

<http://cafe.naver.com/solomade/1>

6 주소복사

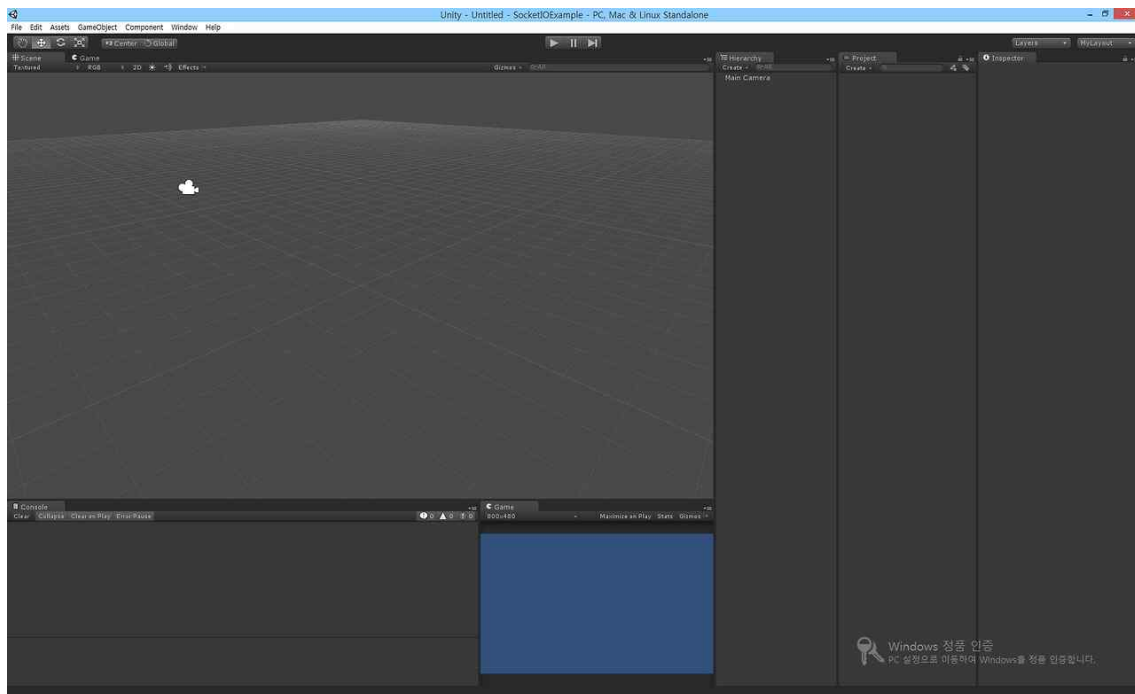
<http://mkgames.tumblr.com/post/90222375821/unity-socket-io>

mkgames님 포스팅을 올립니다. 중요한 이슈! 소켓io 버전이 @0.9.6에서 지원된다는 것..!!!!!!  
몇일을 헤맸는데.... 너무 감사합니다.mkgames님!

## Unity와 Socket.IO 연동하기

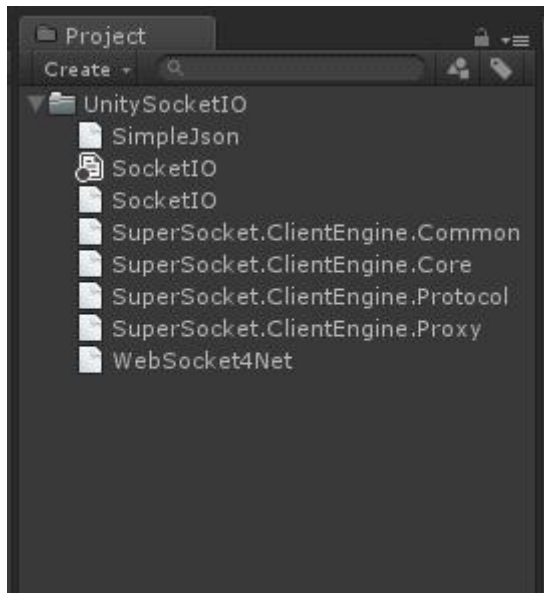
유니티에서 node.js의 socket.io와 통신하는 간단한 예제를 하나 살펴보자.

일단, 유니티에서 새로운 프로젝트를 하나 만든다. (프로젝트 이름을 SocketIOExample  
라고 지었다.)

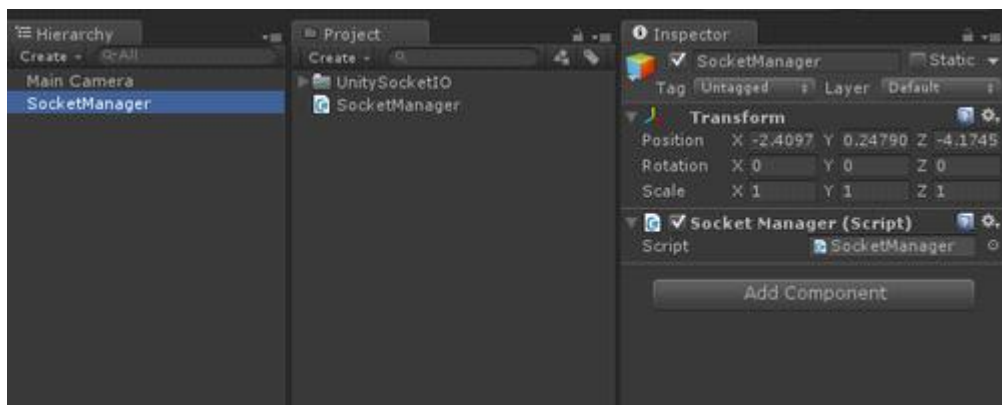


다음으로, GitHub의 NetEase님이 만든 유니티의 socket.io플러그인을 설치하자.  
<https://github.com/NetEase/UnitySocketIO>에서 받을 수 있다.다운을 받은 후에

“SocketIO\Bin\Debug\”폴더에 있는 모든 dll파일들을 유니티의 프로젝트(Assets폴더)에 복사하자. (UnitySocketIO라는 폴더를 하나 만들어서 넣었다.)



그리고 빈 게임 오브젝트와 C#스크립트를 각각 하나씩 만들고, 이름을 SocketManager 라고 지어준다. 그리고 이 스크립트를 게임오브젝트에 연결시켜준다.



SocketManager스크립트를 편집기로 열어서 다음과 같이 적어준다.

```
<ol class="linenums" style="margin-top: 0px; margin-bottom: 0px; color: rgb(174, 174, 174);">
```

```
    using UnityEngine
```

```
    using System.Collections
```

- `using SocketIOClient` // 이 네임스페이스를 반드시 추가해주어야 함.

- public class SocketManager MonoBehaviour

```
string      "http://127.0.0.1:999/"
public static ClientSocket get private set
```

```
void Awake
```

```
Socket new Client
Socket Opened Socket Opened
Socket Connect
```

```
private void SocketOpened object      System EventArgs
```

```
Debug Log "Socket Opened!"
```

```
void OnDisable
```

```
Socket Close
```

•

```
</ol>
```

**3행:** SocketIOClient 네임스페이스를 반드시 추가해주어야한다.

**7행:** url에 “http://[서버 IP주소]:[포트]/”를 입력한다. 끝에 “/”을 절대 빼먹지 않도록 한다. 이거 하나 때문에 거의 3시간을 허비했다.

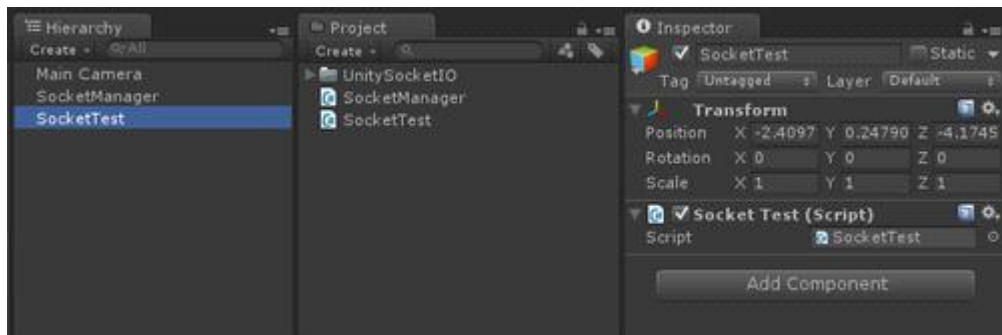
**10~15행:** 새로운 소켓을 하나 만들고, 연결하는 부분이다. 소켓이 연결되었을 때의 이벤트를 하나 설정해주었다.

**17~20행:** 소켓이 연결되었을 때의 발생될 이벤트.

**22~25행:** OnDisable이벤트가 발생 시에 소켓 연결을 끊는다.

(메시지를 받았을 때의 이벤트, 연결이 끊어졌을 때의 이벤트, 에러 발생시의 이벤트에 대해서도 알고 싶다면, <https://github.com/NetEase/UnitySocketIO>에서 참고할 수 있다.)

여기까지 되었다면, 또 빈 게임 오브젝트 하나와 C# 스크립트 하나를 각각 추가한다. (이름을 SocketTest라고 지어주었다.)



그리고 SocketTest스크립트에는 다음과 같이 작성한다.

```
<ol class="linenums" style="margin-top: 0px; margin-bottom: 0px; color: rgb(174, 174, 174);">
```

```
using UnityEngine
```

```
using System.Collections
```

```
• public class SocketTest : MonoBehaviour
```

```
{
```

```
    void Start()
```

```
    {
```

```
        void OnGUI()
```

```
        {
```

```
            if (GUILayout.Button("SEND"))
```

```
            {
```

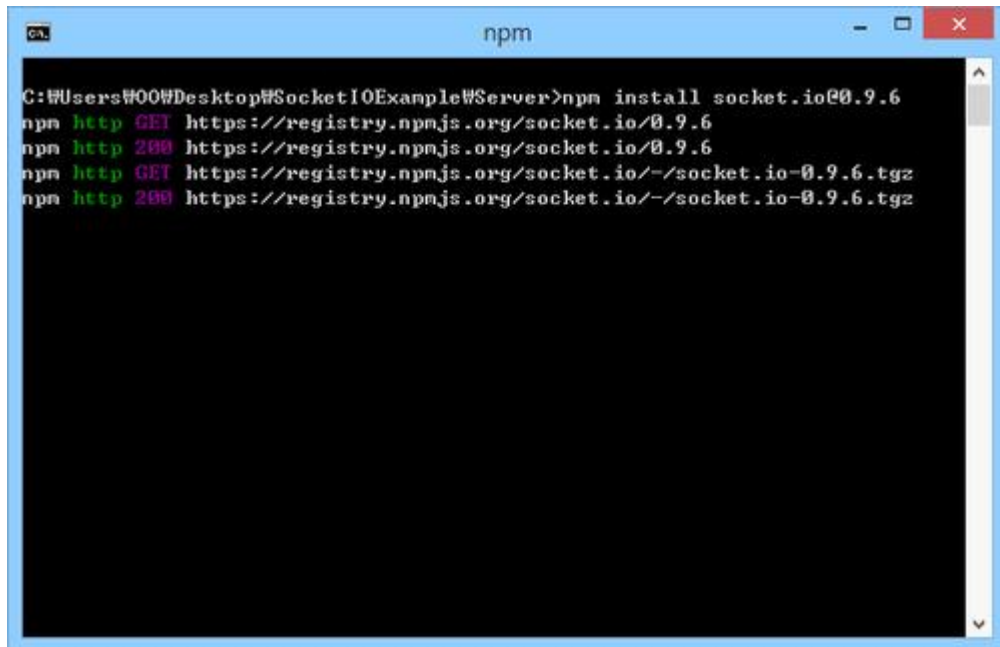
```
                //
```

**5~11행:** “MsgRes” 소켓이벤트가 발생시에, 람다식을 실행함. 서버측으로 부터 전송받은 데이터 정보가 data에 담기고, 데이터는 data.Json.args[0]에서 확인할 수 있다.

**13~17행:** SEND 버튼을 하나 만들고, 클릭 시에, 서버측으로 “Msg”소켓 이벤트를 발생시키면서 “Hello, World”라는 문자열 데이터를 함께 보냄.

이제 서버측을 살펴 보도록 하자. 먼저 node.js의 socket.io를 설치하고, 간단한 코드를 작성해보자.

SocketIOExample폴더에서 Server라는 폴더를 하나 생성 후에 그안에서 [Shfit+오른쪽 버튼 클릭]을 해서 [여기서 명령 창 열기(W)]를 누른다.cmd창이 뜨면 **npm install socket.io@0.9.6**을 입력 후 엔터를 누르면, socket.io 0.9.6버전이 설치가 된다. (\*최신버전은 작동안함)



그리고 같은 폴더에서 새로운 텍스트 파일을 하나 만들어서 이름을 server.js라고 해준다. (확장자는 반드시 .js이어야 함)

server.js안에 코드를 다음과 같이 작성한다.

```
<ol class="linenums" style="margin-top: opx; margin-bottom: opx; color: rgb(174, 174, 174);">
```

```
    var    require "./socket.io"          999
```

```
        "connection" function
```

```
            "A user connected !"
```

```
        "Msg" function
```

```
            "MsgRes"
```

```
    </ol>
```

**1행:** socket.io를 사용하기 위한 모듈을 불러오고, 999번 포트에서 소켓을 연다. (포트는

위의 유니티에서 지정한 포트와 일치해야함)


**3행:** 클라이언트가 연결되었을 때, 콜백을 실행시킨다.

**4행:** 클라이언트가 연결되면, console에 메시지를 출력함.

**6~8행:** “Msg”소켓 이벤트가 발생시에, 콜백이 실행되고, 클라이언트측으로 “MsgRes”이벤트를 발생시킴.

그러면 이제 정상적으로 작동을 하는지 테스트를 한번 해보자.

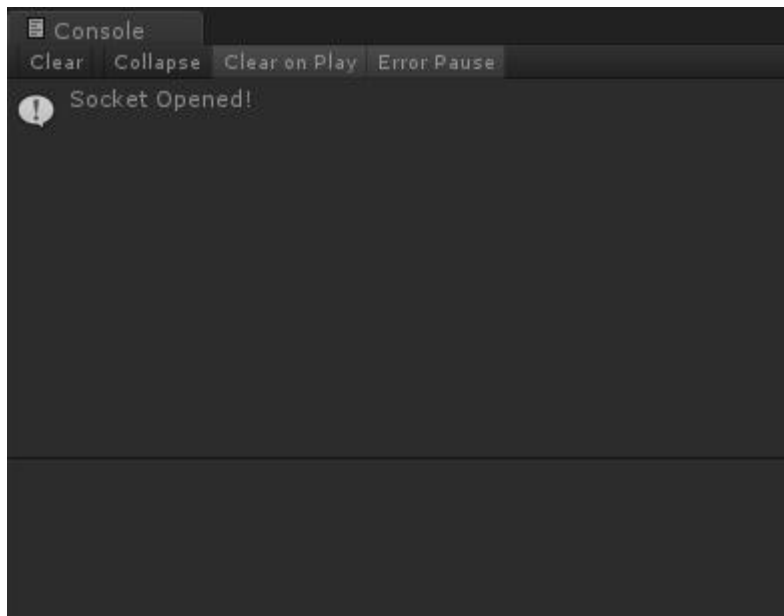
먼저 cmd창에서 “node server”를 입력해서 서버를 작동시킨다.



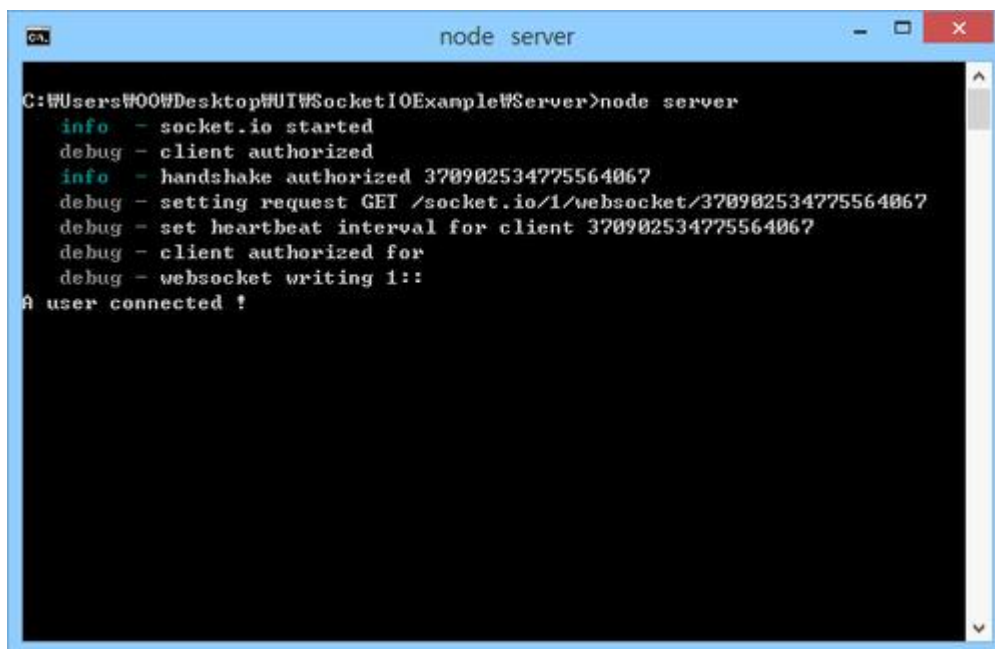
```
node server
C:\Users\#00\Desktop\UI\SocketIOExanple\Server>node server
info - socket.io started
```

그리고 유니티에서 플레이 버튼을 클릭한다.

콘솔창에 다음과 같은 메시지가 출력이 되면 정상적으로 서버에 소켓이 연결되었다는 것이다. (SocketManager의 SocketOpened이벤트가 정상작동 한 것임)

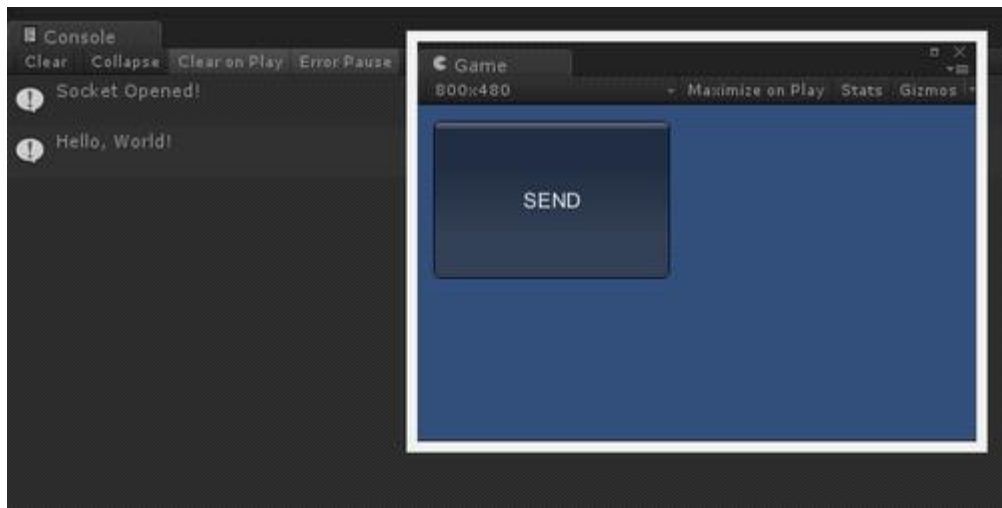


동시에 서버측의 콘솔 또한 다음과 같은 메시지가 뜨면 클라이언트가 정상적으로 연결이 되었다는 것이다.



이제 SEND버튼을 한번 눌러보자.

그러면 다음과 같이 콘솔에 “Hello, World!”라는 메시지가 정상적으로 출력될 것이다.



또한 서버측도 마찬가지로 클라이언트로부터 정상적으로 신호를 받아서 “Hello, World!”라는 메시지가 출력되었을 것이다.

```

C:\Windows\system32\cmd.exe - node server
C:\Users\00\Desktop\UI\SocketIOExample\Server>
C:\Users\00\Desktop\UI\SocketIOExample\Server>
C:\Users\00\Desktop\UI\SocketIOExample\Server>
C:\Users\00\Desktop\UI\SocketIOExample\Server>
C:\Users\00\Desktop\UI\SocketIOExample\Server>
C:\Users\00\Desktop\UI\SocketIOExample\Server>node server
  info - socket.io started
  debug - client authorized
  info - handshake authorized 3416174331344655775
  debug - setting request GET /socket.io/1/websocket/3416174331344655775
  debug - set heartbeat interval for client 3416174331344655775
  debug - client authorized for
  debug - websocket writing 1::
A user connected !
Hello, World!
  debug - websocket writing 5:::{"name":"MsgRes","args":["Hello, World!"]}
  
```