Exercise:
X-Document Messaging

Estimated time: 20 minutes

Learning Objectives:

- Work with the XDM postMessage

Program Objectives

- Modify /etc/hosts

Resources:

Project Folder: crossdoc
Starting Files:

- xdm1.html
- xdm2.html

Directions:

Part 1 - Modify your hosts file

In this section we'll modify our /etc/hosts file to emulate a pair of servers

1. Using "sudo" on a Unix platform, modify your /etc/hosts to include the following entries. On Windows, modify the C:\Windows\System32\drivers\etc\hosts:

```
127.0.0.1       dev1.nextgened.com
127.0.0.1       dev2.nextgened.com
```

2. Use the ping command to ping each name in turn. Make sure you get back a response time:

```
ping -c 5 dev1.nextgened.com
```

3. Open the mainPage.html file. Examine its structure. Note that it is using the crossdomain.js script.
4. Open the inner.html page. Examine its structure. Note that it is using the inner.js script.

5. Open the browser and verify that you can access the mainPage.html from the dev1 host name:

```
http://dev1.nextgened.com:8080/HTML5/crossdomain/solution/
mainPage.html
```

6. Open the browser and verify that you can access the inner.html from the dev2 host name:

```
http://dev2.nextgened.com:8080/HTML5/crossdomain/solution/
inner.html
```

7. In the mainPage.html, look for the comment "Inner Frame Goes Here" and add in the iframe, with a src pointing to dev2:

```
<iframe src="http://dev2.nextgened.com:8080/HTML5/
crossdomain/solution/inner.html" id="feedback"></iframe>
```

8. Open the crossdomain.js file. Add global variables for serviceMessages, divChatMessages, divStatus, and ifFeedback. The btnPressMe variable should already be in the file:

```
var serviceMessages;
var divChatMessages;
var divStatus;
var ifFeedback;
```

9. In the init method, use document.getElementById to assign divChatMessages to the chatMessages div, ifFeedback to the feedback frame, and divStatus to the status div. The btnPressMe is already assigned:

```
function init() {
 btnPressMe = document.getElementById("pressMe");
 divChatMessages = document.getElementById("chatMessages");
 ifFeedback = document.getElementById("feedback");
 divStatus = document.getElementById("status");
}
```

10. Assign the btnPressMe.onclick handler to be a function called sendMessage, which we will write shortly.

```
btnPressMe.onclick = sendMessage;
```

11. Assign the window.onmessage handler to be a function called receiveMessage, which we will write shortly.

```
window.onmessage = receiveMessage;
```

12 . The final init should look something like this:

```
function init() {
 btnPressMe = document.getElementById("pressMe");
 divChatMessages = document.getElementById("chatMessages");
 ifFeedback = document.getElementById("feedback");
 divStatus = document.getElementById("status");
 btnPressMe.onclick = sendMessage;
 window.onmessage = receiveMessage;
}
```

13 . Create the sendMessage function. It takes no arguments.

14 . First check to see if window.postMessage is defined, otherwise post an alert (or set the status if you'd prefer):

15 . If window.postMessage is supported, create a new Object called msg. Assign it an arbitrary messageId, messageType, and message properties:

16 . Use the ifFeedback.contentWindow to post a JSON stringification of the msg. Post to targets on that originated from dev2.

17 . The final sendMessage should look something like the following:

```
function sendMessage() {
 if (typeof window.postMessage === "undefined") {
  alert("XDM is not supported on this browser!");
 } else {
  var msg = new Object();
  msg.messageId = 1234;
  msg.messageType = "OrderSent";
  msg.message = "Your order has been sent";

 ifFeedback.contentWindow.postMessage(JSON.stringify(msg), "http://
dev2.nextgened.com:8080");
 }
}
```

18 . Now create the receiveMessage function that takes a single argument we'll call e:

```
function receiveMessage(e) {
}
```

19 . Check the e.origin against dev2:

```
function receiveMessage(e) {
 if (e.origin == "http://dev2.nextgened.com:8080") {
  // The data can probably be trusted
  // It came from the inner frame as an acknowledgement
 }
}
```

20 . If it is dev2, JSON.parse the e.data:

```
serviceMessages = JSON.parse(e.data);
```

21 . Add the message to the messages block:

```
addMessage("dev2: " + serviceMessages.message);
```

22 . The final receiveMessage function should look something like this:

```
function receiveMessage(e) {
 if (e.origin == "http://dev2.nextgened.com:8080") {
  // The data can probably be trusted
  // It came from the inner frame as an acknowledgement
  serviceMessages = JSON.parse(e.data);
  addMessage("dev2: " + serviceMessages.message);
 }
}
```

23 . Save the file.

24 . Open inner.js. Create global variables for serviceMessages and divStatus:

```
var serviceMessages;
var divStatus;
```

25 . In the init method, assign divStatus to the status div block. The btnPressMe has already been assigned:

```
divStatus = document.getElementById("status");
```

26 . Assign the btnPressMe.onclick to be sendMessage, which we'll write shortly:

```
btnPressMe.onclick = sendMessage;
```

27 . Assign the window.onmessage handler to be receiveMessage, which we'll write shortly:

```
window.onmessage = receiveMessage;
```

28 . The final version of init should look like the following:

```
function init() {
 btnPressMe = document.getElementById("pressMe");
 divStatus = document.getElementById("status");
 btnPressMe.onclick = sendMessage;
 window.onmessage = receiveMessage;
}
```

29 . Create a sendMessage function, similarly testing for support of window.postMessage first:

```
function sendMessage() {
 if (typeof window.postMessage === "undefined") {
  alert("XDM is not supported on this browser!");
}
```

30 . In the else clause, we'll create a similar msg object to the last one, but it will be an acknowledgement message:

```
} else {
  var msg = new Object();
  msg.messageId = 1234;
  msg.messageType = "Ack";
  msg.message = "Acknowledged Order";
 }
```

31 . We'll use window.top.postMessage to post to the outermost window (this code is operating inside the iFrame) to post to files of dev1 origins:

```
window.top.postMessage(JSON.stringify(msg), "http://
dev1.nextgened.com:8080");
```

32 . The final version should look something like the following:

```
function sendMessage() {
 if (typeof window.postMessage === "undefined") {
  alert("XDM is not supported on this browser!");
 } else {
  var msg = new Object();
  msg.messageId = 1234;
  msg.messageType = "Ack";
  msg.message = "Acknowledged Order";
  window.top.postMessage(JSON.stringify(msg), "http://
dev1.nextgened.com:8080");
 }
}
```

33 . Now create a receiveMessage again with an "e" parameter. Test for origin of dev1:

```
function receiveMessage(e) {
  if (e.origin == "http://dev1.nextgened.com:8080") {
  }
}
```

34 . If so, use JSON.parse on the e.data and use addStatus to present the message:

```


   // The data can probably be trusted
   // It came from the mainPage
   serviceMessages = JSON.parse(e.data);
   addStatus("dev1: " + serviceMessages.message);
```

35 . The final version of receiveMessage should look like the following:

```
function receiveMessage(e) {
  if (e.origin == "http://dev1.nextgened.com:8080") {
   // The data can probably be trusted
   // It came from the mainPage
   serviceMessages = JSON.parse(e.data);
   addStatus("dev1: " + serviceMessages.message);
  }
}
```

36 . Save all of your files and open a browser to test with the dev1 host:

```
http://dev1.nextgened.com:8080/HTML5/crossdomain/solution/
mainPage.html
```