, 543  
 onBroadcastStreamUnsubscribe, 543  
 onChunkSize, 704, 708, 739  
 onInactive, 733, 744, 876  
 onInvoke, 704, 708, 739

onItemEnd, 1053, 1063  
 onOnDemandStreamConnect, 543  
 onOnDemandStreamDisconnect, 543  
 onOOBControlMessage, 619, 1014, 1044, 1063, 1090, 1091, 1093, 1098, 1100  
 onPing, 704, 709, 715, 740  
 onPipeConnectionEvent, 624, 1007, 1014, 1044, 1063, 1079, 1090, 1091, 1094, 1098, 1100  
 onRecordStreamStart, 544  
 onRecordStreamStop, 544  
 onSharedObject, 704, 709, 740  
 onSharedObjectClear, 493  
 onSharedObjectConnect, 493  
 onSharedObjectDelete, 493  
 onSharedObjectDisconnect, 494  
 onSharedObjectSend, 494  
 onSharedObjectUpdate, 494, 494, 495  
 onStreamBytesRead, 709  
 onStreamPublishStart, 544  
 onStreamPublishStop, 544  
 OOBControlMessage, 628  
 OPENED, 105  
 operation, 13, 16  
 OPERATION\_AUTHENTICATION, 14  
 OPERATION\_PING, 14  
 OPERATION\_POLL, 15  
 OPERATION\_REGISTER, 15  
 OperationNotSupportedException, 557  
 OPT\_REFERENCE, 180  
 OriginMRTMPHandler, 648  
 Output, 50, 52, 94, 95, 159, 186  
 OutputStream, 1038, 1039  
 ownerMessage, 968

**P**

packet, 678  
 Packet, 838, 839, 839  
 packetReceived, 545  
 params, 244  
 parent, 254  
 parentContext, 277  
 path, 244, 298, 968  
 pause, 522, 531, 551, 554, 1044, 1053, 1063, 1074, 1075  
 peekMessage, 1041  
 PendingCall, 943, 944, 944, 944  
 pendingCalls, 726  
 pendingMessages, 869  
 permission, 2

PERMISSIONS, 259  
 PersistableAttributeStore, 295, 297  
 PERSISTENCE\_NO\_NAME, 911  
 persistenceClass, 254  
 PersistenceUtils, 462  
 persistent, 298, 968  
 PERSISTENT, 405  
 PERSISTENT\_HEADER, 466  
 persistentSO, 968  
 pickupEvents, 638  
 ping, 411, 683, 733, 733  
 Ping, 818, 820  
 PING\_CLIENT, 820  
 pingInterval, 727  
 pingReceived, 733  
 PipeConnectionEvent, 630, 631  
 PipeUtils, 633  
 play, 522, 522, 551, 551, 551, 551, 552, 554, 1044, 1045, 1053, 1063, 1075, 1075, 1075, 1076, 1076  
 PlayBuffer, 1040, 1040  
 PlayEngine, 1042  
 PlayEngine.Builder, 1047  
 PlaylistSubscriberStream, 1048, 1050  
 POLLING, 405  
 pollingDelay, 876  
 PONG\_SERVER, 820  
 position, 30, 64, 64, 124, 171  
 positions, 116  
 postProcessExtension, 181  
 prepareFilename, 19, 25  
 prepareIO, 64  
 preProcessExtension, 197  
 prettyPrintHex, 207, 208, 208  
 previousId, 267  
 previousItem, 528, 530, 1054, 1063, 1071  
 processEvent, 102  
 processHeaders, 674  
 ProtocolException, 656, 656, 656  
 ProtocolState, 657  
 PROVIDER\_CONNECT\_PULL, 631  
 PROVIDER\_CONNECT\_PUSH, 631  
 PROVIDER\_DISCONNECT, 631  
 providers, 608  
 ProviderService, 1055  
 ProxyFilter, 665, 666  
 publish, 552, 552, 552, 1076, 1076, 1076  
 publishedName, 1060  
 pullAndPush, 1045

pullMessage, 620, 620, 626, 626, 626, 627, 627,  
 1007, 1008, 1100, 1100  
 pushMessage, 621, 625, 626, 627, 1008, 1014,  
 1045, 1063, 1079, 1090, 1091  
 put, 8, 201, 447, 571, 577, 581  
 putInteger, 96  
 putMessage, 1041  
 putServerStream, 567  
 putString, 53, 53, 96, 96, 160

**Q**

QuartzSchedulingService, 918, 918  
 QuartzSchedulingServiceJob, 920  
 QuartzSchedulingServiceMBean, 920

**R**

raiseOriginalException, 341  
 RamPersistence, 910, 911, 911  
 RandomGUID, 216  
 raw, 665  
 rawBufferRecieved, 747  
 rawWrite, 734, 744, 871  
 READ, 105  
 readArray, 45, 89, 156, 184  
 readBean, 45  
 readBoolean, 46, 65, 71, 78, 89, 156, 184  
 readByte, 65, 71, 78  
 readByteArray, 46, 89, 156, 184  
 readBytes, 65, 65, 65, 71, 71, 71, 78, 79, 79, 869  
 readCustom, 46, 90, 156, 185  
 readDataType, 46, 46, 90, 157, 185  
 readDate, 47, 90, 157, 185  
 readDouble, 65, 72, 79  
 readerFromNearestKeyFrame, 112, 119  
 readExternal, 6, 9, 86  
 readFloat, 65, 72, 79  
 readHeaders, 690  
 readInt, 66, 72, 80  
 readKeyValues, 47, 47, 157, 185  
 readMap, 47, 90, 157, 185  
 readMediumInt, 201, 214, 755  
 readMediumInt2, 214  
 readMediumIntOld, 755  
 readMessages, 244  
 readMetaCue, 142, 151  
 readMetaData, 142, 152  
 readMultiByte, 66, 72, 80  
 readNull, 48, 91, 157, 185  
 readNumber, 48, 91, 157, 186

readObject, 48, 66, 72, 80, 91, 157, 186  
 readPropertyName, 48  
 readReference, 48, 91, 158, 186  
 readReverseInt, 214, 755  
 readReverseIntOld, 755  
 readShort, 66, 72, 80  
 readSimpleObject, 49  
 readString, 49, 91, 158, 186  
 readTag, 30, 125, 171  
 readUnsignedByte, 66, 72, 80  
 readUnsignedInt, 66, 73, 81  
 readUnsignedMediumInt, 202, 214, 756  
 readUnsignedMediumIntOld, 756  
 readUnsignedShort, 66, 73, 81  
 readUTF, 67, 73, 81  
 readUTFBytes, 67, 73, 81  
 readXML, 49, 92, 158, 186  
 realClose, 871, 876  
 realm, 1105  
 receiveAudio, 553, 555, 1045, 1046, 1054, 1076  
 RECEIVED, 105  
 receivedBytesRead, 734  
 receiveVideo, 553, 555, 1046, 1046, 1054, 1076  
 recordingFilename, 1061  
 RecordSet, 191, 192  
 RecordSetPage, 194, 195  
 Red5, 432, 433, 433  
 red5AppCtx, 292  
 Red5MBean, 435  
 Red5WebPropertiesConfiguration, 593  
 RedirectHTTPServlet, 897  
 refcount, 797  
 referenceMode, 173  
 refId, 173, 176  
 refMap, 173, 176  
 refreshHeaders, 113, 119  
 register, 262, 290, 345, 417, 974  
 registerBasicScope, 251  
 registerBroadcastStream, 1030, 1057  
 registerBWControllable, 1020, 1024, 1067  
 registerCallback, 471, 945  
 registerConnection, 642  
 registerDeferredResult, 734  
 registered, 344  
 registerGlobal, 338, 431  
 registerPendingCall, 734  
 registerServiceHandler, 317, 329, 478, 492, 956,  
 956, 989, 989  
 registerSharedObjectSecurity, 371, 498, 989

registerStream, 734  
registerStreamPlaybackSecurity, 371, 548  
registerStreamPublishSecurity, 371, 548  
registry, 259  
rejectClient, 372, 372  
release, 488, 794, 798, 811, 974, 989  
releaseInternal, 795, 799, 801, 803, 804, 817, 822, 825, 827, 829, 982  
releaseStream, 553, 1077  
remoteAddress, 244  
remoteAddresses, 244  
remotePort, 244  
RemotingCall, 693, 694  
RemotingClient, 671, 672, 672, 672  
RemotingClient.RemotingWorker, 675  
RemotingCodecFactory, 687  
RemotingConnection, 675, 678  
RemotingHeader, 684, 685  
RemotingMessage, 16  
RemotingPacket, 695, 696  
RemotingProtocolDecoder, 688  
RemotingProtocolEncoder, 690  
remove, 9, 256, 448, 448, 461, 462, 571, 571, 577, 577, 581, 581, 907, 907, 913, 913  
removeAlias, 1113  
removeAllItems, 528, 1054, 1064  
removeAttribute, 239, 300, 384, 391, 683, 956, 974, 990  
removeAttributes, 240, 300, 385, 392, 683, 956, 974, 990  
removeChildScope, 283, 318, 348, 423, 426, 440  
removeClient, 226, 267  
removeContext, 292, 593, 1107  
removeEventListener, 256, 454, 990  
removeHeader, 465, 674, 683  
removeItem, 528, 1054, 1064  
removeListener, 338, 338, 372, 431, 431  
removeMapping, 339, 431  
removePipeConnectionListener, 609, 623, 1008  
removeRed5ApplicationContext, 293  
removeScheduledJob, 372, 470, 919  
removeServerStream, 567  
removeSharedObjectListener, 491, 956, 990  
removeStreamListener, 517, 1014  
removeTokenBucket, 1038  
removeValve, 1113  
REPLACE\_GATEWAY\_URL, 466  
request, 679, 696  
RequestDumpServlet, 898  
required, 685  
requiresConfigInterface, 924, 926, 928  
reserveStreamId, 538, 735  
reset, 49, 49, 92, 557, 916, 1028, 1028, 1036, 1037, 1046, 1078, 1082, 1083, 1085, 1087  
resetBuckets, 1020, 1025, 1068  
resetCredentials, 674  
ResetMessage, 1096  
resolve, 144, 154  
Resolver, 153, 153  
resolveScope, 273, 274, 274, 332, 333, 415, 416, 427, 428, 445  
resolveService, 936, 941, 941, 946  
resolvesToAbsolutePath, 541, 1018  
ResourceExistException, 558, 558  
ResourceNotFoundException, 558, 559  
resources, 911  
resultReceived, 472, 671, 1080  
resume, 522, 555, 1046, 1054  
retain, 794, 799, 812  
retrievePendingCall, 735  
returnAttributeValue, 974  
returnConnection, 230  
returnDistinctObjects, 230  
returnError, 669, 669, 974  
returnMessage, 884, 885, 885  
returnPendingMessages, 885  
returnSameObjects, 230  
RhinoScriptFactory, 927, 928  
RhinoScriptUtils, 929  
rollbackRequest, 1068  
roomConnect, 355, 373  
roomDisconnect, 355, 373  
roomJoin, 355  
roomLeave, 355, 373  
roomStart, 356, 373  
roomStop, 356, 374  
RTMP, 765, 767  
RTMPClient, 719, 720  
RTMPClientConnManager, 720  
RTMPCodecFactory, 773  
RTMPConnection, 721, 725  
RTMPConnManager, 720  
RTMPHandler, 738  
RTMPHandshake, 740  
RTMPMessage, 1095  
RTMPMinaCodecFactory, 775  
RTMPMinaConnection, 741, 743  
RTMPMinaConnectionMBean, 744

RTMPMinaloHandler, 745  
 RTMPMinaProtocolDecoder, 777  
 RTMPMinaProtocolEncoder, 778  
 RTMPMinaTransport, 749  
 RTMPMinaTransportMBean, 750  
 RTMPOriginConnection, 751  
 RTMPProtocolDecoder, 779, 781  
 RTMPProtocolEncoder, 786  
 rtmpsEngine, 865  
 RTMPTClient, 871  
 RTMPTClientConnection, 872  
 RTMPTCodecFactory, 889  
 rtmpConfig, 879  
 RTMPTConnection, 873  
 rtmptEngine, 887  
 RTMPTHandler, 877  
 RTMPTLoader, 878  
 RTMPTProtocolDecoder, 890  
 RTMPTProtocolEncoder, 891  
 rtmptServer, 879  
 RTMPTServlet, 880  
 RTMPUtils, 753  
 run, 909, 916, 1094  
 RuntimeStatusObject, 843, 844, 844, 844

**S**  
 save, 462, 908, 913  
 saveAs, 517, 1014, 1064  
 saveKeyFrameMeta, 21, 21  
 saveObject, 908  
 saveStacks, 286  
 SCHEDULED\_JOB, 920  
 scheduleGhostConnectionsCleanup, 374  
 scheduleNextMessage, 1064  
 SCHEDULING\_SERVICE, 920  
 schedulingService, 361  
 scope, 244, 380, 679  
 Scope, 302, 306, 306  
 ScopeHandlerNotFoundException, 586, 586  
 ScopeMBean, 321, 435  
 ScopeNotFoundException, 587, 587  
 scopePath, 696  
 ScopeResolver, 331  
 ScopeServiceResolver, 945  
 ScopeShuttingDownException, 587, 588  
 ScopeUtils, 441  
 ScreenVideo, 1084, 1084  
 searchNextFrame, 171  
 seek, 522, 531, 553, 555, 1022, 1032, 1032,  
 1047, 1054, 1064, 1077, 1100  
 send, 636, 637  
 SEND\_ALL, 1027  
 SEND\_INTERFRAMES, 1027  
 SEND\_KEYFRAMES, 1027  
 SEND\_KEYFRAMES\_CHECK, 1027  
 sendMessage, 491, 956, 975, 990  
 sendNetStreamStatus, 1077  
 sendOOBControlMessage, 610, 610, 620, 622,  
 1008, 1009  
 sendPacket, 1028, 1083  
 sendResponse, 894  
 sendStats, 968  
 sendStatus, 711  
 sendUpdates, 975  
 SENT, 105  
 SEPARATOR, 419, 436  
 SequencedMessage, 4  
 serialize, 183, 193, 197, 300, 459, 849, 860, 975  
 serializeAsXML, 203  
 Serializer, 195  
 serializer, 688, 774, 863, 890  
 serializeStatusObject, 864  
 SerializeUtils, 823  
 Server, 333  
 server, 344, 739, 879, 887, 893  
 ServerBW, 823, 824  
 ServerStream, 1057, 1060  
 service, 886, 894, 896, 897, 897, 898, 902  
 SERVICE\_NAME, 668, 947  
 serviceAMF, 895  
 serviceCall, 283, 349, 426  
 serviceInvoker, 668, 700  
 ServiceInvoker, 946  
 serviceMethodName, 932  
 serviceName, 932  
 ServiceNotFoundException, 588, 947, 948  
 serviceProvider, 701  
 ServiceUtils, 480, 948  
 servlet, 876  
 servletContext, 344  
 servletMappings, 888  
 ServletUtils, 899  
 session, 679  
 SESSION\_KEY, 658  
 sessionClosed, 103, 665, 666, 748  
 sessionCreated, 103, 663, 748  
 sessionId, 245

sessionId, 103  
sessionOpened, 103, 663, 748  
setApplication, 861  
setApplicationContext, 274, 278, 293, 339, 572, 577, 581, 710, 748, 880  
setApplicationLoader, 293  
setArguments, 934  
setAttribute, 240, 301, 385, 392, 683, 956, 975, 990  
setAttributes, 240, 240, 301, 301, 385, 385, 392, 392, 683, 684, 956, 957, 975, 976, 990, 990  
setAutoStart, 318, 330  
setBandwidth, 805, 825  
setBandwidthConfigure, 262, 393, 735, 1001, 1054  
setBaseHost, 1107  
setBitDigits, 208, 208  
setBitflags, 133  
setBody, 27, 134, 1095, 1097  
setBodySize, 27, 134  
setBooleanProperty, 604, 616  
setBufferCheckInterval, 1054  
setBufferSize, 125  
setBufferType, 125  
setByteProperty, 604, 616  
setByteSeparator, 209  
setBytesRead, 801  
setCache, 113, 120  
setCacheConfigs, 577  
setCached, 449, 573  
setCacheManagerEventListener, 578  
setCall, 817  
setCanCallService, 349  
setCanConnect, 349  
setCanSeekToEnd, 140, 148  
setCanStart, 349  
setCapacity, 1042  
setChannel, 713  
setChannelId, 837  
setChildLoadPath, 318, 330  
setClassName, 917  
setClientBufferDuration, 521, 1002  
setClientCallback, 695  
setClientid, 845, 849  
setClientRegistry, 274  
setClientTTL, 374  
setCode, 850, 861  
setCodecFactory, 663, 704, 748, 878  
setCodecInfo, 1004  
setCodecs, 1081  
setCodeSection, 287  
setConnection, 1002  
setConnectionLocal, 435  
setConnectionParams, 817  
setConnectionProperties, 1107  
setConnector, 1108  
setConnectors, 1108  
setConsumer, 633  
setContext, 318  
setContextPath, 275, 345  
setContexts, 1108, 1113  
setContextsConfig, 278  
setCorrelationID, 605, 616  
setCredentials, 674  
setData, 134, 817, 841  
setDataOffset, 109  
setDataType, 27, 134, 837  
setDebug, 771, 822  
setDefaultApp, 295  
setDeliveryMode, 194, 194, 194  
setDepth, 318, 330  
setDescription, 850  
setDescription, 850, 861  
setDeserializer, 114, 127, 152, 688, 690, 775, 786, 890  
setDetails, 845, 850  
setDiskExpiryThreadIntervalSeconds, 578  
setDiskStore, 578  
setDoubleProperty, 605, 616  
setDumpTo, 663  
setDuration, 140, 149  
setEmbedded, 1108  
setEnabled, 319, 330  
setEncoding, 771  
setEndian, 67, 73, 75, 81, 83  
setEndpoints, 670  
setException, 474, 935  
setExecutor, 1054  
setExtension, 908  
setFile, 152  
setFlagAudio, 109  
setFlagReserved01, 109  
setFlagReserved02, 110  
setFlagVideo, 110  
setFloatProperty, 605, 617  
setFLV, 129  
setForward, 663  
setFrameRate, 141, 149

setGenerateMetadata, 127  
setGhostConnsCleanupPeriod, 374  
setGlobalScope, 333, 345, 998  
setHandler, 319, 748, 886  
setHandshake, 771  
setHasAudio, 1088  
setHasVideo, 1088  
setHeader, 799, 812  
setHeight, 141, 149  
setHost, 1108  
setHosts, 1109  
setInStream, 143, 152  
setIntProperty, 605, 617  
setInvokeld, 713, 818  
setIoSession, 744  
setIsPersistent, 982  
setItem, 528, 1055, 1064  
setJoin, 349  
setKeyFrameData, 113, 120  
setLastReadHeader, 771  
setLastReadPacket, 771  
setLastWriteHeader, 772  
setLastWritePacket, 772  
setLength, 565  
setLevel, 850, 861  
setListeners, 610  
setLocalContext, 221  
setLongProperty, 605, 617  
setMappings, 888  
setMappingStrategy, 275  
setMaxEntries, 448, 572, 578, 581  
setMaxHandshakeTimeout, 735  
setMaxInactivity, 735  
setMemoryStoreEvictionPolicy, 578  
setMessage, 841  
setMessageID, 605, 617  
setMessageType, 606, 617  
setMetaCue, 141, 149  
setMetaData, 113, 120  
setMetaService, 113, 120  
setMethodName, 917  
setMethodParams, 917  
setMinaDecoder, 777  
setMinaEncoder, 777  
setMode, 748  
setMsgInput, 565  
setName, 137, 146, 301, 319, 330, 345, 450, 459, 565, 573, 976, 982, 990, 1004  
setObjectProperty, 606, 618  
setOffset, 129  
setOutStream, 143, 152  
setPacket, 684  
setParamMap, 633  
setParamTypes, 917  
setParent, 319, 345  
setParentContext, 279  
setPath, 301, 459, 908, 976, 990  
setPermissions, 263, 402  
setPersistanceStore, 275  
setPersistenceClass, 290, 320, 331  
setPersistenceClassName, 993  
setPersistent, 302, 460, 976  
setPingInterval, 735  
setPlayItem, 532  
setPlaylistController, 529, 1055, 1064  
setPreviousTagSize, 27, 134  
setPreviuosTagSize, 134  
setProvider, 633  
setPublishedName, 518, 1015, 1064  
setRandom, 529, 1055, 1064  
setReadChunkSize, 772  
setRealm, 1109  
setRed5ApplicationContext, 293  
setRemotingClient, 194  
setRepeat, 529, 1055, 1064  
setResolver, 152  
setResourceLoader, 663  
setResult, 472, 629, 713, 917, 945  
setRewind, 529, 1055, 1065  
setSchedulingService, 736  
setScope, 385, 424, 1004  
setScopePath, 697  
setScopeResolver, 275  
setSerializer, 114, 127, 152, 688, 692, 775, 792, 864, 890  
setServer, 290, 345, 740, 866, 880, 888  
setServiceCall, 713  
setServiceInvoker, 275, 670  
setServiceMethodName, 935  
setServiceName, 629, 935  
setServiceParamMap, 629  
setServiceProvider, 705  
setServiceResolvers, 947  
setServices, 37  
setServlet, 876  
setServletContext, 346  
setServletRequest, 876  
setShortProperty, 606, 618

setSignature, 110  
setSize, 566, 803, 838  
setSource, 799, 812  
setStart, 566, 1100  
setState, 772  
setStatus, 474, 935  
setStatusObjectService, 740  
setStore, 302, 460, 976  
setStreamEventDispatcher, 705  
setStreamId, 838, 1002  
setStringProperty, 606, 618  
setTarget, 630  
setThreadPool, 674  
setTime, 137, 146  
setTimeout, 287  
setTimer, 838  
setTimerRelative, 838  
setTimestamp, 27, 134, 799, 812  
setType, 135, 138, 146, 633  
setTypeFlags, 110  
setUnderrunTrigger, 1055  
setup, 736  
setupClassLoader, 786  
setUpstreamBandwidth, 412, 563  
setValue1, 822  
setValue2, 805, 822  
setValue3, 822  
setValue4, 823  
setValves, 1109  
setVersion, 110, 982  
setVideoCodec, 1088  
setVideoCodecId, 141, 149  
setVideoDataRate, 141, 149  
setVirtualHosts, 346  
setWebAppContext, 595  
setWebappFolder, 293  
setWidth, 141, 150  
setWithByteSeparator, 209  
setWriteChunkSize, 773  
SharedMidiObject, 635  
SharedMidiObject.MidiReceiver, 635  
SharedObject, 962, 966, 966, 966, 967  
SharedObjectEvent, 977, 977  
SharedObjectException, 588  
SharedObjectMessage, 978, 980, 980  
sharedObjects, 701  
SharedObjectScope, 982, 985  
SharedObjectService, 991  
SharedObjectTypeMapping, 841  
Shutdown, 339  
shutdown, 593, 597, 909, 1109  
shuttingDown, 344  
signature, 107  
SimpleBandwidthConfigure, 560  
SimpleBWControlService, 1065  
SimpleBWControlService.BWContext, 1069  
SimpleBWControlService.TokenRequest, 1069  
SimpleBWControlService.TokenRequestContext, 1070  
SimpleBWControlService.TokenRequestType, 1070  
SimpleClient, 223  
SimpleConnectionBWConfig, 562  
SimpleMRTMPEdgeManager, 649  
SimpleMRTMPOriginManager, 650  
SimplePlayItem, 563  
SimplePlaylistController, 1070  
SimpleProtocolCodecFactory, 659  
SimpleProtocolDecoder, 660  
SimpleProtocolEncoder, 661  
skipData, 886  
skipEndObject, 50  
skipPropertySeparator, 50  
SLASH, 335  
so, 985  
SO\_CLIENT\_CLEAR\_DATA, 832  
SO\_CLIENT\_DELETE\_DATA, 832  
SO\_CLIENT\_INITIAL\_DATA, 832  
SO\_CLIENT\_SEND\_MESSAGE, 832  
SO\_CLIENT\_STATUS, 832  
SO\_CLIENT\_UPDATE\_ATTRIBUTE, 833  
SO\_CLIENT\_UPDATE\_DATA, 833  
SO\_CONNECT, 833  
SO\_CREATION\_FAILED, 858  
SO\_DELETE\_ATTRIBUTE, 833  
SO\_DISCONNECT, 833  
SO\_NO\_READ\_ACCESS, 858  
SO\_NO\_WRITE\_ACCESS, 858  
SO\_PERSISTENCE\_MISMATCH, 858  
SO\_SEND\_MESSAGE, 833  
SO\_SET\_ATTRIBUTE, 833  
SocketPolicyHandler, 654  
SorensonVideo, 1085, 1086  
SOUND\_RATE\_11\_KHZ, 807  
SOUND\_RATE\_22\_KHZ, 807  
SOUND\_RATE\_44\_KHZ, 807  
SOUND\_RATE\_5\_5\_KHZ, 807  
SOUND\_SIZE\_16\_BIT, 807

SOUND\_SIZE\_8\_BIT, 808  
 source, 16, 797, 969  
 stacks, 286  
 Standalone, 340, 905  
 start, 283, 320, 331, 349, 352, 374, 426, 533, 1015, 1047, 1055, 1065, 1094  
 startBroadcastVOD, 1065  
 startConnector, 705  
 startDecoding, 659  
 startPublishing, 520, 1015  
 startRoundTripMeasurement, 736  
 startUp, 230, 233  
 startWaitForHandshake, 736  
 startWebApplication, 1114  
 state, 1003  
 STATE\_CONNECT, 767  
 STATE\_CONNECTED, 767  
 STATE\_DISCONNECTED, 768  
 STATE\_EDGE\_CONNECT\_ORIGIN\_SENT, 768  
 STATE\_EDGE\_DISCONNECTING, 768  
 STATE\_ERROR, 768  
 STATE\_HANDSHAKE, 768  
 STATE\_ORIGIN\_CONNECT\_FORWARDED, 768  
 StatefulScopeWrappingAdapter, 379  
 StatisticsCounter, 514  
 StatisticsService, 994  
 StatisticsServlet, 901  
 Status, 845, 847, 847, 847  
 STATUS, 848  
 status, 932  
 STATUS\_ACCESS\_DENIED, 932  
 STATUS\_APP\_SHUTTING\_DOWN, 933  
 STATUS\_GENERAL\_EXCEPTION, 933  
 STATUS\_INVOCATION\_EXCEPTION, 933  
 STATUS\_METHOD\_NOT\_FOUND, 933  
 STATUS\_PENDING, 933  
 STATUS\_SERVICE\_NOT\_FOUND, 933  
 STATUS\_SUCCESS\_NULL, 933  
 STATUS\_SUCCESS\_RESULT, 933  
 STATUS\_SUCCESS\_VOID, 933  
 StatusCodes, 851  
 StatusMessage, 1096  
 StatusObject, 858, 859  
 statusObjects, 863  
 statusObjectService, 739  
 StatusObjectService, 861  
 stop, 283, 320, 331, 350, 352, 375, 388, 427, 523, 533, 555, 590, 1015, 1047, 1055, 1065, 1101  
 stopRecording, 1015  
 storage, 969  
 store, 298  
 storeReference, 174, 174, 176  
 STREAM\_CLEAR, 820  
 STREAM\_PLAYBUFFER\_CLEAR, 820  
 STREAM\_RESET, 820  
 StreamableFileFactory, 37  
 StreamBandwidthController, 1093  
 streamBroadcastClose, 375, 534  
 streamBroadcastStart, 375, 534  
 streamBuffers, 727  
 StreamCodecInfo, 1087  
 StreamControlException, 589  
 StreamDataException, 589  
 StreamingProxy, 1078  
 StreamNotFoundException, 1071  
 streamPlaylistItemPlay, 375, 534  
 streamPlaylistItemStop, 376, 535  
 streamPlaylistVODItemPause, 376, 535  
 streamPlaylistVODItemResume, 376, 535  
 streamPlaylistVODItemSeek, 377, 536  
 streamPublishStart, 377, 536  
 streamRecordStart, 377, 536  
 StreamService, 1071  
 streamSubscriberClose, 378, 536  
 streamSubscriberStart, 378, 536  
 StreamTracker, 1077, 1078  
 StreamUtils, 566  
 stringCache, 52  
 stringToDoc, 218  
 stringToHexString, 209  
 subscopeStats, 307  
 subscribe, 610, 611, 620, 622, 626, 626, 627, 627, 1009, 1009  
 supportsDataType, 53, 96, 160  
 syncEvents, 969

**T**

Tag, 130, 131, 131  
 takeMessage, 1042  
 Test, 636, 637  
 Test.MyReceiver, 637  
 threadPool, 673  
 ThreadPool, 914  
 ThreadPoolMBean, 914

timestamp, 798  
 timestamps, 116  
 toBinaryString, 209, 209, 210, 210, 210, 210  
 toByte, 842  
 toByteArray, 210, 210, 211  
 toHexString, 211, 211, 211, 212, 212, 212  
 TomcatApplicationContext, 1101, 1101  
 TomcatApplicationLoader, 1101, 1102  
 TomcatLoader, 1103  
 TomcatLoader.DirectoryFilter, 1109  
 TomcatRTMPSLoader, 865  
 TomcatRTMPTLoader, 886  
 TomcatVHostLoader, 1110  
 TomcatVHostLoaderMBean, 1114  
 toString, 9, 11, 67, 105, 110, 135, 146, 150, 212, 212, 215, 263, 320, 339, 684, 736, 795, 801, 803, 805, 814, 818, 823, 825, 827, 829, 838, 842, 851, 861, 935, 978, 982, 990  
 toStringValue, 180  
 toType, 842  
 TRANSIENT, 405  
 TRANSIENT\_PREFIX, 457  
 trimClassName, 597  
 type, 93, 245, 298  
 TYPE, 419, 437  
 TYPE\_AMF3\_OBJECT, 40  
 TYPE\_ARRAY, 40, 58  
 TYPE\_AUDIO, 36, 808  
 TYPE\_AUDIO\_DATA, 833  
 TYPE\_BOOLEAN, 40  
 TYPE\_BOOLEAN\_FALSE, 58  
 TYPE\_BOOLEAN\_TRUE, 58  
 TYPE\_BYTEARRAY, 59  
 TYPE\_BYTES\_READ, 833  
 TYPE\_CHUNK\_SIZE, 834  
 TYPE\_CLASS\_OBJECT, 40  
 TYPE\_CLIENT\_BANDWIDTH, 834  
 TYPE\_DATE, 40, 59  
 TYPE\_END\_OF\_OBJECT, 41  
 TYPE\_FLEX\_MESSAGE, 834  
 TYPE\_FLEX\_SHARED\_OBJECT, 834  
 TYPE\_FLEX\_STREAM\_SEND, 834  
 TYPE\_INTEGER, 59  
 TYPE\_INVOKE, 834  
 TYPE\_LONG\_STRING, 41  
 TYPE\_METADATA, 36, 808  
 TYPE\_MIXED\_ARRAY, 41  
 TYPE\_MOVIECLIP, 41  
 TYPE\_NOTIFY, 834  
 TYPE\_NULL, 41, 59  
 TYPE\_NUMBER, 41, 59  
 TYPE\_OBJECT, 41, 59  
 TYPE\_OBJECT\_EXTERNALIZABLE, 59  
 TYPE\_OBJECT\_PROPERTY, 59  
 TYPE\_OBJECT\_PROXY, 60  
 TYPE\_OBJECT\_VALUE, 60  
 TYPE\_PING, 834  
 TYPE\_RECORDSET, 41  
 TYPE\_REFERENCE, 41  
 TYPE\_SERVER\_BANDWIDTH, 834  
 TYPE\_SHARED\_OBJECT, 835  
 TYPE\_STREAM\_METADATA, 835  
 TYPE\_STRING, 42, 60  
 TYPE\_UNDEFINED, 42, 60  
 TYPE\_UNSUPPORTED, 42  
 TYPE\_VIDEO, 37, 808  
 TYPE\_VIDEO\_DATA, 835  
 TYPE\_XML, 42, 60  
 TYPE\_XML\_DOCUMENT, 60  
 typeMap, 842

## U

uncompress, 67  
 UNDEFINED, 820  
 uninit, 320, 1114  
 Unknown, 825, 826  
 UNKNOWN\_2, 820  
 UNKNOWN\_5, 821  
 UNKNOWN\_8, 821  
 unloadContext, 279  
 unlock, 492, 957, 990  
 unregister, 263, 346, 976  
 unregisterBasicScope, 251  
 unregisterBroadcastStream, 1031, 1057  
 unregisterBWControllable, 1020, 1025, 1068  
 unregisterCallback, 472, 945  
 unregisterConnection, 643  
 unregisterDeferredResult, 737  
 unregisterMBean, 597  
 unregisterServiceHandler, 320, 331, 478, 492, 957, 991  
 unregisterSharedObjectSecurity, 378, 498, 991  
 unregisterStream, 737  
 unregisterStreamPlaybackSecurity, 378, 548  
 unregisterStreamPublishSecurity, 379, 549  
 unreserveStreamId, 539, 737  
 unsubscribe, 611, 611, 621, 622, 1009, 1010  
 updateBandwidthConfigure, 1047

updateBWConfigure, 1021, 1025, 1068  
 updateBytesRead, 737  
 updateCounter, 969  
 updateMBeanAttribute, 598, 598  
 updateScopeStatistics, 512, 996  
 updateSharedObjectStatistics, 512, 996

**V**

validateHandshakeReply, 773  
 VALUE\_FALSE, 42  
 VALUE\_TRUE, 42  
 valves, 1105  
 version, 107, 969  
 VERSION, 433  
 VIDEO\_ON2\_VP6, 808  
 VIDEO\_SCREEN\_VIDEO, 808  
 VIDEO\_SORENSEN\_H263, 808  
 VideoCodecFactory, 1080  
 VideoData, 827, 828, 828  
 VideoData.FrameType, 829  
 VideoFrameDropper, 1081, 1082  
 virtualHosts, 344

**W**

W3CAppender, 221  
 WarLoaderServlet, 1117  
 WARNING, 848  
 wasSent, 713  
 WebappClassLoader, 1115, 1116, 1116  
 webAppCtx, 893  
 webappFolder, 292  
 webappRoot, 1112  
 WebScope, 341  
 Worker, 914  
 wrap, 288  
 write, 143, 153, 640, 641, 712, 737, 744, 871  
 writeArbitraryObject, 53, 96  
 writeArray, 53, 53, 54, 97, 97, 97, 160, 161, 161, 188, 188, 188  
 writeArrayType, 197  
 writeBasic, 197  
 writeBoolean, 54, 67, 75, 83, 97, 161, 189  
 writeByte, 67, 75, 83  
 writeByteArray, 54, 97, 161, 189  
 writeBytes, 68, 68, 68, 75, 75, 75, 83, 83, 84  
 writeComplex, 198  
 writeCustom, 54, 161, 189  
 writeCustomType, 198  
 writeDate, 54, 97, 161, 189

writeDocument, 198  
 writeDouble, 68, 76, 84  
 writeExternal, 6, 9, 86  
 writeFloat, 68, 76, 84  
 writeHeader, 31, 129  
 writeInt, 68, 76, 84  
 writeIterator, 198  
 writeList, 199  
 writeListType, 199  
 writeMap, 54, 54, 97, 98, 161, 162, 189, 190  
 writeMediumInt, 202, 215, 756  
 writeMetaCue, 143, 153  
 writeMetaData, 143, 153  
 writeMultiByte, 68, 76, 84  
 writeNull, 55, 98, 162  
 writeNumber, 55, 98, 162, 190  
 writeObject, 55, 55, 69, 76, 85, 98, 98, 162, 162, 190, 190  
 writeObjectType, 199  
 writeRecordSet, 55, 98, 162, 190  
 writeReference, 55, 162, 191  
 writeReverseInt, 215, 756  
 writeReverseIntOld, 756  
 writerFromNearestKeyFrame, 114, 120  
 writeShort, 69, 76, 85  
 writeStream, 31, 129  
 writeString, 55, 99, 163, 191  
 writeTag, 32, 32, 130, 130  
 writeUnsignedInt, 69, 76, 85  
 writeUTF, 69, 77, 85  
 writeUTFBytes, 69, 77, 85  
 writeXML, 56, 99, 163, 191  
 writeXMLType, 199  
 writingMessage, 737  
 WRITTEN, 105  
 written, 1055  
 writtenBytes, 869  
 writtenMessages, 245

**X**

XmlRpcScopeStatistics, 996, 997, 997  
 XMLUtils, 216

**Z**

ZAMFGatewayServlet, 902  
 ZAMFGatewayServlet.Handler, 902