**Web Socket**

웹 소켓은 사용자의 브라우저와 서버 사이의 동적인 연결을 위한 기술입니다. 웹 소켓 API를 통해 서버로 메시지를 보내고 요청 없이 응답을 받아오는 것이 가능합니다.

웹 소캣은 ws 프로토콜 기반의 클리아언트와 서버 사이에서 full-duplex 연결 스트림을 설정할 수 있게 하는 기술입니다. 일반적인 웹 소켓 클라이언트는 플랫폼에 종속적인 프로토콜이 아닙니다만, 사용자의 브라우저에는 종속적일 수 있습니다.

웹 소켓 API는 [Window](https://developer.mozilla.org/ko/docs/Web/API/Window) DOM 객체나 직접 구현하는 객체로 JavaScript 코드에서 사용 가능하며, Web Workers 안에서도 사용 가능합니다.

Internet Explorer 10 및 JavaScript로 작성한 Windows 스토어 앱은 W3C(World Wide Web 컨소시엄)의 WebSocket API 사양에 정의된 대로 WebSocket API 지원을 추가합니다.

WebSockets 기술은 인터넷을 통한 양방향 통신을 위한 새로운 W3C JavaScript API 및 프로토콜을 제공합니다.

이 새로운 프로토콜을 사용하면 고정된 데이터 형식으로 더 쉽게 직접 작업할 수 있습니다.

이 프로토콜은 더 느린 문서 기반 HTTP 프로토콜을 무시합니다.

현재의 HTTP 표준 프로토콜은 서버에서 문서를 요청하고 웹 페이지를 표시하기 전에 문서가 전송되기를 기다려야 하기 때문에 속도가 느립니다.

WebSockets를 사용하면 텍스트, 이진 배열 또는 BLOB를 통해 데이터를 즉시 주고받을 수 있습니다.

WebSocket API는 단순하고 복잡하지 않으며 코드가 거의 필요하지 않습니다.

느린 대기 시간 양방향 데이터 교환을 손쉽게 활용하여 온라인 게임, 소셜 네트워크 즉시 알림, 재고 및 날씨 정보 실시간 표시, 기타 적시 데이터를 더 빨리 만들 수 있습니다.

**WebSockets 구현**

다음 단계를 수행하면 이 새로운 데이터 교환 기술을 간단하게 구현할 수 있습니다.

1. WebSocket 프로토콜을 구현하는 클라이언트 브라우저를 사용합니다.

2. 클라이언트 websocket을 만드는 웹 페이지에서 코드를 작성합니다.

3. Websocket을 통해 클라이언트 요청에 응답하는 웹 서버에서 코드를 작성합니다.

**WebSocket 클라이언트 사용**

**Internet Explorer 10에서는 다른 주요 브라우저와 마찬가지로 WebSocket 프로토콜을** 구현합니다.

[caniuse.com](http://go.microsoft.com/fwlink/p/?LinkId=228084) 웹 사이트에서 브라우저 지원의 현재 상태를 확인할 수 있습니다.

[IETF 도구](http://go.microsoft.com/fwlink/p/?LinkId=228085) 웹 사이트에 정의된 WebSocket 프로토콜은 다음과 같은 새로운 URL 체계를 사용합니다.

HTML :

**ws://**

**wss://**

**WebSocket 클라이언트 코드 작성**

웹 페이지 코드는 다음 작업을 해야 합니다.

1. WebSocket을 초기화하고 서버에 연결합니다.

2. 테스트를 통해 연결에 성공했는지 확인합니다.

3. 데이터를 보내고 받습니다.

**사용법**  
사용법도 간단합니다.  
  
Web Socket 을 생성하는 구문은 다음과 같습니다.

var wsSocker = **new WebSocket**("**ws://**Web Socket Server URL");

WebSocket 생성시 Web Socket Server 의 URL 을 Parameter로 사용하며 Web Socket 프로토콜 "**ws://**" 을 사용합니다.  
  
생성한 WebSocket 으로 send Method 를 사용하여 Message 를 전송합니다.

wsSocket.**send**("전송할 Message");

wsSocket interface 에는 다음과 같은 Event Handler 가 있습니다.

|  |  |
| --- | --- |
| **Event Handler** | **설명** |
| onmessage | Server 로 Message를 전달 받을때 |
| onopen | Web Socket Server 가 열릴 때 |
| onclose | Web Socket Server 가 닫힐 때 |

다음과 같이 사용합니다.

wsSocket.onmessage = function() {

...

};  
  
wsSocket.onopen = function() {

...

};  
  
wsSocket.onclose = function() {

...

};

다음 코드는 WebSocket URL을 정의하는 일반적인 코드를 보여 줍니다.

JavaScript:

**var host = 'ws://example.microsoft.com';**

다음 코드는 WebSocket에 연결하고 테스트를 통해 연결에 성공했는지 확인하는 방법을 보여 줍니다.

JavaScript :

**var host = "ws://sample-host/echo";**

**try {**

**socket = new WebSocket(host);**

**socket.onopen = function (openEvent) {**

**document.getElementById("serverStatus").innerHTML =**

**'WebSocket Status:: Socket Open';**

**};**

**socket.onmessage = function (messageEvent) {**

**if (messageEvent.data instanceof Blob) {**

**var destinationCanvas = document.getElementById('destination');**

**var destinationContext = destinationCanvas.getContext('2d');**

**var image = new Image();**

**image.onload = function () {**

**destinationContext.clearRect(0, 0,**

**destinationCanvas.width, destinationCanvas.height);**

**destinationContext.drawImage(image, 0, 0);**

**}**

**image.src = URL.createObjectURL(messageEvent.data);**

**} else {**

**document.getElementById("serverResponse").innerHTML =**

**'Server Reply:: ' + messageEvent.data;**

**}**

**};**

**socket.onerror = function (errorEvent) {**

**document.getElementById("serverStatus").innerHTML =**

**'WebSocket Status:: Error was reported';**

**};**

**socket.onclose = function (closeEvent) {**

**document.getElementById("serverStatus").innerHTML =**

**'WebSocket Status:: Socket Closed';**

**};**

**}catch (exception) { if (window.console) console.log(exception); }**

**}**

**function sendTextMessage() {**

**if (socket.readyState != WebSocket.OPEN) return;**

**var e = document.getElementById("textmessage");**

**socket.send(e.value);**

**}**

**function sendBinaryMessage() {**

**if (socket.readyState != WebSocket.OPEN) return;**

**var sourceCanvas = document.getElementById('source');**

**socket.send(sourceCanvas.msToBlob());**

**}**

이전코드에서는 **serverStatus**, **destination**, **serverResponse**, **textmessage** 및 **serverData**가 ID를 가진 요소로 웹 페이지에 포함되어 있다고 가정합니다.

F12 개발자 도구가 실행 중인 경우 캐시 결과가 콘솔 창에 표시됩니다.

텍스트 메시지 데이터를 보내려면 다음 코드 형식을 사용합니다.

JavaScript:

**var e = document.getElementById("msgText");**

**socket.send(e.value);**

이전 코드 예제에서는 ID를 가진 웹 페이지의 **msgText** 요소에 보내려는 메시지 텍스트가 있다고 가정합니다.

마찬가지로, **[onmessage](https://msdn.microsoft.com/ko-kr/library/hh772757(v=vs.85).aspx)** 이벤트를 사용하여 새 메시지를 검색하거나 send 메서드를 사용하여 메시지를 서버로 보낼 수 있습니다.

[**send**](https://msdn.microsoft.com/ko-kr/library/hh772767(v=vs.85).aspx) 메서드를 사용하여 텍스트, 이진 배열 또는 BLOB 데이터를 보낼 수 있습니다.

**WebSocket 서버 코드 작성**

소켓을 처리하는 서버 코드는 임의의 서버 언어로 작성할 수 있습니다.

선택한 언어에 관계없이 WebSocket 요청을 수락하고 적절하게 처리하는 코드를 작성해야 합니다.

**WebSocket 프로그래밍**

WebSocket은 WebSocket 프로그래밍에 사용할 수 있는 개체, 메서드 및 속성 집합을 제공합니다.

|  |  |  |
| --- | --- | --- |
| **이름** | **유형** | **설명** |
| [**WebSocket**](https://msdn.microsoft.com/ko-kr/library/hh772739(v=vs.85).aspx) | 개체 | 원격 호스트에 대한 양방향 채널을 제공합니다. |
| [**close**](https://msdn.microsoft.com/ko-kr/library/hh772748(v=vs.85).aspx) | 메서드 | WebSocket을 닫습니다. |
| [**send**](https://msdn.microsoft.com/ko-kr/library/hh772767(v=vs.85).aspx) | 메서드 | WebSocket을 사용하여 데이터를 서버로 보냅니다. |
| [**binaryType**](https://msdn.microsoft.com/ko-kr/library/hh772744(v=vs.85).aspx) | 속성 | [**onmessage**](https://msdn.microsoft.com/ko-kr/library/hh772757(v=vs.85).aspx)에 수신되는 이진 데이터 형식입니다. |
| [**bufferedAmount**](https://msdn.microsoft.com/ko-kr/library/hh772746(v=vs.85).aspx) | 속성 | send를 사용하여 대기 중인 데이터 바이트 수입니다. |
| [**extensions**](https://msdn.microsoft.com/ko-kr/library/hh772750(v=vs.85).aspx) | 속성 | 서버가 선택한 확장을 보고합니다. |
| [**onclose**](https://msdn.microsoft.com/ko-kr/library/hh772753(v=vs.85).aspx) | 속성 | 소켓을 닫을 때 호출되는 이벤트 처리기입니다. |
| [**onerror**](https://msdn.microsoft.com/ko-kr/library/hh772755(v=vs.85).aspx) | 속성 | 오류가 있을 때 호출되는 이벤트 처리기입니다. |
| [**onmessage**](https://msdn.microsoft.com/ko-kr/library/hh772757(v=vs.85).aspx) | 속성 | 메시지가 수신되었음을 알리는 이벤트 처리기입니다. |
| [**onopen**](https://msdn.microsoft.com/ko-kr/library/hh772759(v=vs.85).aspx) | 속성 | WebSocket이 연결된 경우 호출되는 이벤트 처리기입니다. |
| [**protocol**](https://msdn.microsoft.com/ko-kr/library/hh772760(v=vs.85).aspx) | 속성 | 서버가 선택한 프로토콜을 보고합니다. |
| [**readyState**](https://msdn.microsoft.com/ko-kr/library/hh772762(v=vs.85).aspx) | 속성 | WebSocket 연결 상태를 보고합니다. |
| [**url**](https://msdn.microsoft.com/ko-kr/library/hh772768(v=vs.85).aspx) | 속성 | 소켓의 현재 URL을 보고합니다. |

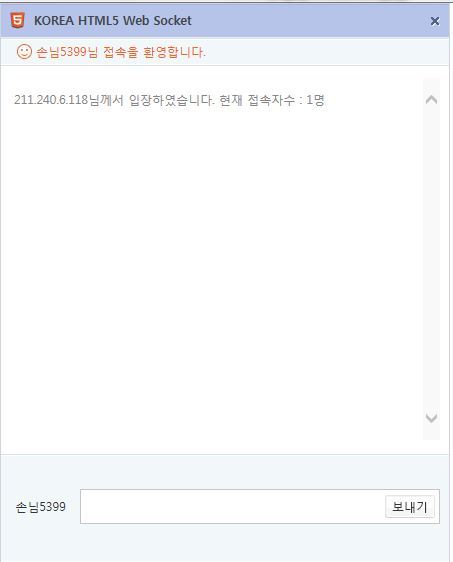
**API 참조**

https://developer.mozilla.org/ko/docs/WebSockets

웹소켓을 이용한 다중 채팅 예제 입니다.- html5 활용예제

소스다운로드

[[http://i1.daumcdn.net/cfs.tistory/resource/1519/blog/image/extension/zip.gif](http://incblue.tistory.com/attachment/cfile2.uf@2159F14854608CD31DAAD3.zip) webSocket\_chatting.zip](http://incblue.tistory.com/attachment/cfile2.uf@2159F14854608CD31DAAD3.zip)





접속하는 사용자를 체크해서 접속된 사용자의 수를 화면에 보여줍니다.

WebSocket 예제 2- html5 활용예제

소스다운로드

[[http://i1.daumcdn.net/cfs.tistory/resource/1519/blog/image/extension/zip.gif](http://incblue.tistory.com/attachment/cfile24.uf@2779904054608D6133E98B.zip) websocket2.zip](http://incblue.tistory.com/attachment/cfile24.uf@2779904054608D6133E98B.zip)





웹소켓 웹워커를 이용하여 만들어본 예제입니다.

소스다운로드

[[http://i1.daumcdn.net/cfs.tistory/resource/1519/blog/image/extension/zip.gif](http://incblue.tistory.com/attachment/cfile3.uf@27293A4554608DD111663A.zip) websocketEx.zip](http://incblue.tistory.com/attachment/cfile3.uf@27293A4554608DD111663A.zip)

