

# SignalR

Welcome to the Real-time world of web



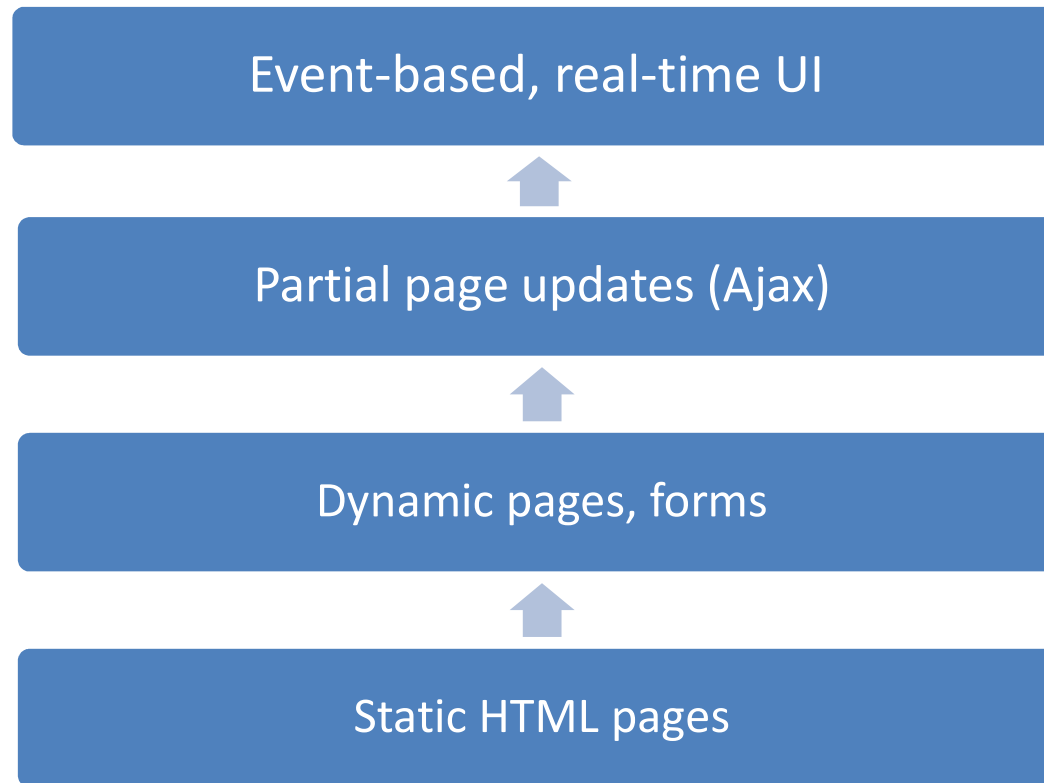
 /melshafey1

 @elshafe3y

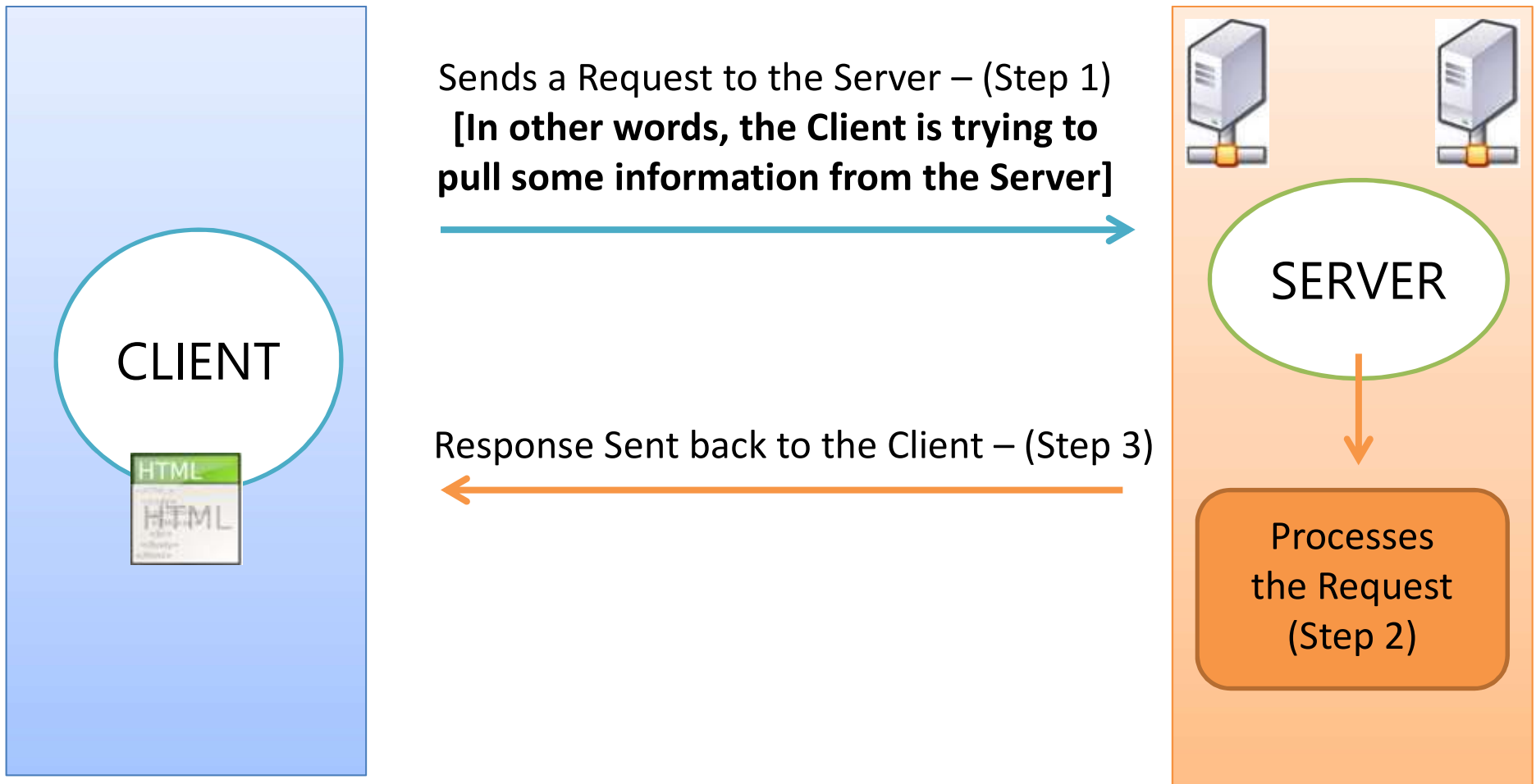
# Topics Focused On!!!

- Web Evolution
- What does “Real Time” mean?
- Poll & Push
- WebSocket
- SignalR – “Modern WEB”
- How do I get SignalR?
- Demo

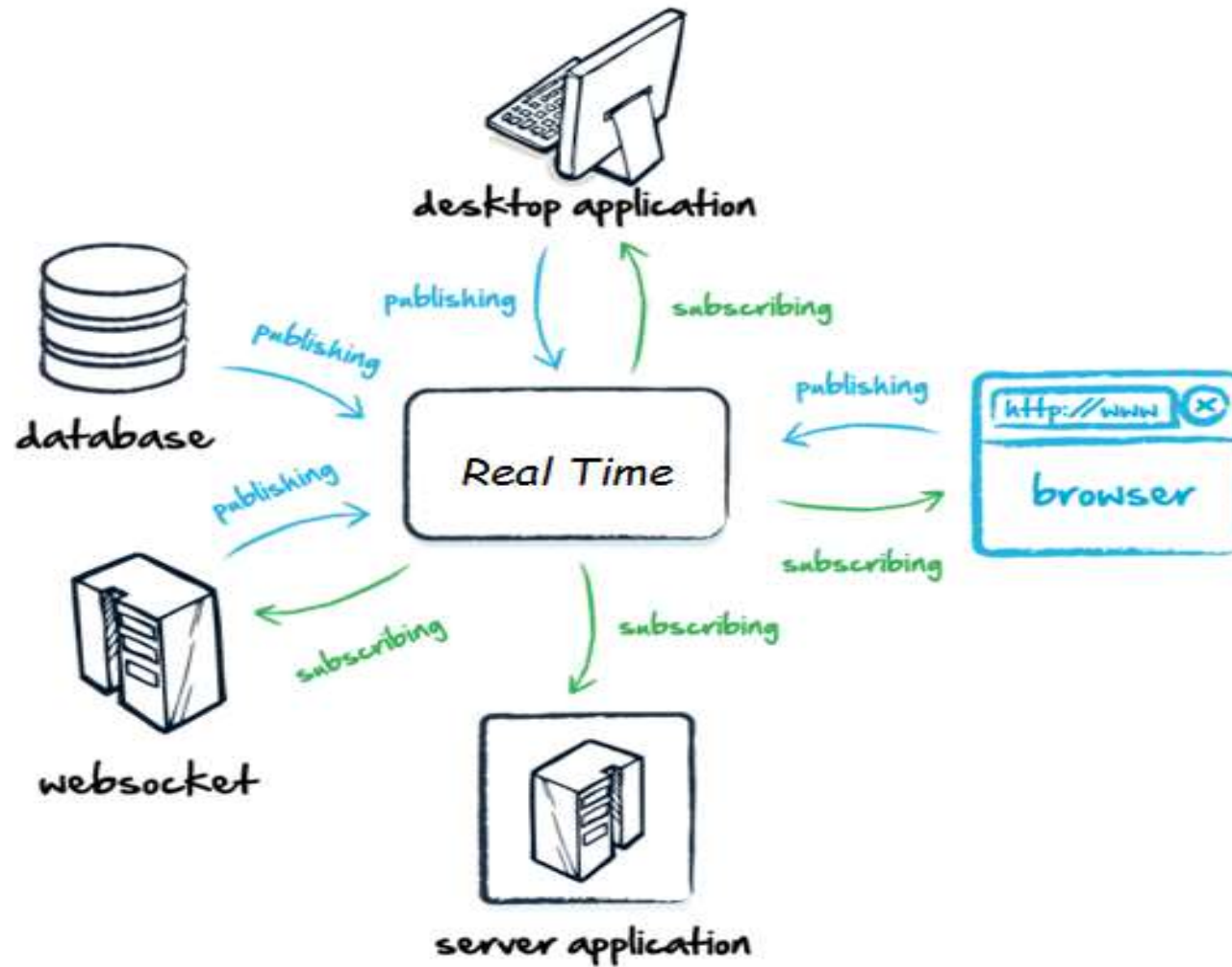
# Web Evolution



# Traditional Web Approach



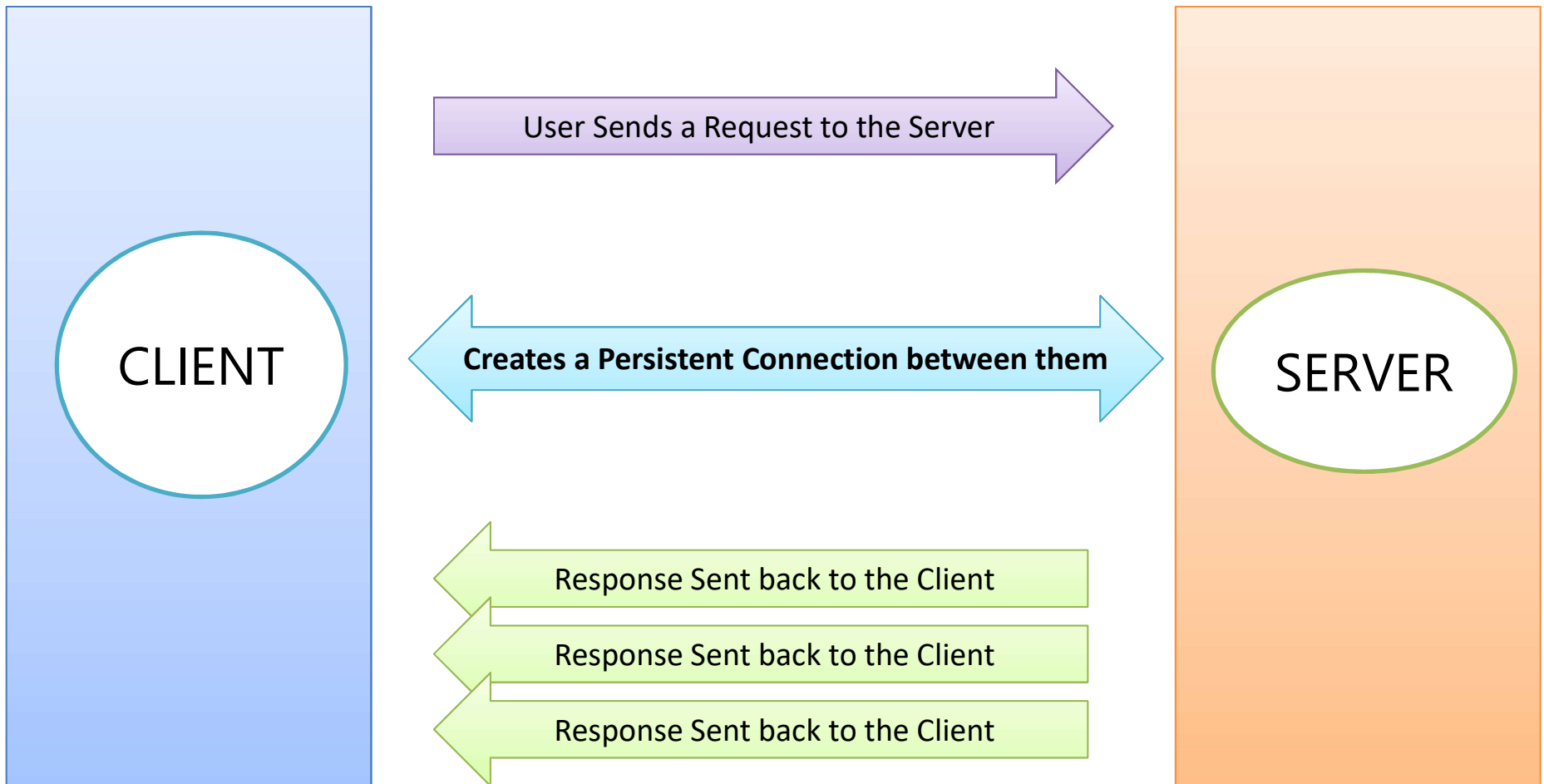
# Real Time Web Applications



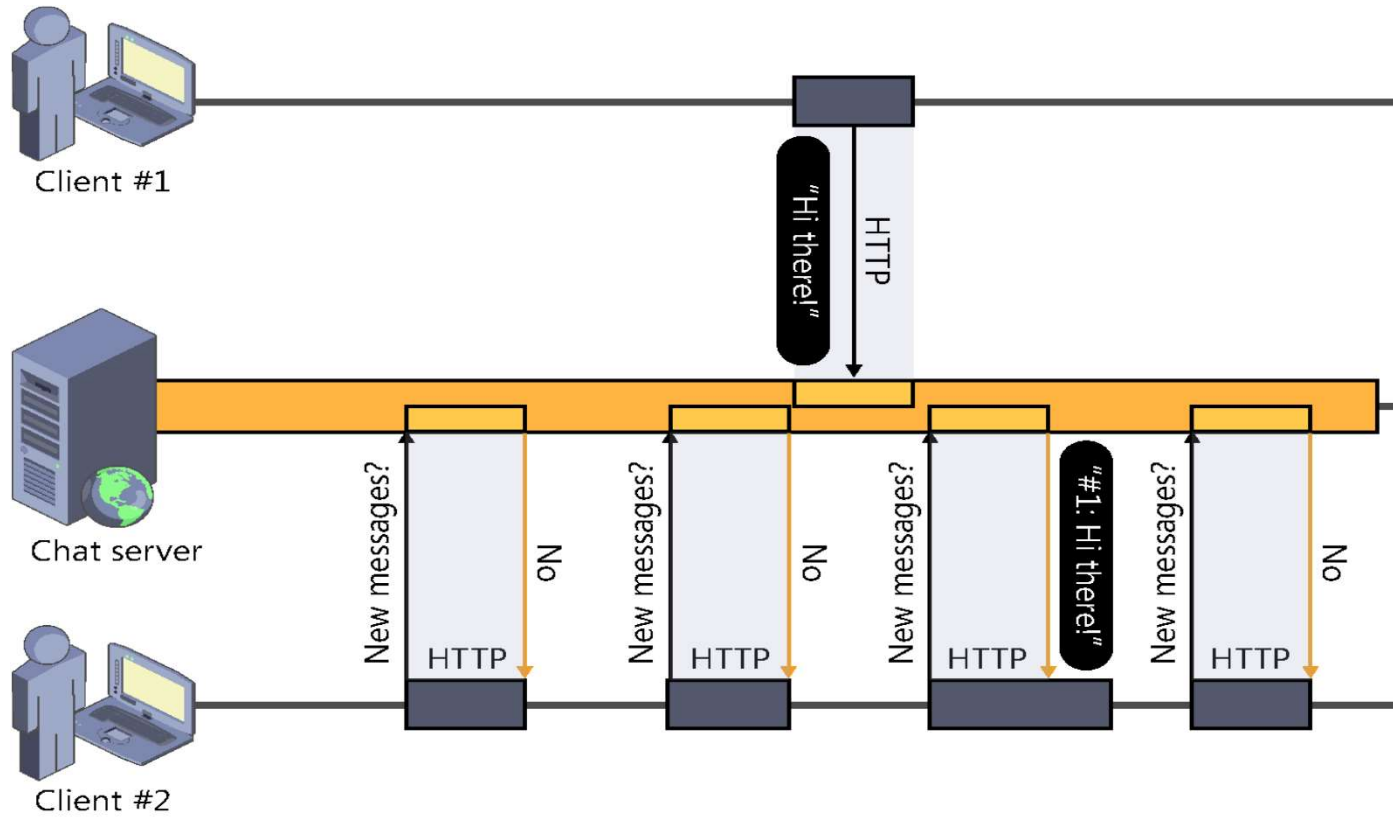
# What is Real Time Web Application?

- In simple terms, "Real Time" means an immediate response being sent by the Server to the Client.
- Real Time is all about "Pushing" instead of "Pulling"
- Push Technology is completely different from Pull Technology. Its about getting told what's new, instead of asking for what's new!!!
- Facebook, Twitter, Yahoo Cricket Live, Stock Ticker

# Real Time Web Approach

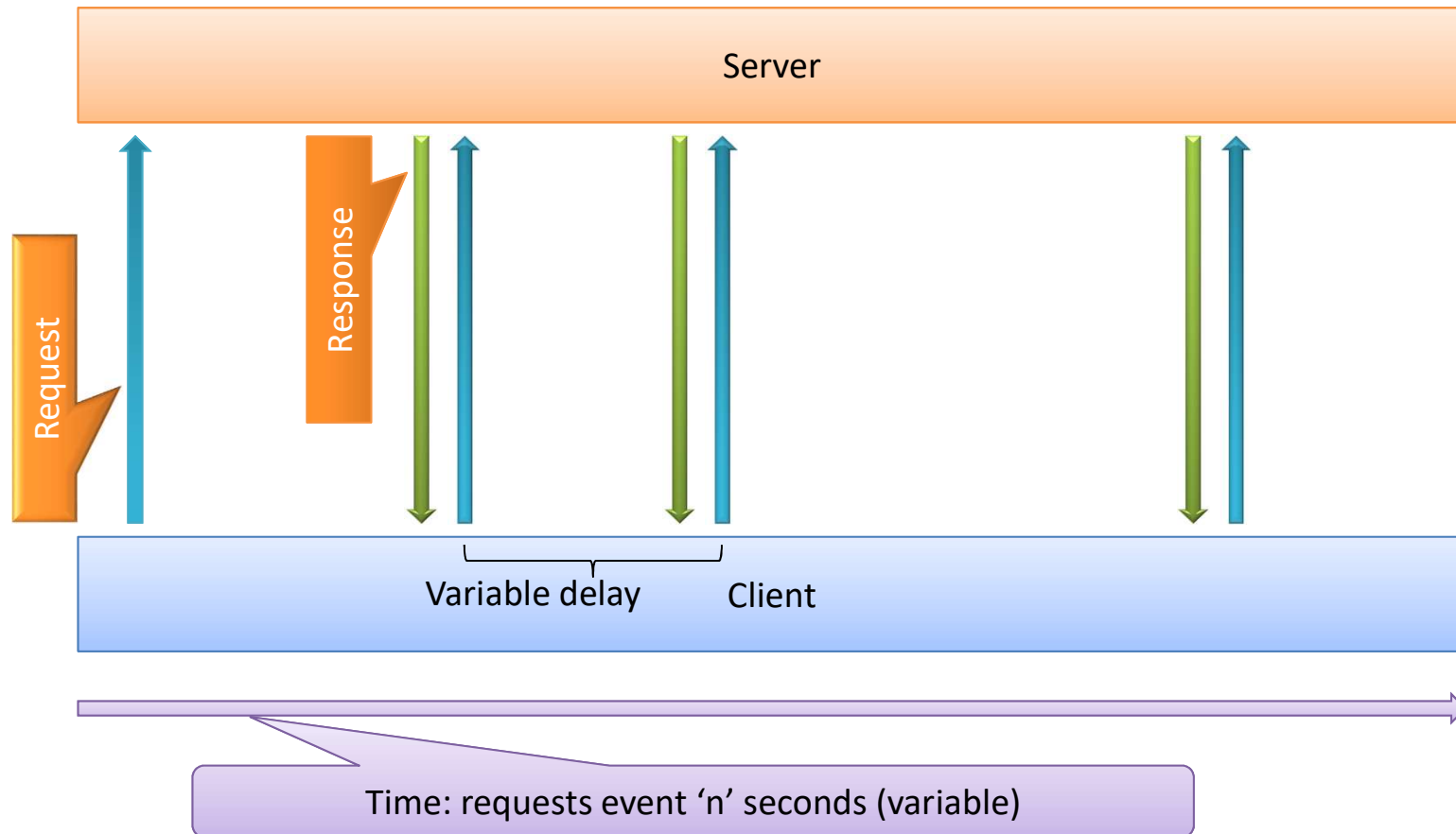


# Polling





# Long Polling

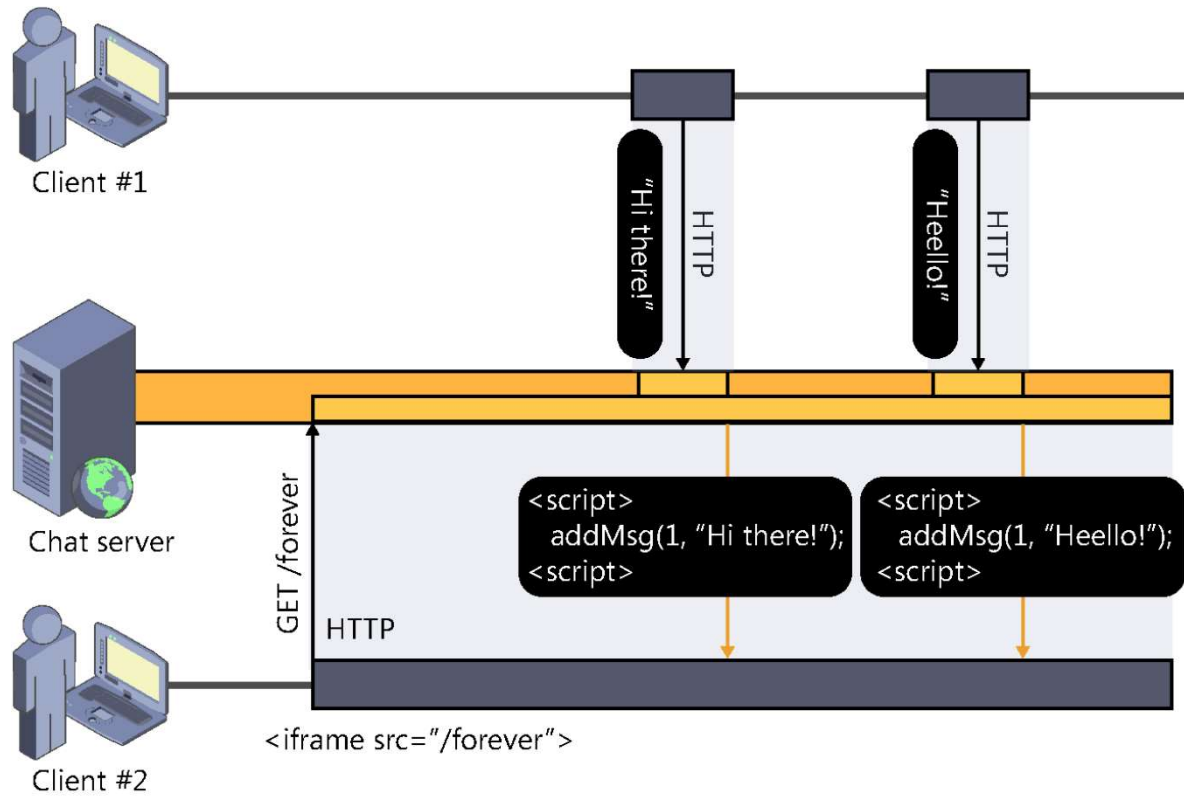


# Forever Frames

- Data is sent out in chunks.
- Chunked Encoding : is the feature in the [HTTP 1.1 specification](#) allowing a server to start sending a response before knowing its total length
- A hidden iframe element is opened in the browser after page load, establishing a long-lived connection inside the hidden iframe..

Pros	Cons
Supported on IE Browser.	<ul style="list-style-type: none"><li>▪ Iframes are loaded again and again with chunks of data.</li><li>▪ All script tags remain on the page.</li><li>▪ one-way realtime connection from server to client</li></ul>

# Forever Frames

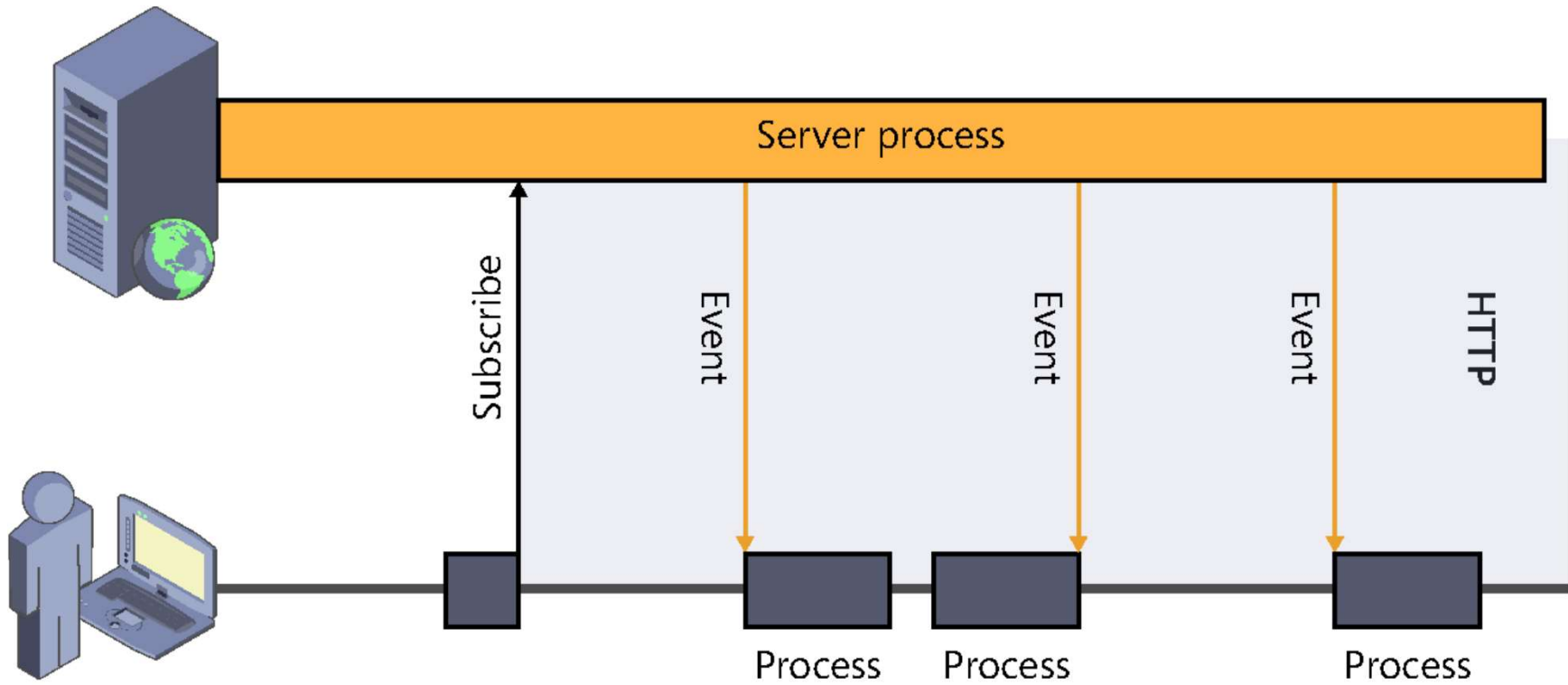


# Server Sent Events

- Requires a single connection between Client-Server.
- Uses Javascript API – “EventSource” through which Client can request a particular URL to receive data stream.
- Used to send Message Notifications or Continuous Data Streams.

Pros	Cons
No need to reconnect	Works in server-to-client direction only
	Not supported in IE

# Server Sent Events(SSE)



# Server Sent Events(SSE)

```
<script>  
    var source = new EventSource('/elshafei/getevents');  
  
    source.onmessage = function (event) {  
        alert(event.data);  
    };  
</script>
```

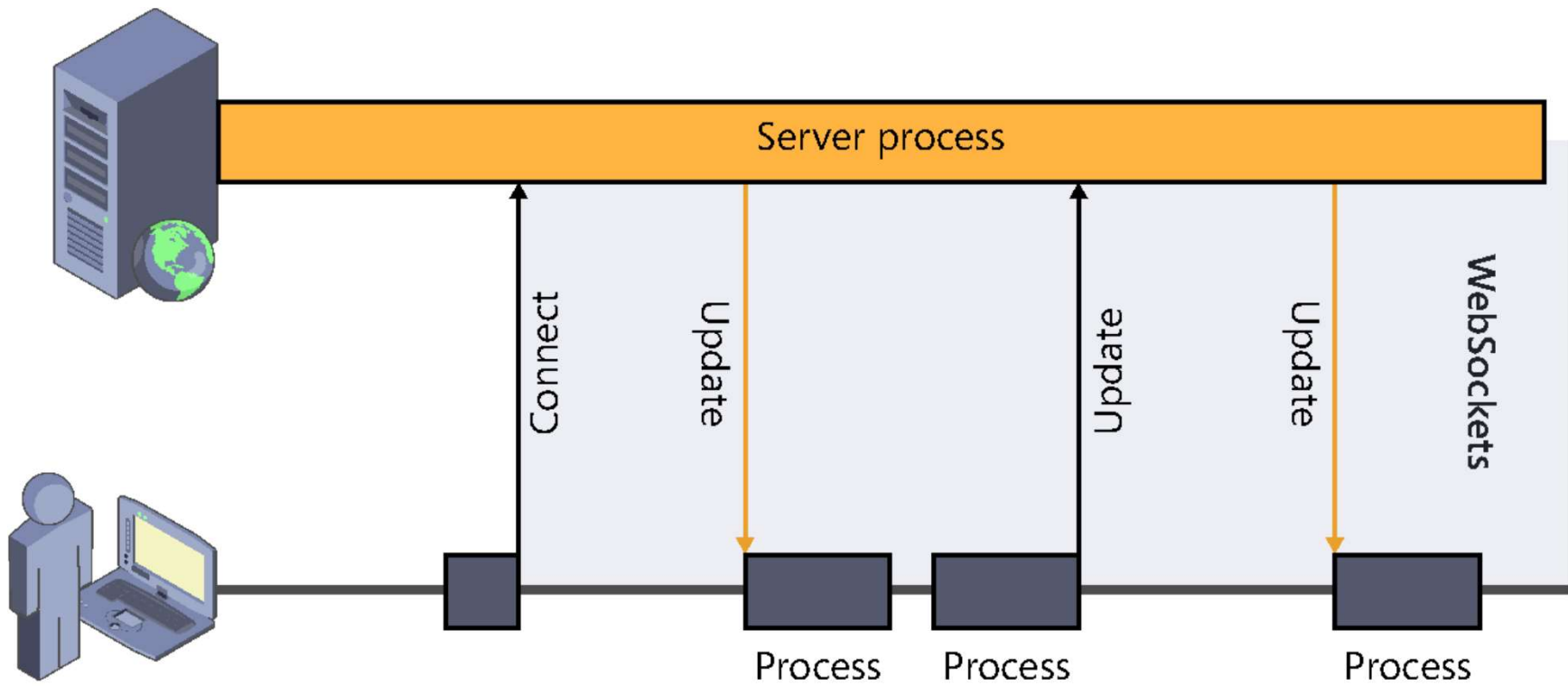
Events	Description
onopen	When a connection to the server is opened
onmessage	When a message is received
onerror	When an error occurs

# WebSocket

- A new transport technique which came up with HTML5.
- a full-duplex single socket connection over which messages can be sent between client and server
- It internally works on top of TCP protocol.

Pros	Cons
Full-duplex persistent connection (both ways)	Supported only on latest browsers – (IE 10) Windows 8, Windows Server 2012 or later
Fastest solution	Works only with IIS-8.0 .Net Framework 4.5+

# WebSocket





# WebSocket

```
<script>
  var ws = new WebSocket("ws://localhost:9998/index");
  ws.onopen = function () {
    // Web Socket is connected, send data using send()
    ws.send("Message to send");
    alert("Message is sent...");
  };
  ws.onmessage = function (evt) {
    var received_msg = evt.data;
    alert("Message is received...");
  };
  ws.onclose = function () {
    // WebSocket is closed.
    alert("Connection is closed...");
  };
</script>
```

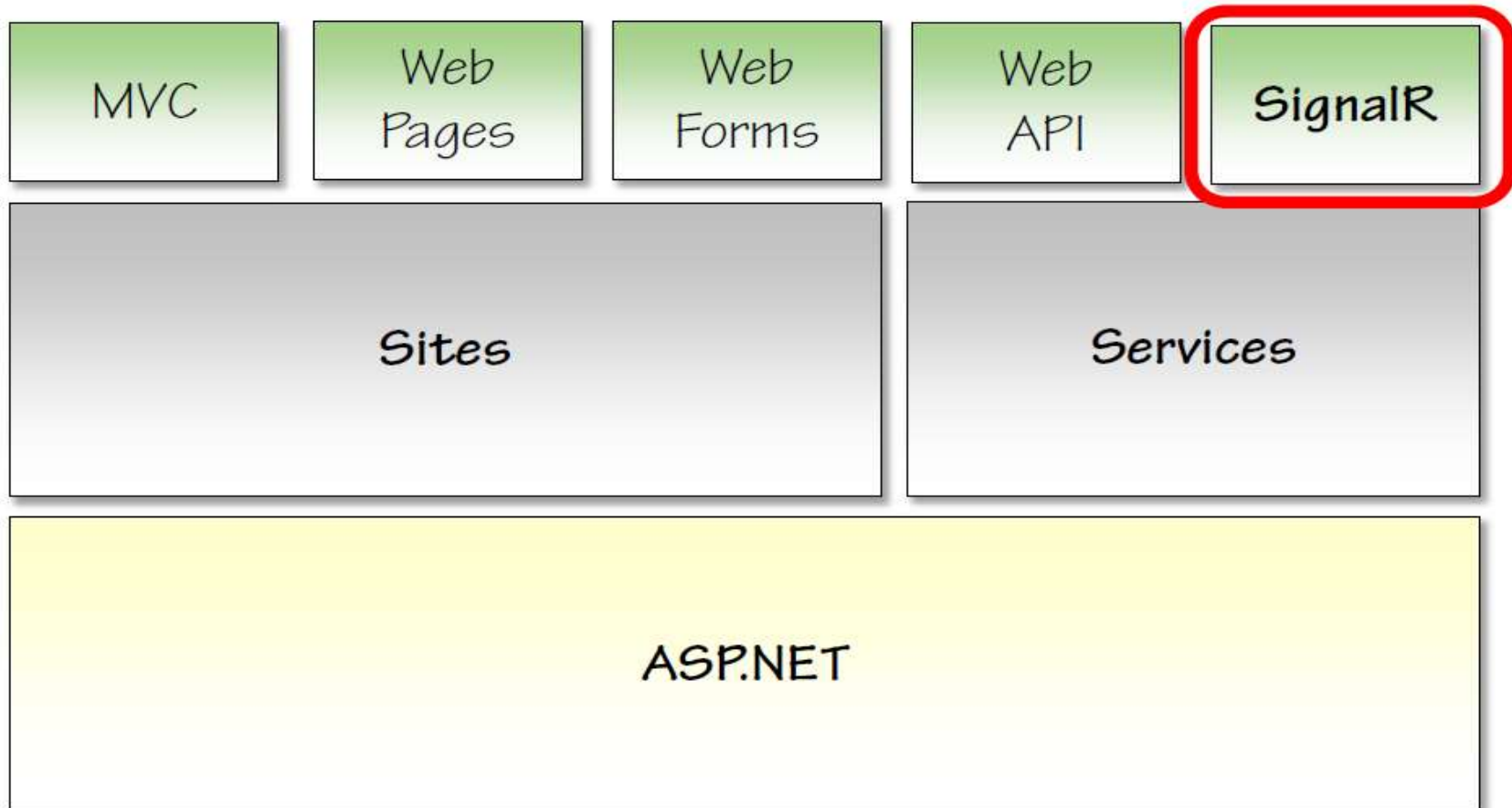
# SignalR – Modern Web

- SignalR is an asynchronous library. Used to develop Real Time Web Application.
- Concept initiated by “David Fowler” and “Damien Edwards”
- a server-side framework to write push services
- a set of client libraries to make push service communication easy to use on any platform
- optimized for asynchronous processing
- Open Source available on Github!!!

# SignalR

- SignalR handles connection management automatically, and lets you broadcast messages to all connected clients simultaneously, like a chat room. You can also send messages to specific clients. The connection between the client and server is persistent, unlike a classic HTTP connection, which is re-established for each communication.
- SignalR supports "server push" functionality, in which server code can call out to client code in the browser using Remote Procedure Calls (RPC), rather than the request-response model common on the web today.

# SignalR



# SignalR Connections

Client side

JS, .NET/WinRT, WP, Silverlight, iOS/Android



**Hubs**



**Persistent Connection**

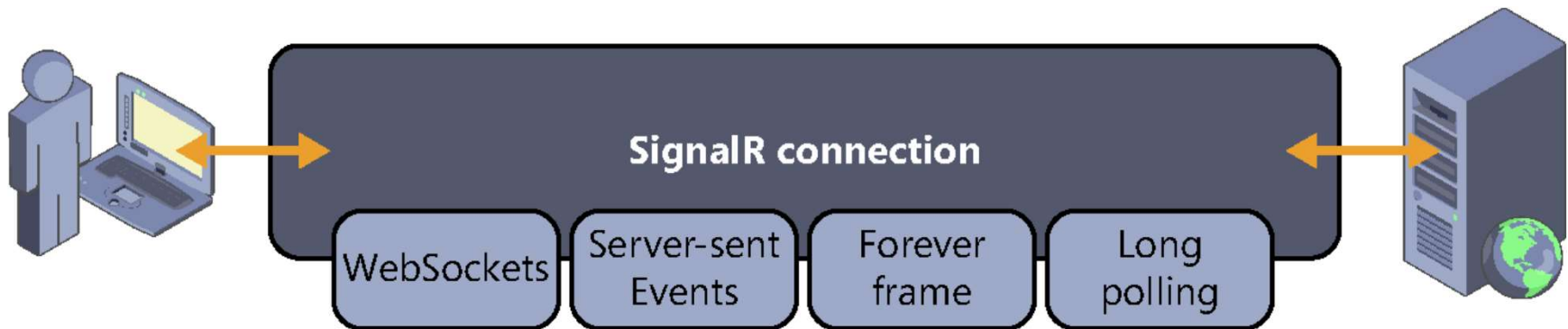
# Persistent Connection

- Provides a raw / low-level control to manage connection.
- Contain events like “OnConnection”, “OnDisconnection”, “OnReconnection”
- We can write our own logic in these events.

# Hubs

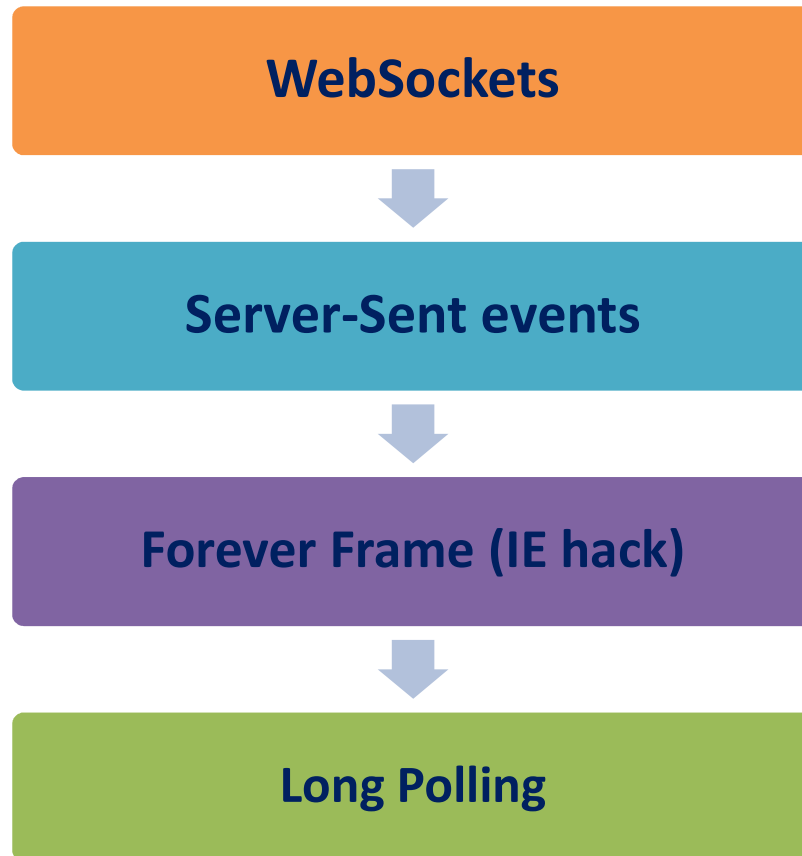
- Provides a High-level API.
- Client calling Server.
- Server calling Clients. (All, Groups, One).
- Broadcasting messages to all connected clients.
- Works in a similar way like a “Controller”

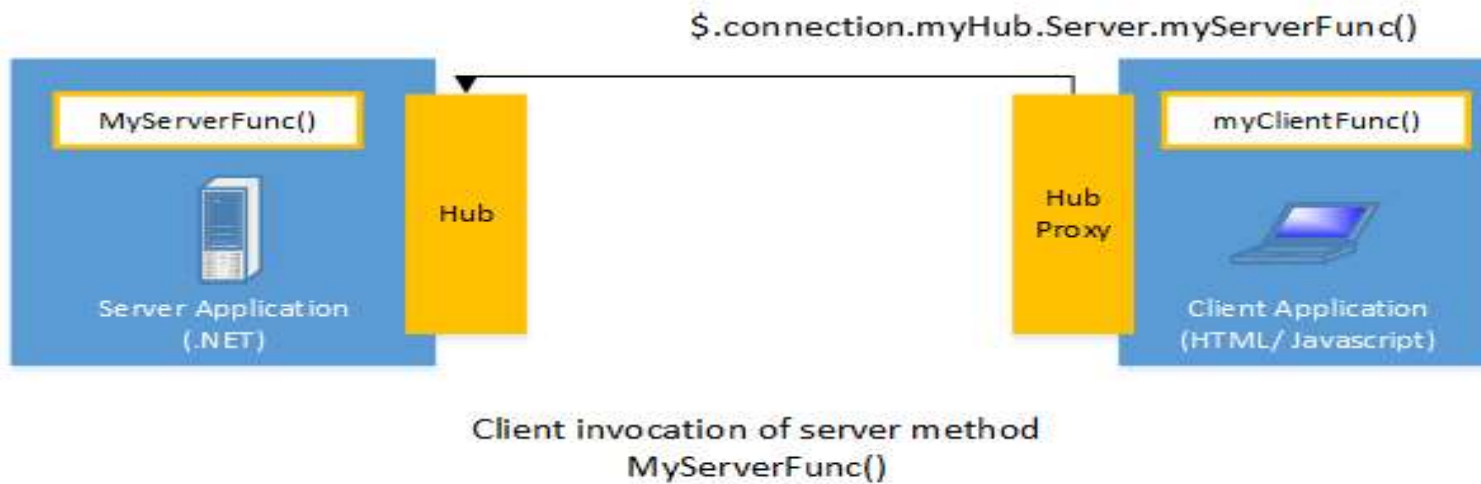
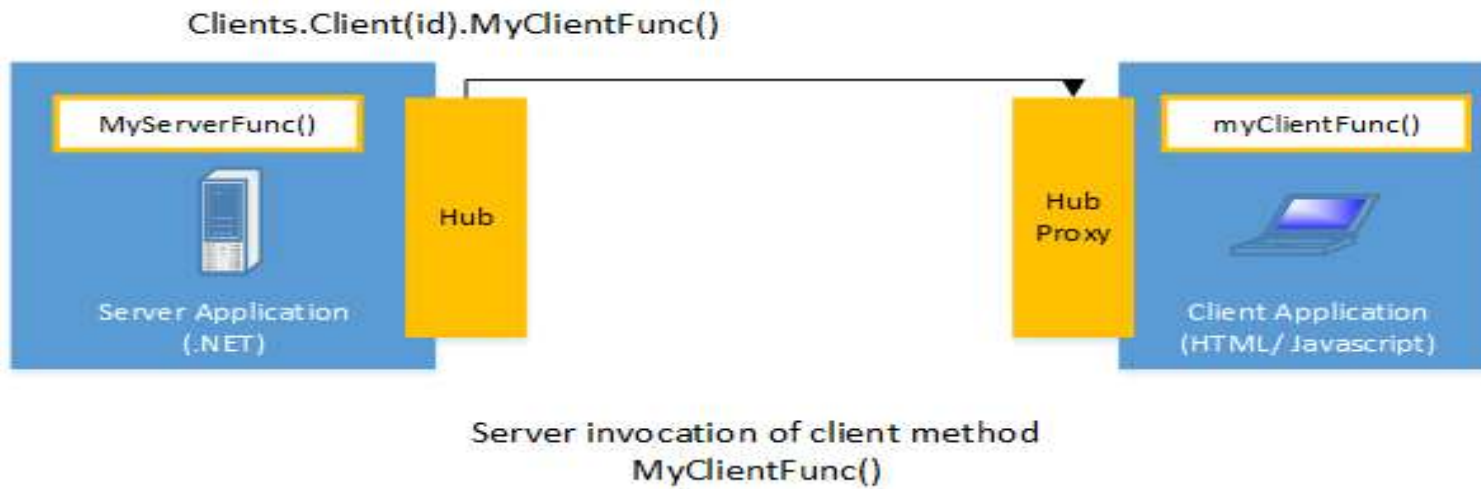
# SignalR Connections



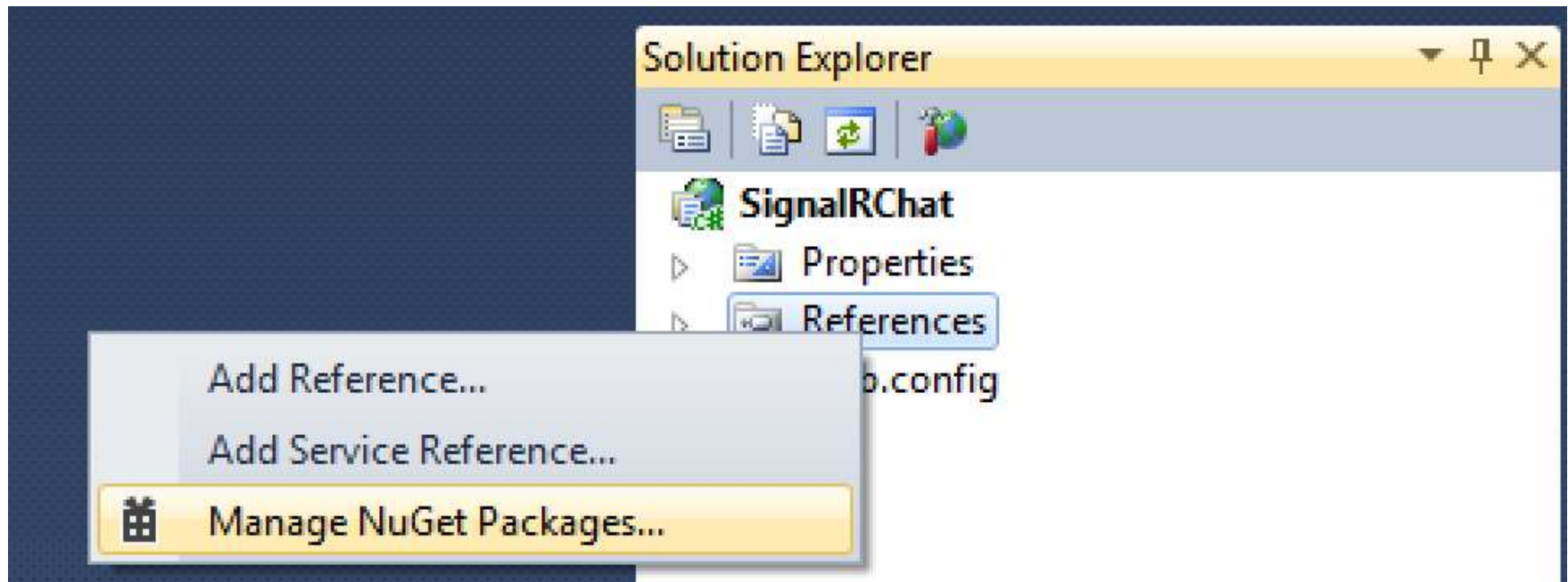


# Transport Priority





# How do I get SignalR?



SignalRChat - Manage NuGet Packages

Include Prerelease    Sort by: Relevance    SignalR

**Installed packages**

**Online**

NuGet official package source  
Search Results

**Updates**

Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.

**Microsoft ASP.NET SignalR JS**  
Javascript library for SignalR  
*Prerelease*

**Microsoft ASP.NET SignalR Core**  
Async signaling library for .NET to help build real-time, multi-user interactive web applications.  
*Prerelease*

**Microsoft ASP.NET SignalR**  
A client and server side library for .NET that provides messaging and an abstraction over...  
*Prerelease*    **Install**

**Microsoft ASP.NET SignalR Hosting for ASP.NET**  
Asp.Net host for SignalR  
*Prerelease*

**Microsoft ASP.NET SignalR Common Hosting Libraries**  
Assembly containing common components for implementing SignalR hosts.  
*Prerelease*

**SignalR.EventStream**  
Monitor events on your website live.

**jquery.captureDocumentWrite**  
Allows getting the content written by script files which use the document.write() method.

1 2 3 4 ▶

**SignalR**

**Created by:** Microsoft  
**Id:** [Microsoft.AspNet.SignalR](#)  
**Version:** 1.0.0-rc1 (Prerelease)  
**Last Published:** 12/13/2012  
**Downloads:** 4722  
[View License Terms](#)  
[Project Information](#)  
[Report Abuse](#)

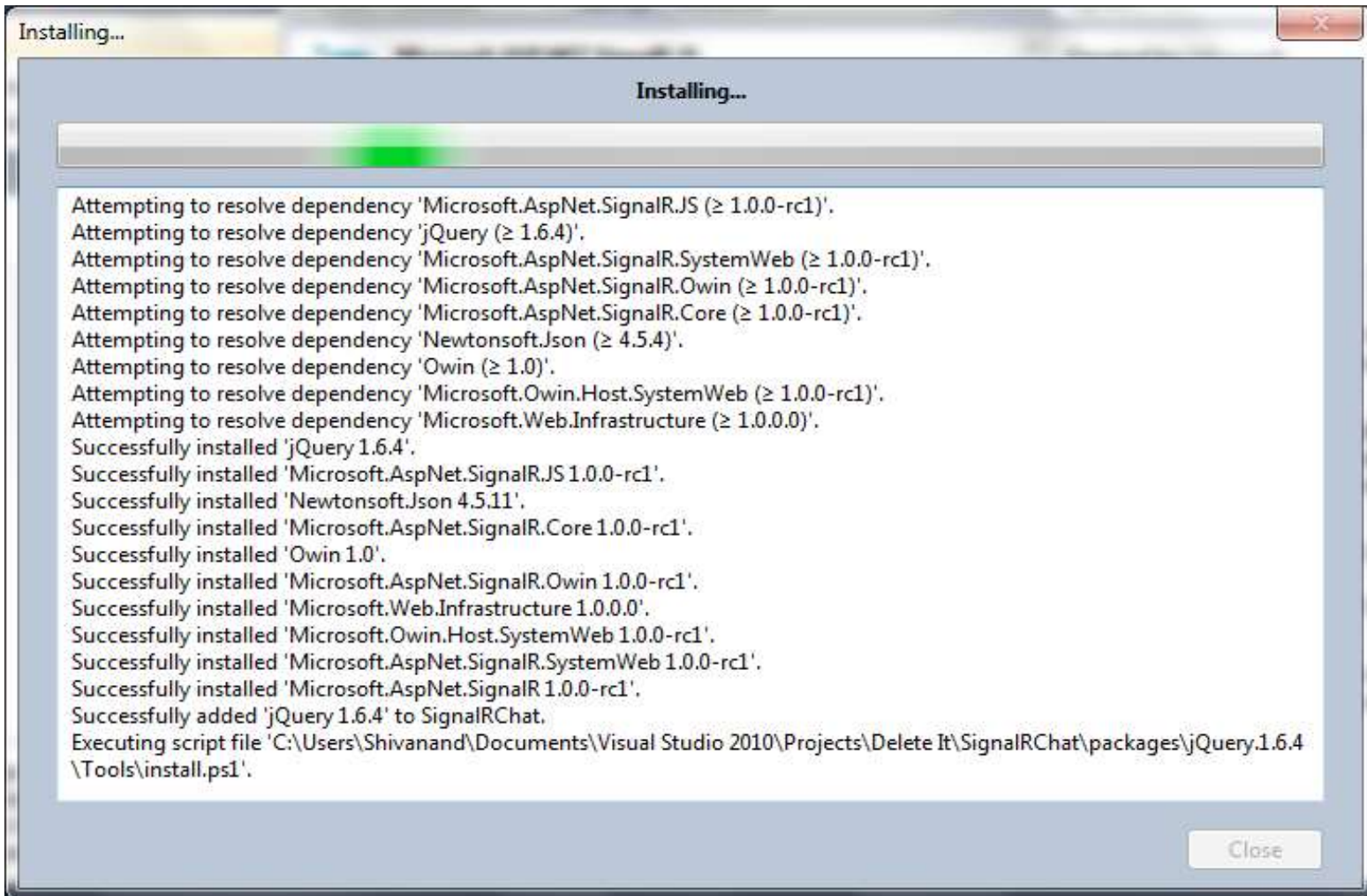
**Description:**  
A client and server side library for .NET that provides messaging and an abstraction over a persistent connection.

**Tags:** Microsoft AspNet SignalR  
AspNetSignalR websockets real-time realtime comet HTTP streaming

**Dependencies:**  
Microsoft.AspNet.SignalR.JS (≥ 1.0.0-rc1)  
Microsoft.AspNet.SignalR.SystemWeb (≥ 1.0.0-rc1)

*Each item above may have sub-dependencies subject to additional license agreements.*

**Settings**    **Close**



SignalRChat - Manage NuGet Packages

Include Prerelease    Sort by: Relevance    SignalR


**Installed packages**


**Online**


NuGet official package source  
Search Results


**Updates**


Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.


 **Microsoft ASP.NET SignalR JS**  
 Javascript library for SignalR  
Prerelease


 **Microsoft ASP.NET SignalR Core**  
 Async signaling library for .NET to help build real-time, multi-user interactive web applications.  
Prerelease

 **Microsoft ASP.NET SignalR**  
 A client and server side library for .NET that provides messaging and an abstraction over a persistent connecti...  
Prerelease

 **Microsoft ASP.NET SignalR Hosting for ASP.NET**  
 Asp.Net host for SignalR  
Prerelease

 **Microsoft ASP.NET SignalR Common Hosting Libraries**  
 Assembly containing common components for implementing SignalR hosts.  
Prerelease

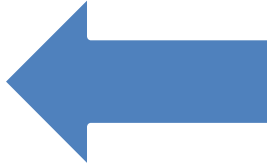
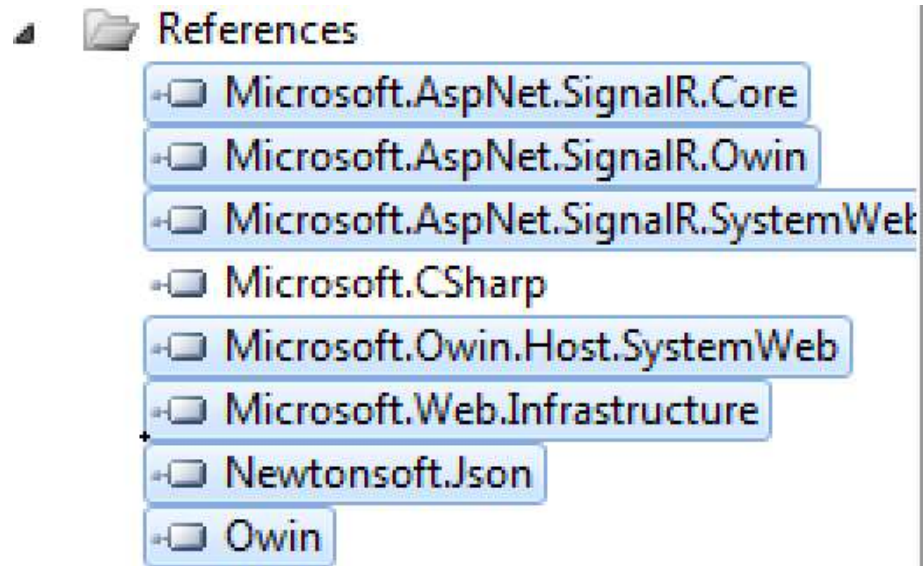
 **SignalR.EventStream**  
 Monitor events on your website live.

 **jquery.captureDocumentWrite**  
 Allows getting the content written by script files which use the document.write() method.

1 2 3 4 ▶

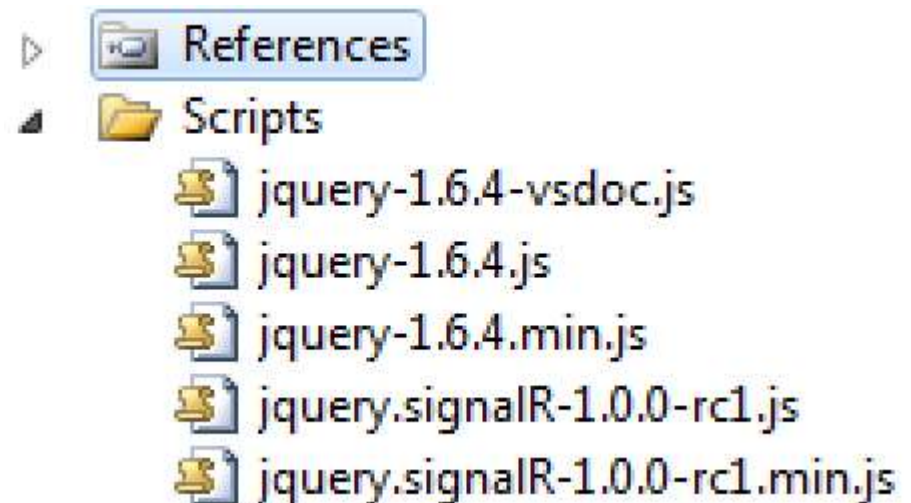
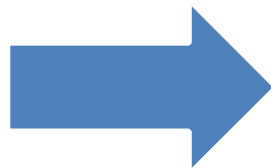
**Created by:** Microsoft  
**Id:** [Microsoft.AspNet.SignalR](#)  
**Version:** 1.0.0-rc1 (Prerelease)  
**Last Published:** 12/13/2012  
**Downloads:** 4722  
[View License Terms](#)  
[Project Information](#)  
[Report Abuse](#)  
**Description:**  
 A client and server side library for .NET that provides messaging and an abstraction over a persistent connection.  
**Tags:** Microsoft AspNet SignalR  
 AspNetSignalR websockets real-time realtime comet HTTP streaming  
**Dependencies:**  
 Microsoft.AspNet.SignalR.JS (≥ 1.0.0-rc1)  
 Microsoft.AspNet.SignalR.SystemWeb (≥ 1.0.0-rc1)  
*Each item above may have sub-dependencies subject to additional license agreements.*

Settings    Close



Some References!!!

Some JQuery!!!



# SignalR Mapping

```
using System;
using System.Threading.Tasks;
using Microsoft.Owin;
using Owin;

[assembly: OwinStartup(typeof(chat.Startup))]

namespace chat
{
    1 reference
    public class Startup
    {
        0 references
        public void Configuration(IAppBuilder app)
        {
            //signalR mapping...
            app.MapSignalR();
        }
    }
}
```



# Server Calling Client

```
public class ChatHub : Hub
{
    public void Send(string message, string author)
    {
        Clients.All.sendMessage(message, author);           // everyone
        Clients.Others.sendMessage(message, author);        // everyone except the caller
        Clients.AllExcept(connectionId).sendMessage(message, author); // everyone except
        Clients.Caller.sendMessage(message, author);        // caller
        Clients.Group("foo").sendMessage(message, author);  // everyone in group
        Clients.OthersInGroup("foo").sendMessage(message, author); // everyone else in group
        Clients.Group("foo", connectionId).sendMessage(message, author); // everyone in group except
        Clients.Client(connectionId).sendMessage(message, author); // specific client
    }
}
```

# Groups

- Can add connections to groups and send messages to particular groups
- Groups are not persisted on the server
- No automatic group count
- Need to manage groups by yourself
  - `Groups.Add(ConnectionId, groupName)`
  - `Groups.Remove(ConnectionId, groupName)`

# Hub consumers

- Consumers can be classic client applications or other services/hubs
- SignalR provides a variety of client libraries
- Microsoft SignalR team
  - .NET 4.0+
  - WinRT
  - Windows Phone 8
  - Silverlight 5
  - jQuery
  - C++
- Community
  - iOS native
  - iOS via Mono
  - Android via Mono

# JavaScript client

- `$.connection.{hubname}`
  - access a client side hub from the generated proxy
- `$.connection.hub.start()`
  - starts the connection for all hubs
- `myHub.{method} = callback`
  - declares a function the server can invoke.
  - `method` - name of the client side method
  - `callback` - function to execute when the server invokes the method

# JavaScript client

- [hub].server.abc
  - Call methods on the server hub
- [hub].client.xyz
  - Define client-side methods to be invoked by server hub
- transport configuration
  - \$.connection.hub.start({ transport: 'longPolling' });

# Client Calling Server Function

```
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script src="~/Scripts/jquery.signalR-2.2.0.min.js"></script>
<script src="~/signalr/hubs"></script>

<script>
  var chat;
  $(function () {
    //connect to Hub Proxy
    chat = $.connection.chat;

    //start connection
    $.connection.hub.start();

    //Declare method so server can invoke it
    chatt.client.newmessage = function (message, name) {
      $("#messarea").append("<li><b>" + name + " :</b>" + message + "</li>")
    }

  })
  function send() {

    //call the method in the server
    chatt.server.sendMessage($("#txt").val(), name);
  }
</script>
```

## Specifying a transport

- `connection.start({ transport: 'longPolling' });`
- `connection.start({ transport: ['webSockets','longPolling'] });`
- The string constants for specifying transports are defined as follows:
  - `webSockets`
  - `serverSentEvents`
  - `foreverFrame`
  - `longPolling`

DEMO



# Routing is Very Important!!!

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.SessionState;
using System.Web.Routing;
using Microsoft.AspNet.SignalR;

namespace SignalRChat
{
    public class Global : System.Web.HttpApplication
    {
        protected void Application_Start(object sender, EventArgs e)
        {
            RouteTable.Routes.MapHubs();
        }
        . . . . .
    }
}
```

# .NET client

- Simple steps to get going
  1. Create HubConnection
  2. Create hub proxy via CreateHubProxy
  3. Wire up event handlers via On
  4. Start connection with Start
  5. Call methods via Invoke

# .NET client

```
IHubProxy proxy;
```

```
1 reference
```

```
private void button4_Click(object sender, EventArgs e)
```

```
{
```

```
    //connect....
```

```
    HubConnection hub = new HubConnection("http://localhost:24258/signalr/hubs");
```

```
    proxy = hub.CreateHubProxy("chat");
```

```
    proxy.On<string>("newMessage", msg => messagelist.Invoke(
```

```
        new Action( () => messagelist.Items.Add(msg))));
```

```
    hub.Start();
```

```
}
```

```
1 reference
```

```
private void button2_Click(object sender, EventArgs e)
```

```
{
```

```
    //send message...
```

```
    proxy.Invoke("sendMessage", textBox2.Text);
```

```
}
```

# Hub connections have a lifecycle

```
public override Task OnConnected()
{
    ...
}
public override Task OnDisconnected()
{
    ...
}
public override Task OnReconnected()
{
    ...
}
```

# Real-time apps use SignalR

- **ShootR**: Multiplayer space shooter game
- **JabbR**: Real time chat application
- **LoggR**: Real time web app monitoring

Thanks for Listening!!!