

SignalR 기초와 활용

최흥배

<https://github.com/jacking75/choiHeungbae>

이 문서는 아래의 글을 많이 참조 했습니다^^;

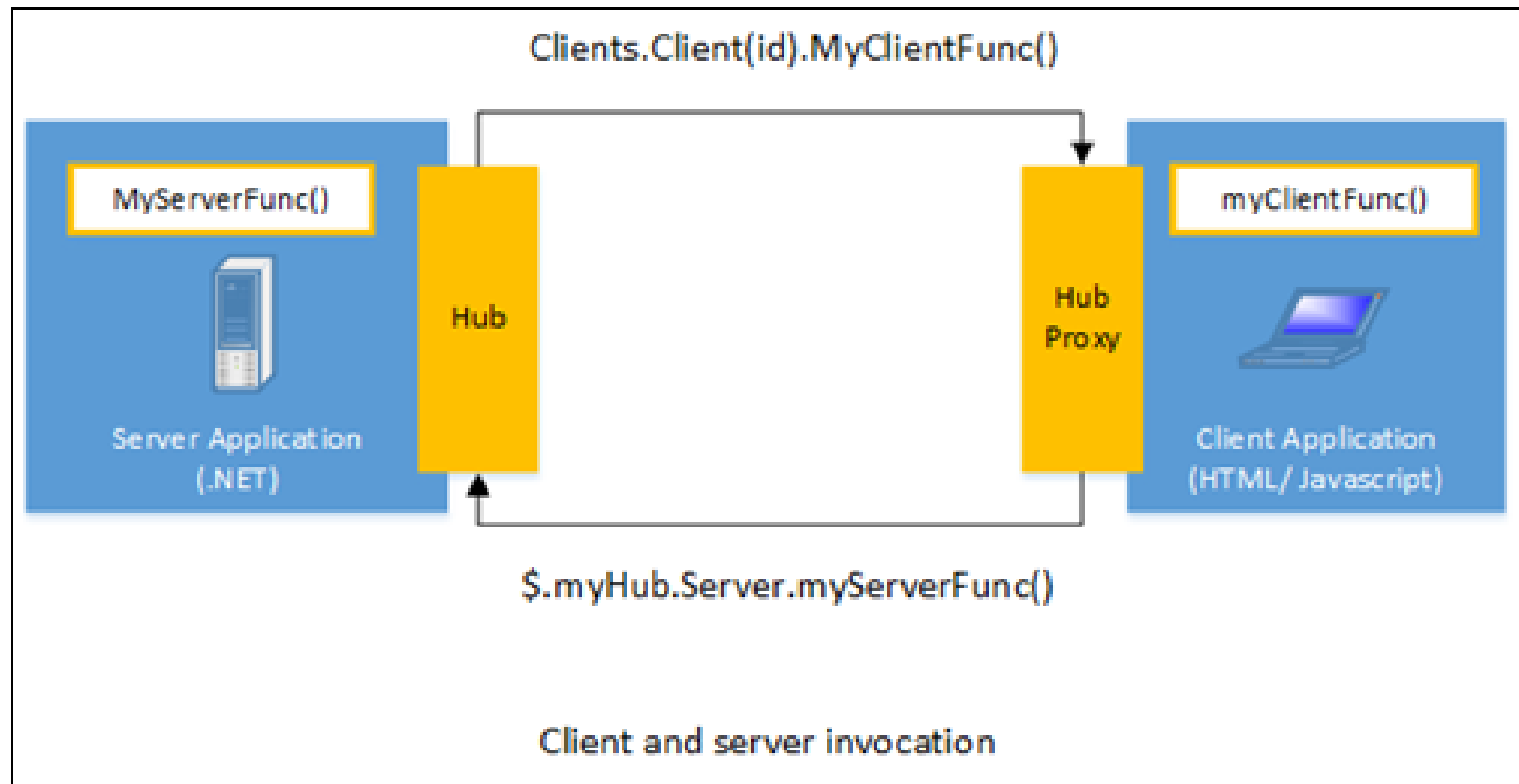
ASP.NET SignalRを知る

<http://www.atmarkit.co.jp/ait/articles/1303/19/news099.html>

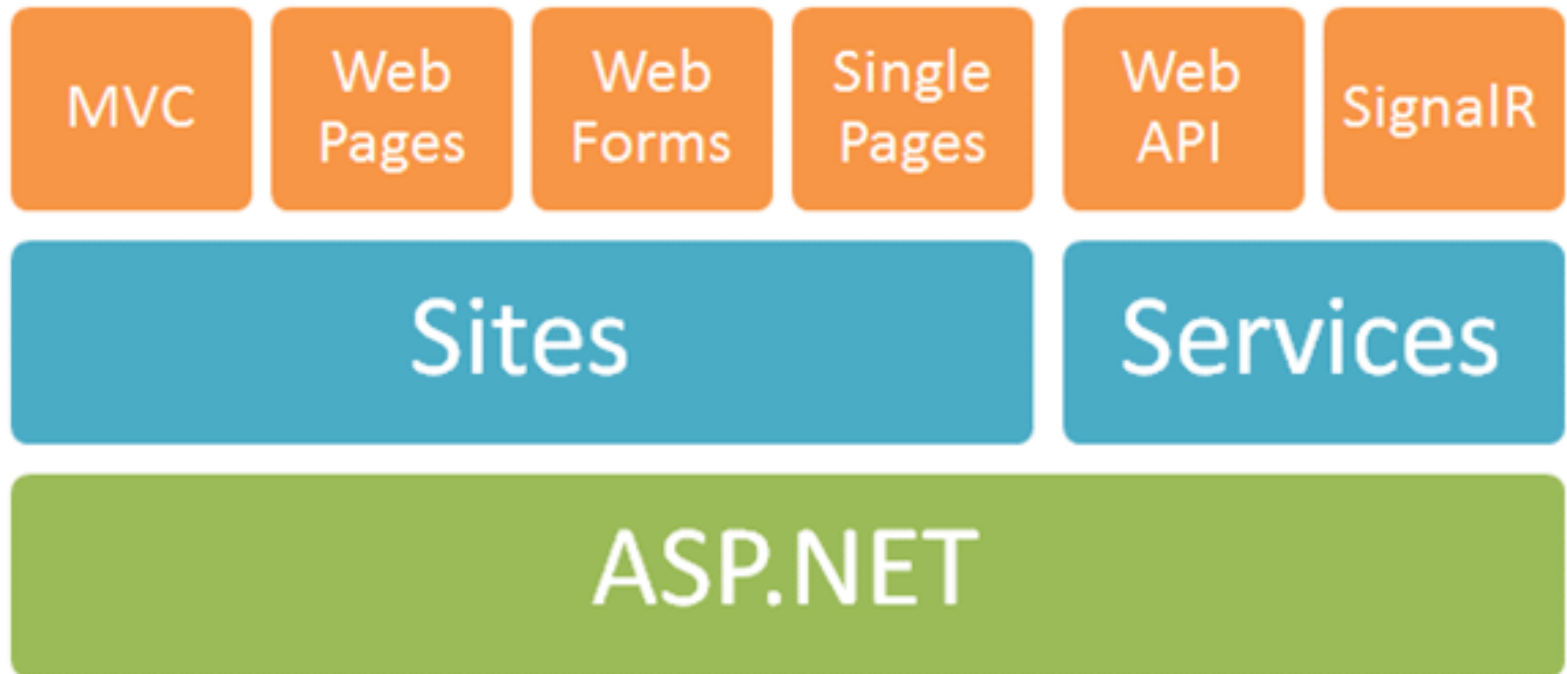
SignalR 이란? (1/3)

- 최신 ASP.NET에 추가된 라이브러리.
이것을 사용하면 **리얼타임,비동기,쌍방향 통신(Push나 RPC) 기능**을 Web 애플리케이션에 쉽게 사용할 수 있다.
- **NuGet**을 통해서 입수할 수 있다. **Open Source**
- 클라이언트/서버 간에 영속적 연결. 일반 웹은 비 영속적 연결
- 트랜스포트로 자동으로 네고에이션.
웹서버나 클라이언트에 따라서 **WebSocket, Server-Send Events, Forever Frames, Long Polling** 사용
- 하나의 서버 당 수천의 접속을 비동기로 처리한다.
- 비슷한 기술로 Node.js가 있다.
- ASP.NET 개발팀의 David Fowler, Damian Edwards가 개인 프로젝트로 시작하다가 정식으로 ASP.NET에 들어옴.
- 최신 버전은 2.0

SignalR 이란? (2/3)



SignalR 이란? (3/3)



SignalR 샘플

- ASP.NET SignalR Stock Ticker Sample
<http://chackr.azurewebsites.net/SignalR.Sample/StockTicker.html>
- SignalR ShootR
<https://github.com/NTaylorMullen/ShootR>

ASP.NET SignalR Stock Ticker Sample

Live Stock Table

Symbol	Price	Open	High	Low	Change	%
GOOG	569.98	570.3	571.1	569.42	▼ -0.32	-0.06%
MSFT	32.00	30.31	32.03	30.31	▲ 1.69	5.28%
APPL	577.52	578.18	578.18	576.81	▼ -0.66	-0.11%

Live Stock Ticker

1.69 (5.28%) **APPL** 577.52 ▼ -0.66 (-0.11%)



RPC 통신 (1/2)

- WebSocket, HTTP 위에서 동작하는 RPC(리모트 프로시저 콜)을 구현.
- 클라이언트에서 서버에 만들어진 함수를 호출하는 느낌으로 서버측 API를 실행할 수 있다.
- SignalR은 클라이언트에서 호출 가능한 함수 집합을 Hub라고 부른다. 클라이언트에서는 허브의 프록시 오브젝트를 만들어서 함수를 호출하는 형식이다.
- 물론 함수의 반환 값도 얻을 수 있다. 다만 통신은 비동기 이므로 반환 값은 JQuery로 만들어진 Deferred 오브젝트를 사용하여 얻는다.

RPC 통신 (2/2)

```
<script type="text/javascript">
    $(function() {
        var connection = $.hubConnection();
        var sample = connection.createHubProxy("sample");

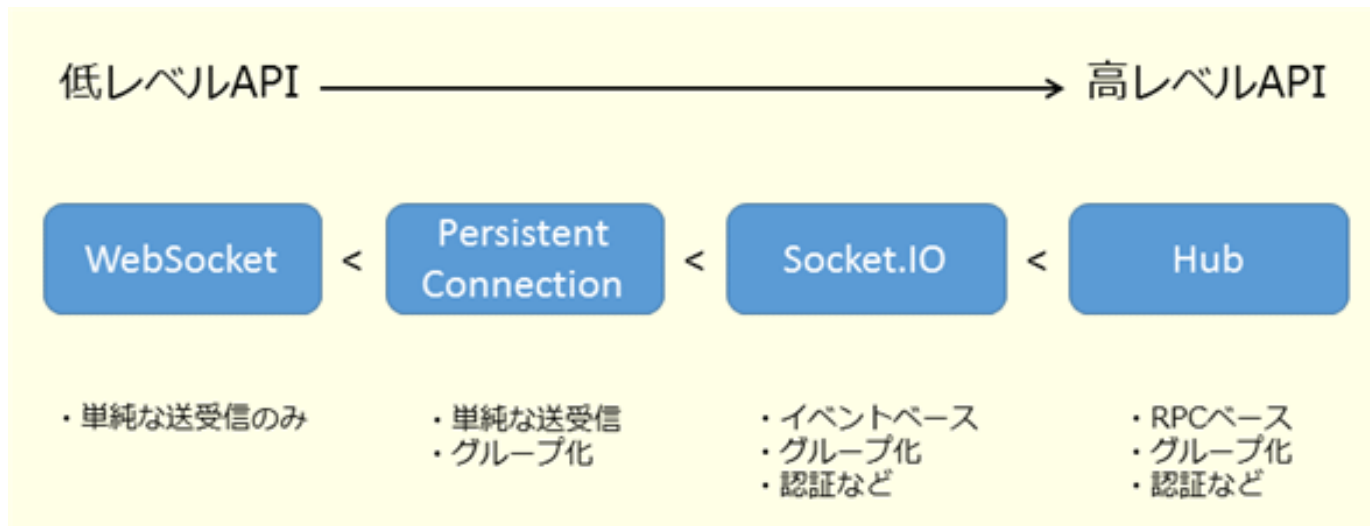
        connection.start(function () {
            sample.invoke("Say", "Hello, world");
        });
    })
</script>
```

```
<script type="text/javascript">
    $(function() {
        var connection = $.hubConnection();
        var sample = connection.createHubProxy("sample");

        connection.start(function () {
            sample.invoke("Say", "Hello, world").done(function(result) {
                alert(result);
            });
        });
    })
</script>
```


PersistentConnection와 Hub (1/2)

- PersistentConnection과 Hub 라는 두 개의 API가 있다.
- PersistentConnection는 이름대로 서버와 클라이언트 간의 '지속적인 접속'을 제공하면서 클라이언트를 관리하는 기능을 가지고 있다. 통신 기능면에서는 하나의 메시지를 송수신하는 기능 밖에 없는 저 레벨 API 이다.
- Hub는 PersistentConnection 위에 구현된 것으로 고 레벨 API이다. 클라이언트/서버 간의 함수 호출이라는 기본적인 기능을 제공한다. HubPipeline를 이용하여 클라이언트 인증을 구현. 물론 자체 확장도 가능하다.

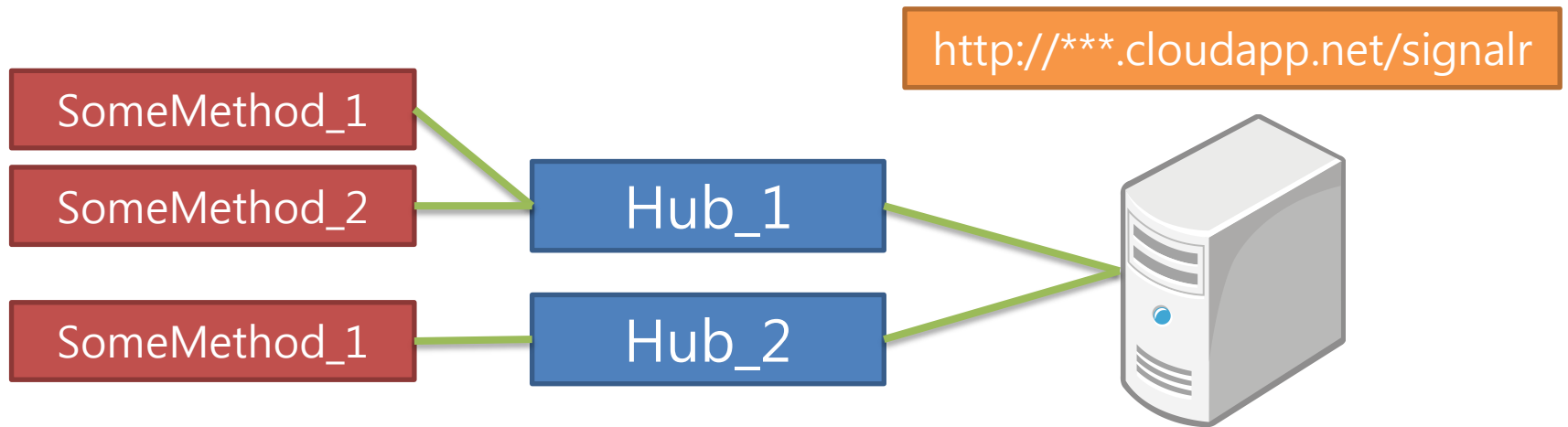


PersistentConnection와 Hub (2/2)

- 보통 PersistentConnection 보다 Hub API를 사용한다.
- Hub는 RPC를 사용하기 때문에 PersistentConnection와 비교하면 리퀘스트 해석이나 리플렉션을 사용하기 때문에 오버헤드가 더 크다.
- 고 성능을 원한다면 PersistentConnection, 빠른 개발을 원한다면 Hub를 사용한다.

Hub (1/3)

- 클라이언트에서 호출 가능한 함수의 집합



Hub (2/3)

- 구현 예

```
[HubName("chat")]  
public class ChatHub : Hub  
{  
    {  
        public void SendMessage(string text)  
        {  
            Clients.ReceiveMessage(text);  
        }  
    }  
}
```

클라이언트에 공개하는 이름

클라이언트에서 호출 가능한 함수

Hub (3/3)

Hub 클래스에 구현되어 있는 프로퍼티

Context

현재 요청 정보를 유지

Groups

그룹을 관리하는 클래스

Clients

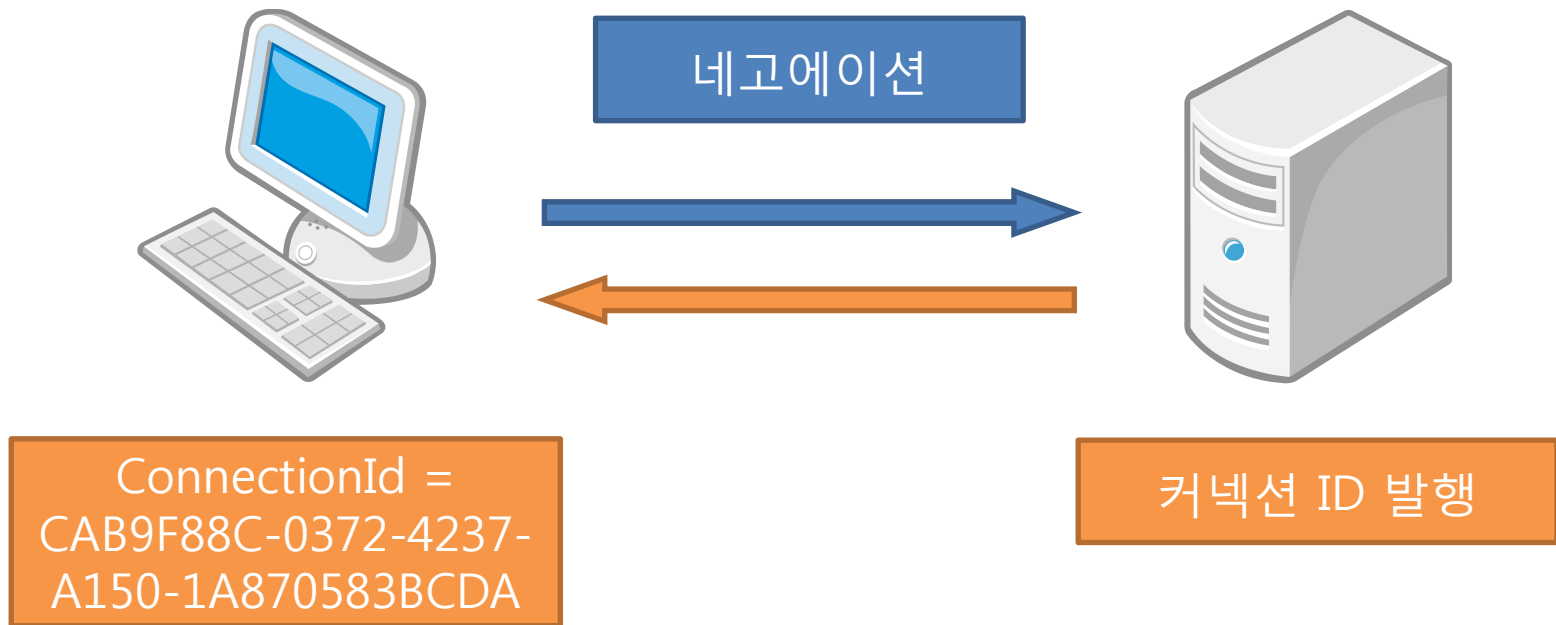
SignalR에서 관리하고 있는 모든 클라이언트를 뜻한다

Caller

리퀘스트를 요청한 클라이언트를 뜻한다.

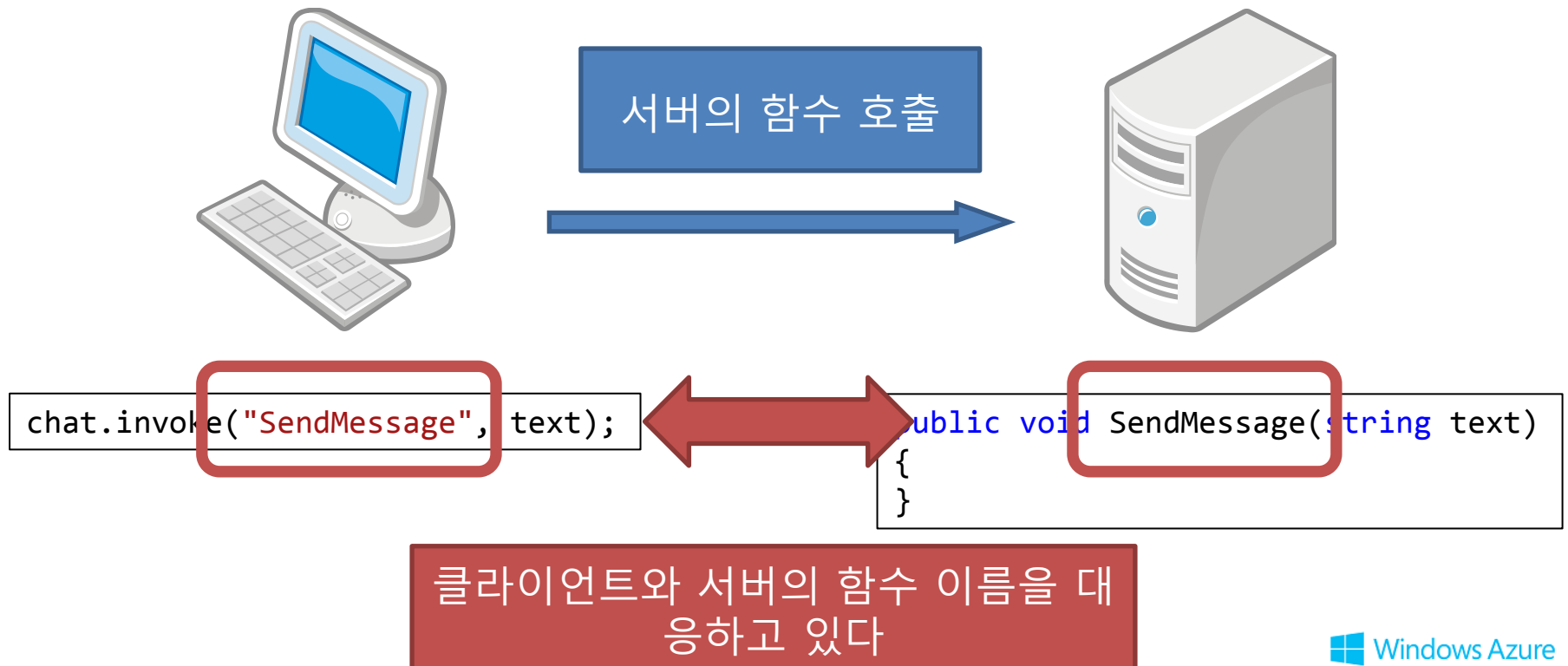
서버와 클라이언트 접속 흐름 (1/6)

- SignalR 이 클라이언트 관리
 - 클라이언트 마다 유일한 커넥션 ID 발행



서버와 클라이언트 접속 흐름 (2/6)

- 서버에 준비된 함수를 호출한다
 - 클라이언트는 반환 값을 얻을 수 있다

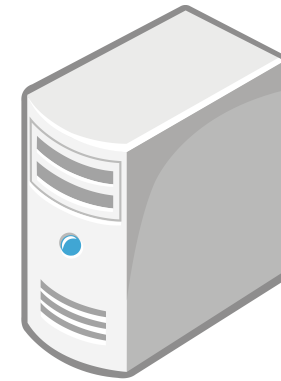


서버와 클라이언트 접속 흐름 (3/6)

- 클라이언트에 등록된 함수를 호출한다.
 - 서버는 반환 값을 얻을 수 없다.



클라이언트 함수 호출



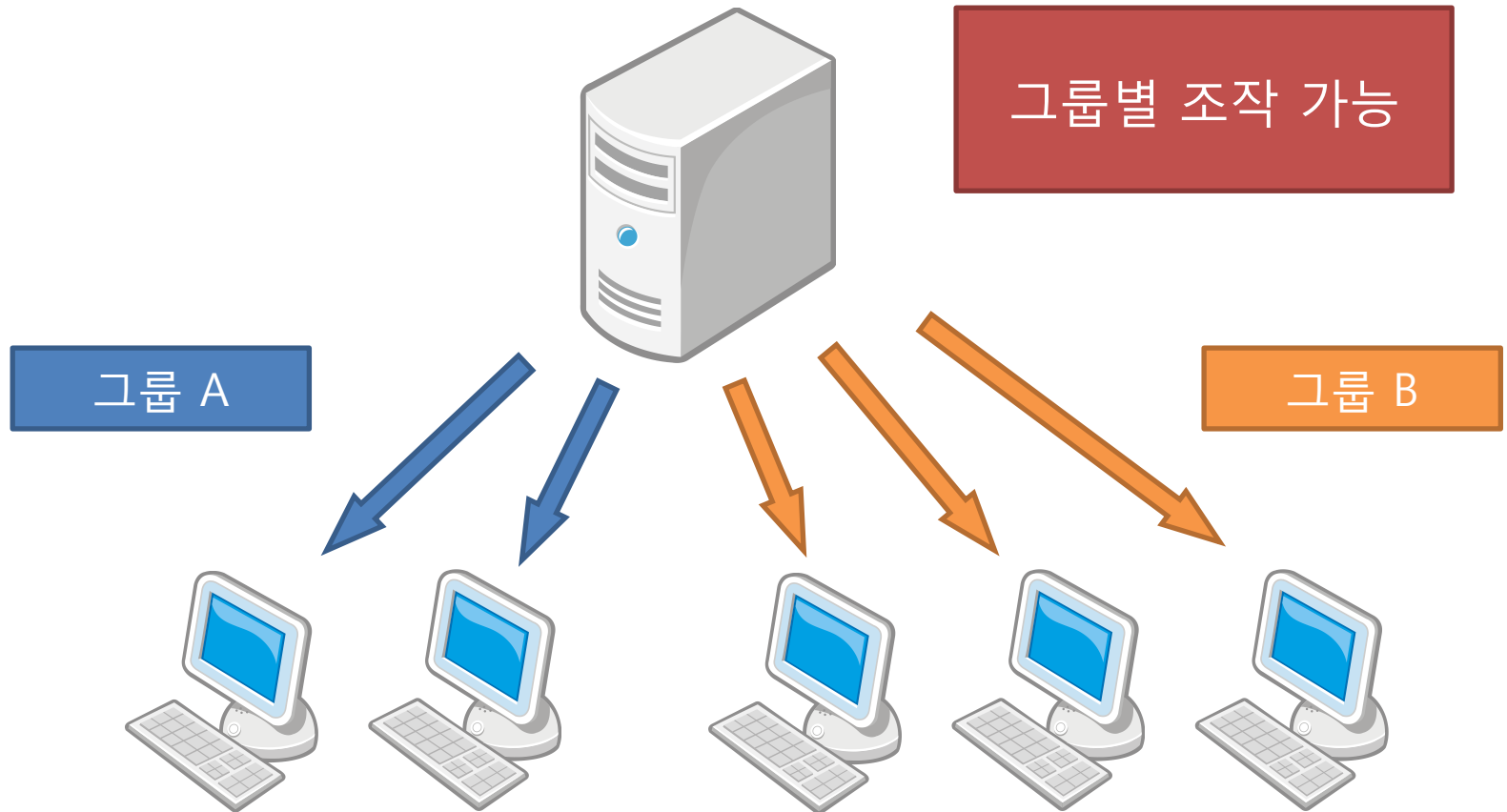
```
chat.on("ReceiveMessage", function(text)
```

```
clients.ReceiveMessage(text);
```

클라이언트와 서버의 함수 이름을 대응하고 있다.

서버와 클라이언트 접속 흐름 (4/6)

- 클라이언트를 그룹화 하여 관리



서버와 클라이언트 접속 흐름 (5/6)

- 그룹에 접속을 추가하는 것을 구현

```
[HubName("chat")]  
public class ChatHub : Hub  
{  
    public void JoinGroup(string groupName)  
    {  
        Groups.Add(Context.ConnectionId, groupName);  
    }  
  
    public void LeaveGroup(string groupName)  
    {  
        Groups.Remove(Context.ConnectionId, groupName);  
    }  
}
```


그룹에 추가

그룹에서 삭제

서버와 클라이언트 접속 흐름 (6/6)

- 그룹에 함수 호출

```
[HubName("chat")]  
public class ChatHub : Hub  
{  
    public void SendMessage(string groupName, string text)  
    {  
        Clients[groupName].ReceiveMessage(text);  
    }  
}
```



그룹에 대해서 함수를 호출

다양한 클라이언트 지원

- SignalR은 JavaScript 이외의 클라이언트에서도 사용할 수 있다.
- Windows 8 RT, .NET Framework, iOS, Mac, Android 등

```
class Program
{
    static void Main(string[] args)
    {
        Start();
    }
}
```

현재 Xamarin에 의해서 iOS와 Android 지원.

장래에는 C++ 지원 예정.

```
private static async void Start()
{
    var connection = new HubConnection("http://example.com/signalr");
    var sample = connection.CreateHubProxy("sample");

    await connection.Start();

    var result = await sample.Invoke<string>("Say", "Hello, world");

    Console.WriteLine(result);
}
}
```



erizet / **SignalA**

A SignalR client for Android.

74 commits

5 branches

0 releases

1 contributor



branch: master ▾

SignalA / +

Update readme.md



erizet authored 3 months ago

latest commit 7c0962b723

Demo	relative path to http-client	6 months ago
HubDemo	fixed strings	4 months ago
HubGroupDemo	updated demo	3 months ago
SignalA.LongPolling	increased version	3 months ago

<https://github.com/erizet/SignalA>



DyKnow / SignalR-ObjC



Objective-C Client for the SignalR Project works with iOS and Mac <http://dyknow.github.com/SignalR-ObjC/>

359 commits

3 branches

10 releases

6 contributors



branch: master ▾

SignalR-ObjC / +

Merge pull request #152 from 0xcd/NSJSONSerialization ...



abillingsley authored 11 days ago

latest commit fd41ca5d2e



SignalR.Client.ObjC.xcwork...	Update Dependencies to Latest AFNetworking 2.0.	2 months ago
SignalR.Client.ObjC	Update Dependencies to Latest AFNetworking 2.0.	2 months ago
SignalR.Client	Remove AnyJSON dependency by using 'NSJSONSerialization' instead	11 days ago
.gitignore	add podfile to manage dependencies	a year ago
ACKNOWLEDGEMENTS.md	Remove AnyJSON dependency by using 'NSJSONSerialization' instead	11 days ago

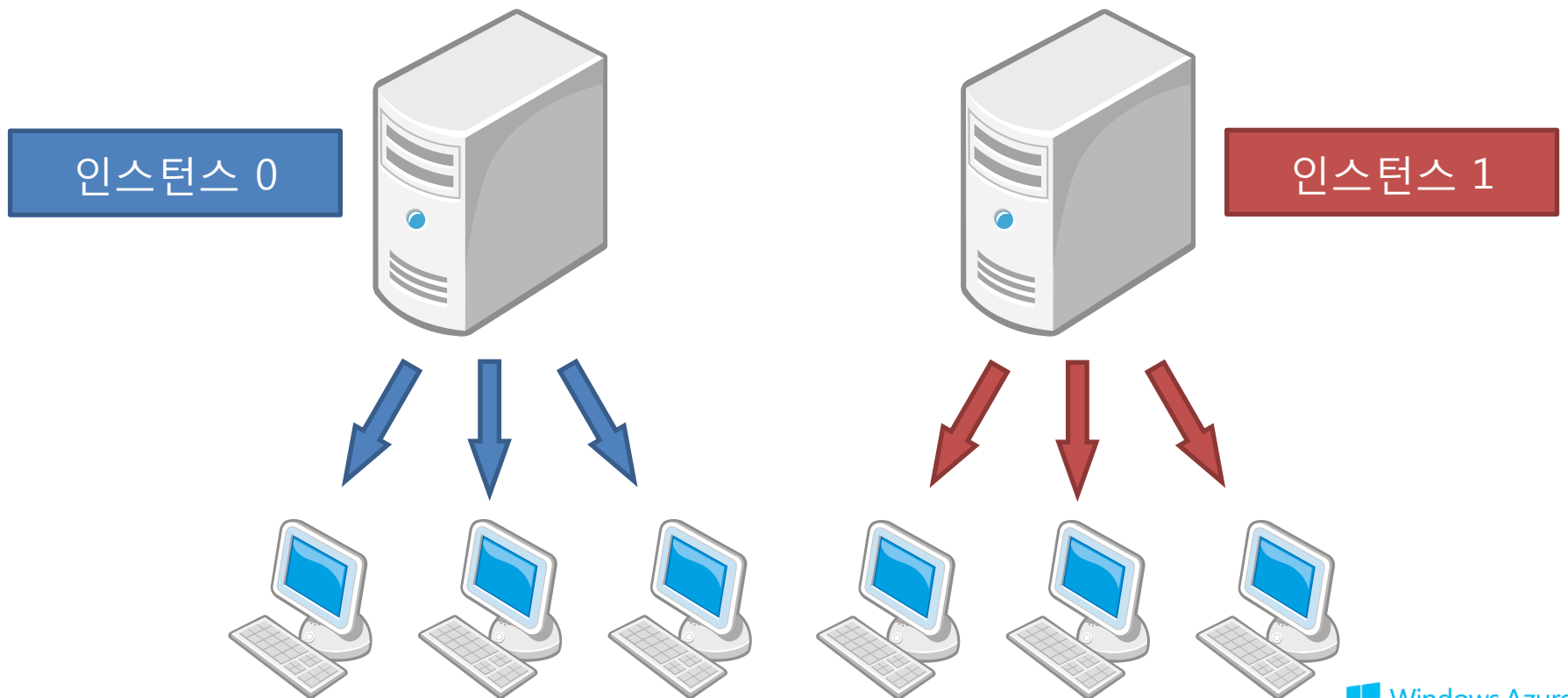
<https://github.com/DyKnow/SignalR-ObjC>

개발 환경

- Visual Studio를 사용한다.
 - **Visual Studio 2012의 IIS Express는 WebSocket 대응.**
 - SignalR을 IIS가 아닌 일반 .NET 애플리케이션에서 서버로 사용 가능(**Self-Host**).
 - Visual Studio Express에서도 개발할 수 있다.
 - Windows Azure Web 사이트에 배포 가능
-
- **WebSocket을 사용하기 위해서는 Window Server 2012(Windows 8) 이상에서 지원하는 최신 IIS + .NET Framework 4.5 이상을 사용해야 한다.**
 - **클라이언트의 경우 WebSocket을 지원하는 웹브라우저에서는 WebSocket으로 통신. 그러나 .NET 클라이언트에서는 Win8 이상에서만 WebSocket으로 통신**

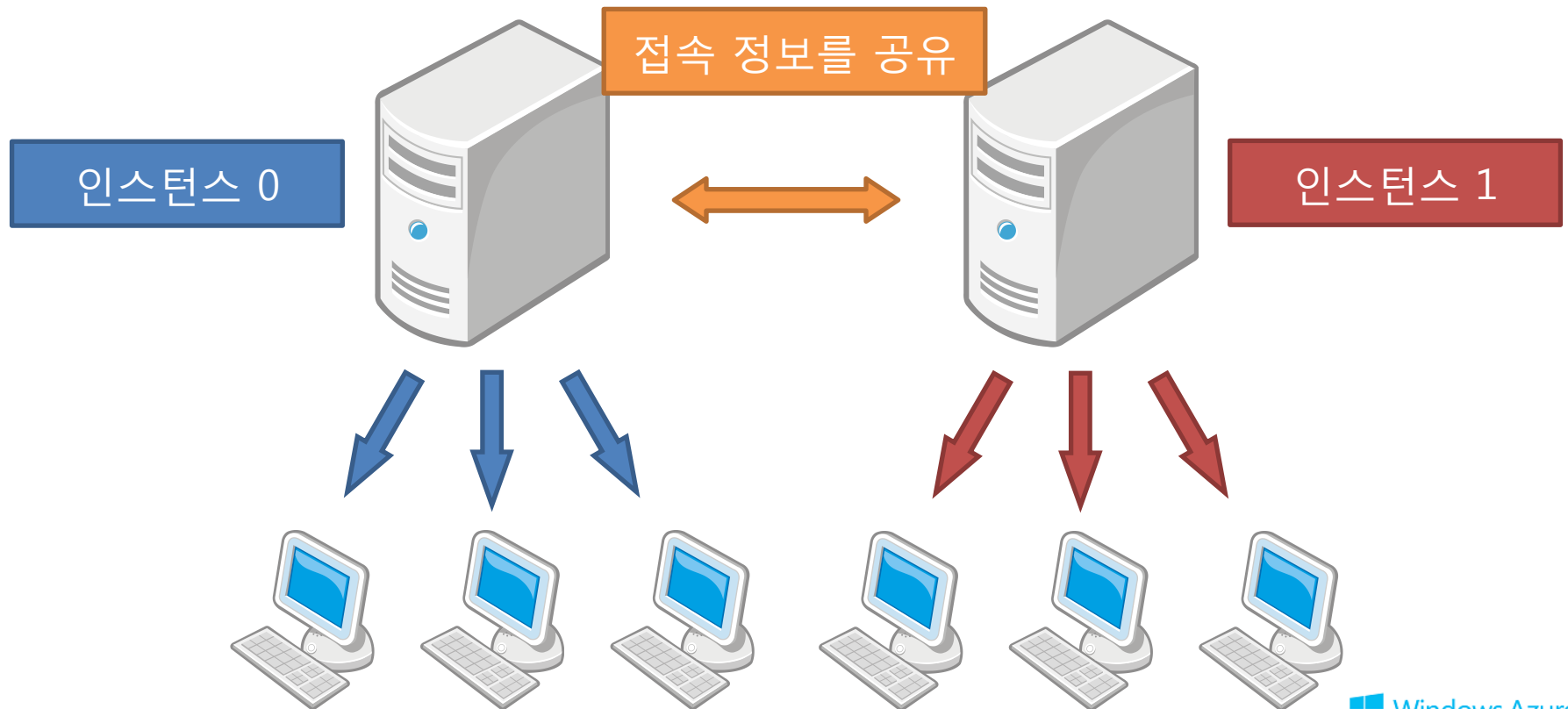
Scale Out (1/5)

- 영속적인 연결을 인스턴스 마다 생성
 - 단순히 수를 늘리는 것만으로 대응할 수 없다



Scale Out (2/5)

- 인스턴스간에 접속을 공유해야 한다
 - 메시징을 이용한다



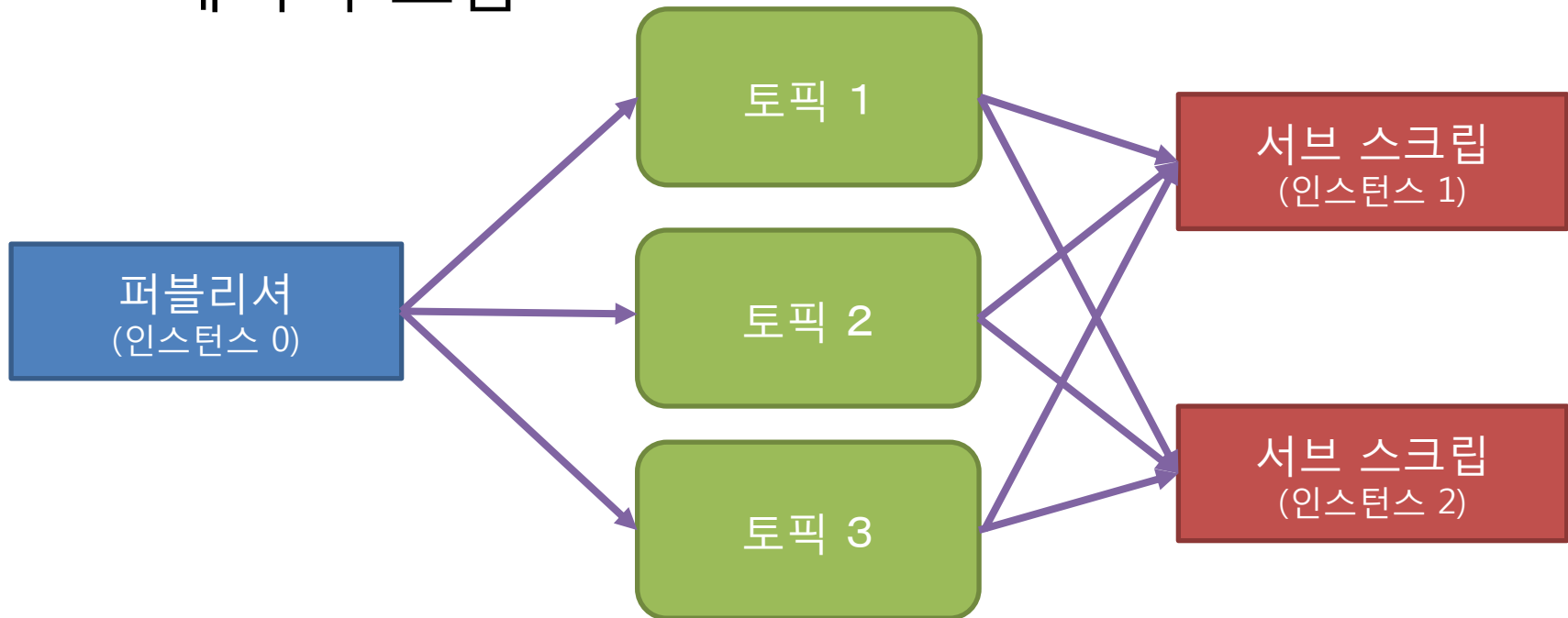
Scale Out (3/5)

- SignalR을 Scale Out 하기 위한 방법으로 Redis, ServiceBus (Azure용), SQL Server 를 사용한다.
- 모두 NuGet을 통해 입수 할 수 있다.

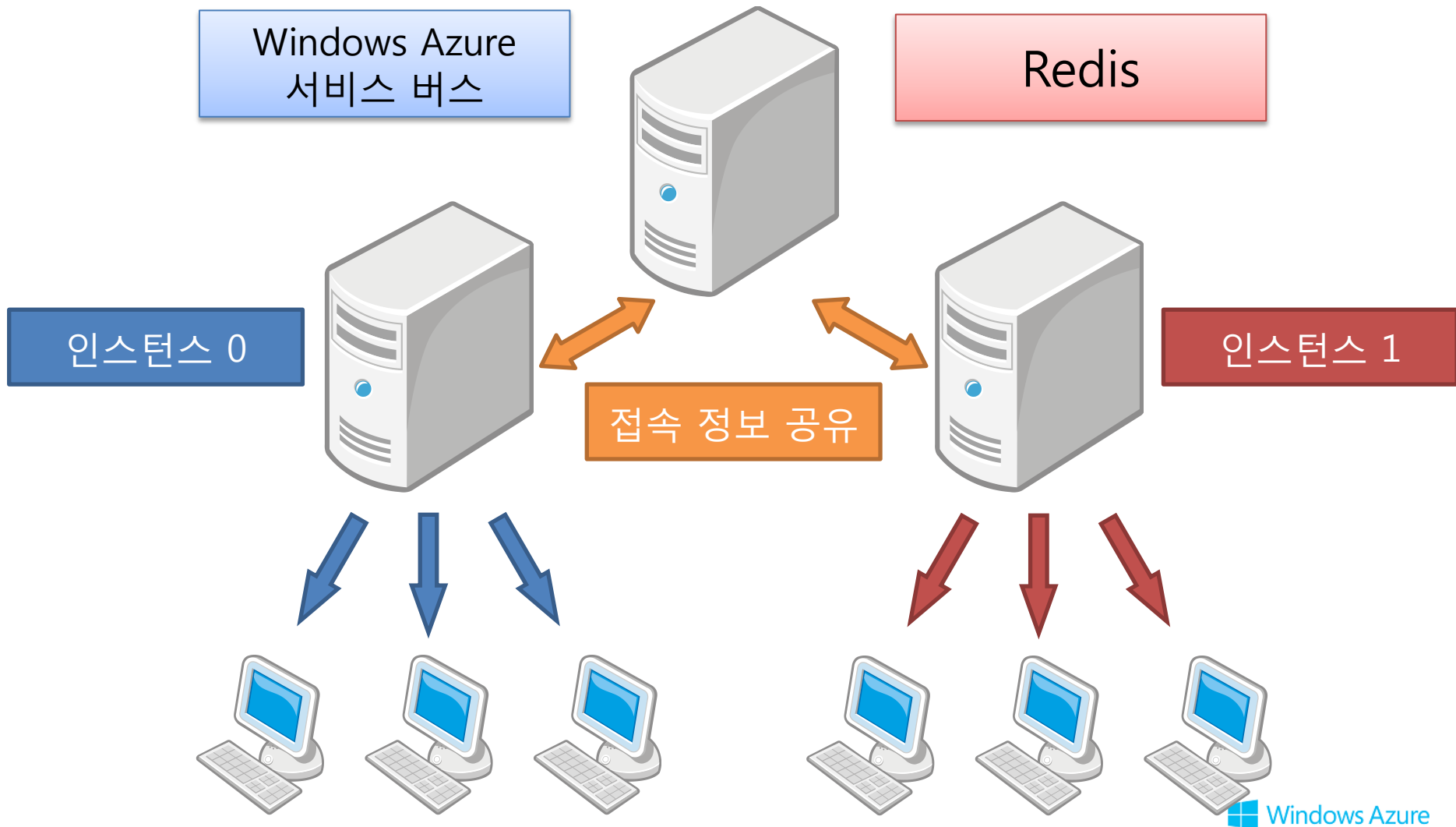
Scale Out (4/5)

Pub/Sub 형 메시징

- Windows Azure 서비스 버스
 - 메시지 흐름



Scale Out (5/5)



SignalR vs Node.js

- 기능적으로 비슷하다.
- Node.js는 JavaScript를 사용하고, SignalR은 C# 및 Visual Basic.NET 사용.
- Windows 플랫폼 프로그래머, 한국의 게임 프로그래머에게는 SignalR 사용이 쉬움.
- JavaScript를 잘 알고 있거나 혹은 앞으로 자주 사용한다면 Node.js가 좋은 선택이 되지만 그렇지 않다면 (게임 프로그래머의 경우) C# 및 Windows 플랫폼 사용이 훨씬 쉽기 때문에 SignalR을 선택하는 것이 더 좋음.
- Windows 라이선스 문제는 로직 부분에만 사용하면 큰 문제가 안되고, 특히 클라우드 플랫폼을 사용한다면 크게 문제 되지 않음.
Linux는 그냥 사용할 수 있는 쉬운 OS가 아님.
- Node.js는 npm을 통해 풍부한 외부 라이브러리를 사용할 수 있고, SignalR은 NuGet을 통해 풍부한 외부 라이브러리를 사용할 수 있다.
- SignalR도 Mono를 사용하여 Linux에서 사용할 수 있다.

SignalR과 Node.js 비교표

node.js + socket.io	SignalR
개발 언어 JavaScript	개발 언어 C#, VB
서버 node.js	서버는 IIS
싱글 스레드로 동작	멀티 스레드로 동작
논 블럭킹 I/O을 사용한 비동기	스레드 풀을 사용한 비동기
이벤트 베이스 API	RPC베이스의 API
Redis를 사용한 Scale Out	Redis와 서비스 버스를 사용한 Scale Out
공식 클라이언트는 JavaScript	공식 클라이언트는 JavaScript와 C#
컴파일 불 필요	컴파일 필요
일본어 정보가 많음	일본어 정보가 적음
운용 사례가 많은	운용 사례가 적음
node.js 전용의 호스팅 서비스가 있음	IIS 8과 .NET 4.5를 사용하는 호스팅 서비스가 아직은 적음(일본)

SignalR의 장점

- 간단한 클라이언트/서버 API
- 비동기·멀티 스레드로 동작
- 최신·최적의 통신 방법을 자동으로 선택
- Scale Out이 아주 쉬움(?)
- 클라이언트 그룹 관리 제공

프로그래머는 로직 구현에만 집중할 수 있다.

SignalR의 단점(?)

- 접속한 클라이언트의 IP 주소를 알 수 없음.
 - IIS 호스팅인 경우 ASP.NET의 기능으로 알 수 있음.
- 서버에서 접속한 클라이언트의 접속을 짜를 수 있는 기능이 없음

Demo

참고

- SignalR / SignalR
<https://github.com/SignalR/SignalR>
wiki <https://github.com/SignalR/SignalR/wiki>
<http://www.asp.net/signalr>
- MSDN
[http://msdn.microsoft.com/ko-kr/library/jj943739\(v=vs.111\).aspx](http://msdn.microsoft.com/ko-kr/library/jj943739(v=vs.111).aspx)
- ASP.NET SignalR 시작하기 (C#)
<http://www.taeyo.pe.kr/Columns/View.aspx?SEQ=458&PSEQ=35>
- Sample
<https://github.com/SignalR/Samples>
- SignalR - Simple Chat Application in C#
<http://www.codeproject.com/Articles/524066/SignalR-Simple-Chat-Application-in-Csharp>
- SignalR을 이용한 퀴즈 게임 앱
<https://github.com/jsakamoto/quiz-webapp>