# ORBX.js IFRAME API

**Version 1.0**

**OTOY Inc**

# Table of Contents

1. Introduction....3
2. Query String for Client....3
2.1. connect=<connection-string>....3
2.2. cid=<cid>....3
2.3. origin=<parent-origin>....3
3. Messaging....3
3.1. Object Message Format....3
3.1.1. type (string)....3
3.1.2. sid (string)....3
3.1.3. cid (string)....4
3.1.4. data....4
3.2. String Message Format....4
3.3. Listening for Messages from the Client....4
3.4. Sending Messages to the Client....4
3.5. Message Types....4
3.5.1. ready....4
3.5.2. wpos....4
3.5.3. wnew....5
3.5.4. wttl....5
3.5.5. udat....6
3.5.6. orbx....6
4. Fullscreen Support....6

# 1 Introduction

This document describes the IFRAME messaging API for the ORBX.js client.

# 2 Query String for Client

When launching the client in an iframe, the client HTML supports parameters passed in the query string. The following sections detail the currently supported parameters.

## *2.1 connect=<connection-string>*

This parameter is mandatory, and supplies the connection string for the client. The connection string must be safely URL encoded.

## *2.2 cid=<cid>*

This parameter is optional, and allows the parent to supply a client ID string to the client. This is necessary when a parent supports multiple clients rendered in separate iframes, and allows the parent to identify which client is the source of messages that it receives.

## *2.3 origin=<parent-origin>*

This parameter is a URL (without the path) indicating the origin of the parent, and allows the client to reject all messages that do not originate from the parent.

*Note that this is an important security measure, without which malicious scripts can control the client through the messaging.*

# 3 Messaging

The ORBX.js client supports communication with the parent document through javascript messages.

## *3.1 Object Message Format*

A message is a javascript object that has a few mandatory properties and possibly some optional properties as defined by the type of the message. The following sections detail the properties.

### 3.1.1 type (string)

Mandatory field that indicates the type of the message.

### 3.1.2 sid (string)

For messages from the client to the parent, this is set to the base64 encoded sessionid of the current session.

### 3.1.3 cid (string)

For messages from the client to the parent, and where a Client ID is provided by the parent, this is set to that client id. The parent may set a client ID for the client by including cid=<cid>; in the query string of the iframe source.

### 3.1.4 data

This optional field contains data for messages that require it. This field may be a string, and ArrayBuffer or a Typed Array.

## *3.2 String Message Format*

For messages from the parent that require only a type and optionally a data field, the parent may send the message as a single string, where a space character is used to separate the type from the data field.

## *3.3 Listening for Messages from the Client*

The parent may listen for messages from the client by handling the "message" event.

```
window.addEventListener("message", function(e) { … });
```

## *3.4 Sending Messages to the Client*

The parent may send messages to the client by using postMessage on the iframe content window.

iframe.contentWindow.postMessage(msg, "*");

*Note, for security reasons, you should always put in the url (minus the path) of the iframe source instead of "*", as this will avoid any messages being eavesdropped by malicious code.*

## *3.5 Message Types*

The following sections detail each of the currently supported message types.

### 3.5.1 ready

This message is sent from the client to the parent to indicate that the client is ready to receive messages from the parent. Any messages posted to the client from the parent before this message is received will not get processed.

### 3.5.2 wpos

This message is sent from the client to the parent to notify about changes in window position and size, usually directed from the server. The message is also sent from the parent to the client to request that the window size be changed, for example if the parent is simulating window chrome and the user resizes the window.

The message has the following properties:

- **x** – The x coordinate of the top left corner of the window.
- **y** – The y coordinate of the top left corner of the window.
- **w** – The width of the window.
- **h** – The height of the window.

### 3.5.3 wnew

This message is sent from the client to the parent to notify about a new window. This message provides the initial properties of the window.

The message has the following properties:

- **x** – The x coordinate of the top left corner of the window.
- **y** – The y coordinate of the top left corner of the window.
- **w** – The width of the window.
- **h** – The height of the window.
- **style** – The style bitfield for the window chrome.

The window style bitfield is a combination of the following values:

| name | value | description |
|---|---|---|
| `owsTitleBar` | 1 | The window has a caption or title bar. |
| `owsMinimize` | 2 | The window has a minimize button. |
| `owsMaximize` | 4 | The window has a maximize button. |
| `owsClose` | 8 | The window has a close button. |
| `owsSizable` | 16 | The window is sizable by the user. |
| `owsToolWindow` | 32 | The window is a tool window. |
| `owsTopMost` | 64 | The window should be kept topmost. |
| `owsClientCoords` | 128 | The window size describes client coordinates. |
| `owsBorderless` | 256 | The window is borderless |
| `owsMaxFSExclusive` | 2048 | The window should go fullscreen when maximized. |

### 3.5.4 wttl

This message is sent from the client to the parent to notify about changes to the window title.

The message has the following properties:

- **data** – The new title string.

### 3.5.5 udat

This message may be sent or received by the parent, and represents arbitrary binary user data, which gets passed to or from the server.

The message has the following properties:

- **data** – The binary data payload (array or typed array).

### 3.5.6 orbx

This message may be sent or received by the parent, and represents an arbitrary text based command. This message may be sent as a string rather than an object.

The message has the following properties:

- **data** – The command (string).

## 4 Fullscreen Support

The browsers do not support entering fullscreen programatically except through a user interactive event such as a keyboard or button press. So in order to put the client into fullscreen, the following code may be used, but it must be called within a button click or keyboard event handler for it to work.

```
function toggleFullScreen(elem) {
     if(!elem.requestFullscreen)
          elem.requestFullscreen = elem.requestFullscreen ||
               elem.webkitRequestFullscreen ||
               elem.mozRequestFullScreen;
     if(!document.exitFullscreen)
          document.exitFullscreen = document.exitFullscreen ||
               document.webkitExitFullscreen ||
               document.mozExitFullscreen ||
               document.webkitCancelFullScreen ||
               document.mozCancelFullScreen;
     if(elem.requestFullscreen && document.exitFullscreen) {
          var fs = document.fullscreenElement ||
               document.mozFullScreenElement ||
               document.webkitFullscreenElement;
          if(fs)
               document.exitFullscreen();
          else
               elem.requestFullscreen();
     }
}
```

The iframe containing the client should be passed to this routine to toggle the fullscreen state.