| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ClientRequestInterceptorOperations.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInterceptor.html)   [**NEXT CLASS**](http://docs.google.com/org/omg/PortableInterceptor/Current.html) | [**FRAMES**](http://docs.google.com/index.html?org/omg/PortableInterceptor/ClientRequestInterceptorOperations.html)    [**NO FRAMES**](http://docs.google.com/ClientRequestInterceptorOperations.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#tyjcwt) |

## **org.omg.PortableInterceptor**

Interface ClientRequestInterceptorOperations

**All Superinterfaces:** [InterceptorOperations](http://docs.google.com/org/omg/PortableInterceptor/InterceptorOperations.html) **All Known Subinterfaces:** [ClientRequestInterceptor](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInterceptor.html)

public interface **ClientRequestInterceptorOperations**extends [InterceptorOperations](http://docs.google.com/org/omg/PortableInterceptor/InterceptorOperations.html)

Client-side request interceptor.

A request Interceptor is designed to intercept the flow of a request/reply sequence through the ORB at specific points so that services can query the request information and manipulate the service contexts which are propagated between clients and servers. The primary use of request Interceptors is to enable ORB services to transfer context information between clients and servers. There are two types of request Interceptors: client-side and server-side.

To write a client-side Interceptor, implement the ClientRequestInterceptor interface.

**See Also:**[ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html)

| **Method Summary** | |
| --- | --- |
| void | [**receive\_exception**](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInterceptorOperations.html#receive_exception(org.omg.PortableInterceptor.ClientRequestInfo))([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)            Indicates to the interceptor that an exception occurred. |
| void | [**receive\_other**](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInterceptorOperations.html#receive_other(org.omg.PortableInterceptor.ClientRequestInfo))([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)            Allows an Interceptor to query the information available when a request results in something other than a normal reply or an exception. |
| void | [**receive\_reply**](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInterceptorOperations.html#receive_reply(org.omg.PortableInterceptor.ClientRequestInfo))([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)            Allows an Interceptor to query the information on a reply after it is returned from the server and before control is returned to the client. |
| void | [**send\_poll**](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInterceptorOperations.html#send_poll(org.omg.PortableInterceptor.ClientRequestInfo))([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)            Allows an Interceptor to query information during a Time-Independent Invocation (TII) polling get reply sequence. |
| void | [**send\_request**](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInterceptorOperations.html#send_request(org.omg.PortableInterceptor.ClientRequestInfo))([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)            Allows an Interceptor to query request information and modify the service context before the request is sent to the server. |

| **Methods inherited from interface org.omg.PortableInterceptor.**[**InterceptorOperations**](http://docs.google.com/org/omg/PortableInterceptor/InterceptorOperations.html) |
| --- |
| [destroy](http://docs.google.com/org/omg/PortableInterceptor/InterceptorOperations.html#destroy()), [name](http://docs.google.com/org/omg/PortableInterceptor/InterceptorOperations.html#name()) |

| **Method Detail** |
| --- |

### send\_request

void **send\_request**([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)  
 throws [ForwardRequest](http://docs.google.com/org/omg/PortableInterceptor/ForwardRequest.html)

Allows an Interceptor to query request information and modify the service context before the request is sent to the server.

This interception point may throw a system exception. If it does, no other Interceptors' send\_request operations are called. Those Interceptors on the Flow Stack are popped and their receive\_exception interception points are called. This interception point may also throw a ForwardRequest exception. If an Interceptor throws this exception, no other Interceptors' send\_request operations are called. Those Interceptors on the Flow Stack are popped and their receive\_other interception points are called.

Compliant Interceptors shall properly follow completion\_status semantics if they throw a system exception from this interception point. The completion\_status shall be COMPLETED\_NO.

**Parameters:**ri - Information about the current request being intercepted. **Throws:** [ForwardRequest](http://docs.google.com/org/omg/PortableInterceptor/ForwardRequest.html) - If thrown, indicates to the ORB that a retry of the request should occur with the new object given in the exception.

### send\_poll

void **send\_poll**([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)

Allows an Interceptor to query information during a Time-Independent Invocation (TII) polling get reply sequence.

With TII, an application may poll for a response to a request sent previously by the polling client or some other client. This poll is reported to Interceptors through the send\_poll interception point and the response is returned through the receive\_reply or receive\_exception interception points. If the response is not available before the poll time-out expires, the system exception TIMEOUT is thrown and receive\_exception is called with this exception.

This interception point may throw a system exception. If it does, no other Interceptors' send\_poll operations are called. Those Interceptors on the Flow Stack are popped and their receive\_exception interception points are called.

Compliant Interceptors shall properly follow completion\_status semantics if they throw a system exception from this interception point. The completion\_status shall be COMPLETED\_NO.

**Parameters:**ri - Information about the current request being intercepted. **Throws:** TIMEOUT - thrown if the response is not available before the poll time-out expires

### receive\_reply

void **receive\_reply**([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)

Allows an Interceptor to query the information on a reply after it is returned from the server and before control is returned to the client.

This interception point may throw a system exception. If it does, no other Interceptors' receive\_reply operations are called. The remaining Interceptors in the Flow Stack shall have their receive\_exception interception point called.

Compliant Interceptors shall properly follow completion\_status semantics if they throw a system exception from this interception point. The completion\_status shall be COMPLETED\_YES.

**Parameters:**ri - Information about the current request being intercepted.

### receive\_exception

void **receive\_exception**([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)  
 throws [ForwardRequest](http://docs.google.com/org/omg/PortableInterceptor/ForwardRequest.html)

Indicates to the interceptor that an exception occurred. Allows an Interceptor to query the exception's information before it is thrown to the client.

This interception point may throw a system exception. This has the effect of changing the exception which successive Interceptors popped from the Flow Stack receive on their calls to receive\_exception. The exception thrown to the client will be the last exception thrown by an Interceptor, or the original exception if no Interceptor changes the exception.

This interception point may also throw a ForwardRequest exception. If an Interceptor throws this exception, no other Interceptors' receive\_exception operations are called. The remaining Interceptors in the Flow Stack are popped and have their receive\_other interception point called.

If the completion\_status of the exception is not COMPLETED\_NO, then it is inappropriate for this interception point to throw a ForwardRequest exception. The request s at-most-once semantics would be lost.

Compliant Interceptors shall properly follow completion\_status semantics if they throw a system exception from this interception point. If the original exception is a system exception, the completion\_status of the new exception shall be the same as on the original. If the original exception is a user exception, then the completion\_status of the new exception shall be COMPLETED\_YES.

Under some conditions, depending on what policies are in effect, an exception (such as COMM\_FAILURE) may result in a retry of the request. While this retry is a new request with respect to Interceptors, there is one point of correlation between the original request and the retry: because control has not returned to the client, the PortableInterceptor.Current for both the original request and the retrying request is the same.

**Parameters:**ri - Information about the current request being intercepted. **Throws:** [ForwardRequest](http://docs.google.com/org/omg/PortableInterceptor/ForwardRequest.html) - If thrown, indicates to the ORB that a retry of the request should occur with the new object given in the exception.

### receive\_other

void **receive\_other**([ClientRequestInfo](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInfo.html) ri)  
 throws [ForwardRequest](http://docs.google.com/org/omg/PortableInterceptor/ForwardRequest.html)

Allows an Interceptor to query the information available when a request results in something other than a normal reply or an exception. For example, a request could result in a retry (e.g., a GIOP Reply with a LOCATION\_FORWARD status was received); or on asynchronous calls, the reply does not immediately follow the request, but control shall return to the client and an ending interception point shall be called.

For retries, depending on the policies in effect, a new request may or may not follow when a retry has been indicated. If a new request does follow, while this request is a new request, with respect to Interceptors, there is one point of correlation between the original request and the retry: because control has not returned to the client, the request scoped PortableInterceptor.Current for both the original request and the retrying request is the same.

This interception point may throw a system exception. If it does, no other Interceptors' receive\_other operations are called. The remaining Interceptors in the Flow Stack are popped and have their receive\_exception interception point called.

This interception point may also throw a ForwardRequest exception. If an Interceptor throws this exception, successive Interceptors' receive\_other operations are called with the new information provided by the ForwardRequest exception.

Compliant Interceptors shall properly follow completion\_status semantics if they throw a system exception from this interception point. The completion\_status shall be COMPLETED\_NO. If the target invocation had completed, this interception point would not be called.

**Parameters:**ri - Information about the current request being intercepted. **Throws:** [ForwardRequest](http://docs.google.com/org/omg/PortableInterceptor/ForwardRequest.html) - If thrown, indicates to the ORB that a retry of the request should occur with the new object given in the exception.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ClientRequestInterceptorOperations.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/org/omg/PortableInterceptor/ClientRequestInterceptor.html)   [**NEXT CLASS**](http://docs.google.com/org/omg/PortableInterceptor/Current.html) | [**FRAMES**](http://docs.google.com/index.html?org/omg/PortableInterceptor/ClientRequestInterceptorOperations.html)    [**NO FRAMES**](http://docs.google.com/ClientRequestInterceptorOperations.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#tyjcwt) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).