Artsofte

Что нового в SignalR ?

by Дима Егоров

План доклада

1. Что нового в .net 9 ? (официальная документация) и разбор с исходным кодом + issue и mr https://github.com/dotnet/aspnetcore/pulls?q=signalr
2. Что еще сделали, но не сказали об этом в анонсе ?
3. Чего не хватает в SignalR ? (на мой субъективный взгляд)

Подписывайтесь и ставьте лайки

@ArtsofteEducation

@dimoner1

Artsofte

Polymorphic type support in SignalR Hubs

```
public class MyHub : Hub
{
    public void Method(JsonPerson person)
    {
        if (person is JsonPersonExtended)
        {
        }
        else if (person is JsonPersonExtended2)
        {
        }
        else
        {
        }
    }
}

[JsonPolymorphic]
[JsonDerivedType(typeof(JsonPersonExtended), nameof(JsonPersonExtended))]
[JsonDerivedType(typeof(JsonPersonExtended2), nameof(JsonPersonExtended2))]
private class JsonPerson
{
    public string Name { get; set; }
    public Person Child { get; set; }
    public Person Parent { get; set; }
}

private class JsonPersonExtended : JsonPerson
{
    public int Age { get; set; }
}

private class JsonPersonExtended2 : JsonPerson
{
    public string Location { get; set; }
}
```

Artsofte

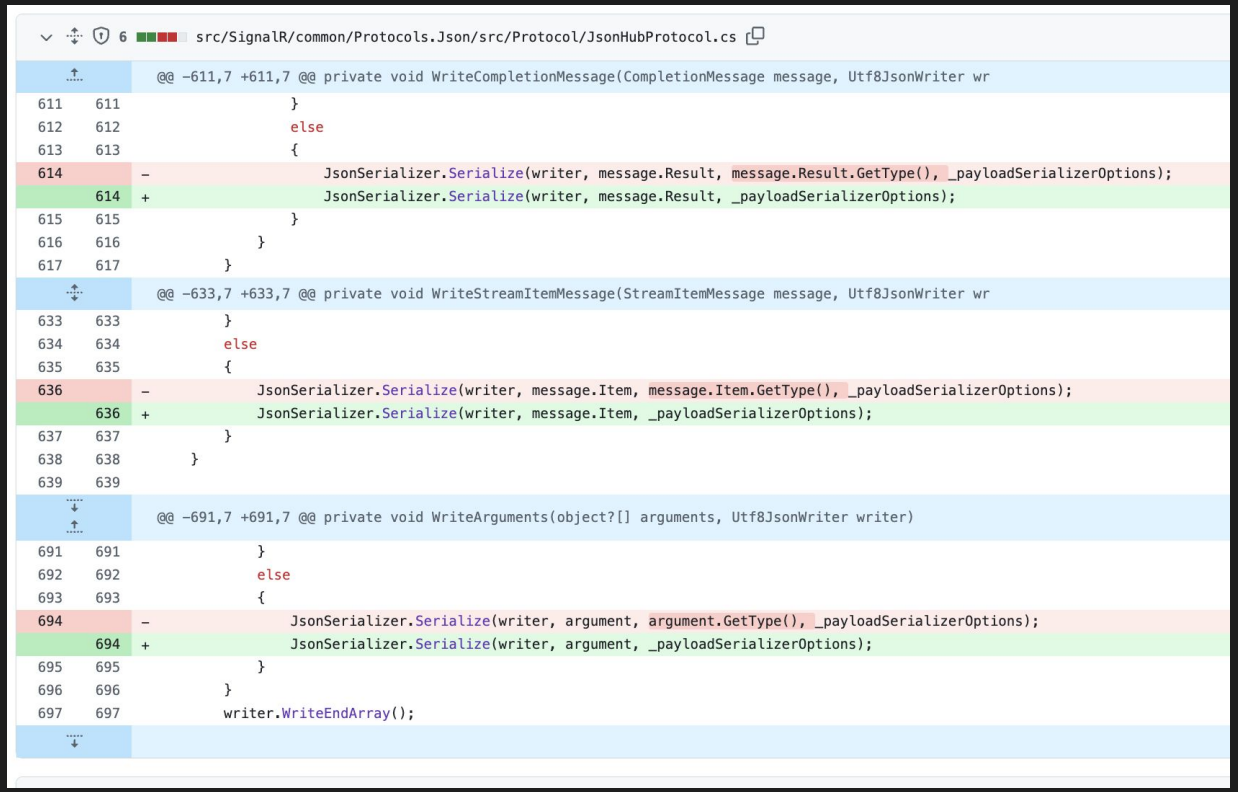
Polymorphic type support in SignalR Hubs

```
var weather = new WeatherForecastWithCity
{
    City = "Ekb",
    TemperatureCelsius = 15,
};

// {"$type":"withCity","City":"Ekb","TemperatureCelsius":15}
var json1 = JsonSerializer.Serialize<WeatherForecastBase>(weather);
// {"City":"Ekb","TemperatureCelsius":15}
var json2 = WeatherForecastBase.Execute<WeatherForecastWithCity>(weather);

[JsonDerivedType(typeof(WeatherForecastBase), typeDiscriminator: "base")]
[JsonDerivedType(typeof(WeatherForecastWithCity), typeDiscriminator:
public class WeatherForecastBase
{
    public int TemperatureCelsius { get; set; }

    public static string Execute(WeatherForecastBase weather)
    {
        return JsonSerializer.Serialize(weather);
    }
}

public class WeatherForecastWithCity : WeatherForecastBase
{
    public string? City { get; set; }
}
```

Artsofte

Polymorphic type support in SignalR Hubs

Github PR



src/SignalR/common/Protocols.Json/src/Protocol/JsonHubProtocol.cs

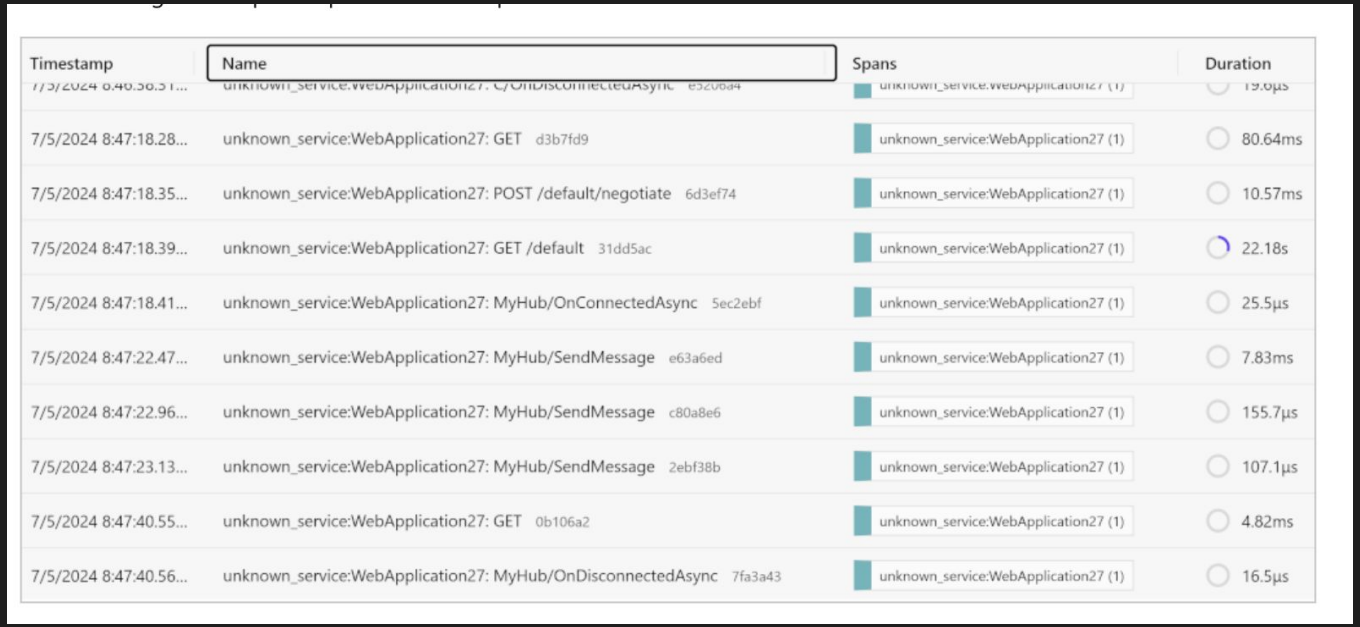
```
@@ -611,7 +611,7 @@ private void WriteCompletionMessage(CompletionMessage message, Utf8JsonWriter wr
611 611                 }
612 612                 else
613 613                 {
614     -                   JsonSerializer.Serialize(writer, message.Result, message.Result.GetType(), _payloadSerializerOptions);
    614 +                   JsonSerializer.Serialize(writer, message.Result, _payloadSerializerOptions);
615 615                 }
616 616             }
617 617         }
@@ -633,7 +633,7 @@ private void WriteStreamItemMessage(StreamItemMessage message, Utf8JsonWriter wr
633 633         }
634 634         else
635 635         {
636     -           JsonSerializer.Serialize(writer, message.Item, message.Item.GetType(), _payloadSerializerOptions);
    636 +           JsonSerializer.Serialize(writer, message.Item, _payloadSerializerOptions);
637 637         }
638 638     }
639 639
@@ -691,7 +691,7 @@ private void WriteArguments(object?[] arguments, Utf8JsonWriter writer)
691 691             }
692 692             else
693 693             {
694     -               JsonSerializer.Serialize(writer, argument, argument.GetType(), _payloadSerializerOptions);
    694 +               JsonSerializer.Serialize(writer, argument, _payloadSerializerOptions);
695 695             }
696 696         }
697 697         writer.WriteEndArray();
```

https://github.com/dotnet/aspnetcore/pull/53035

Artsofte

Improved Activities for SignalR

Обращение на изменение



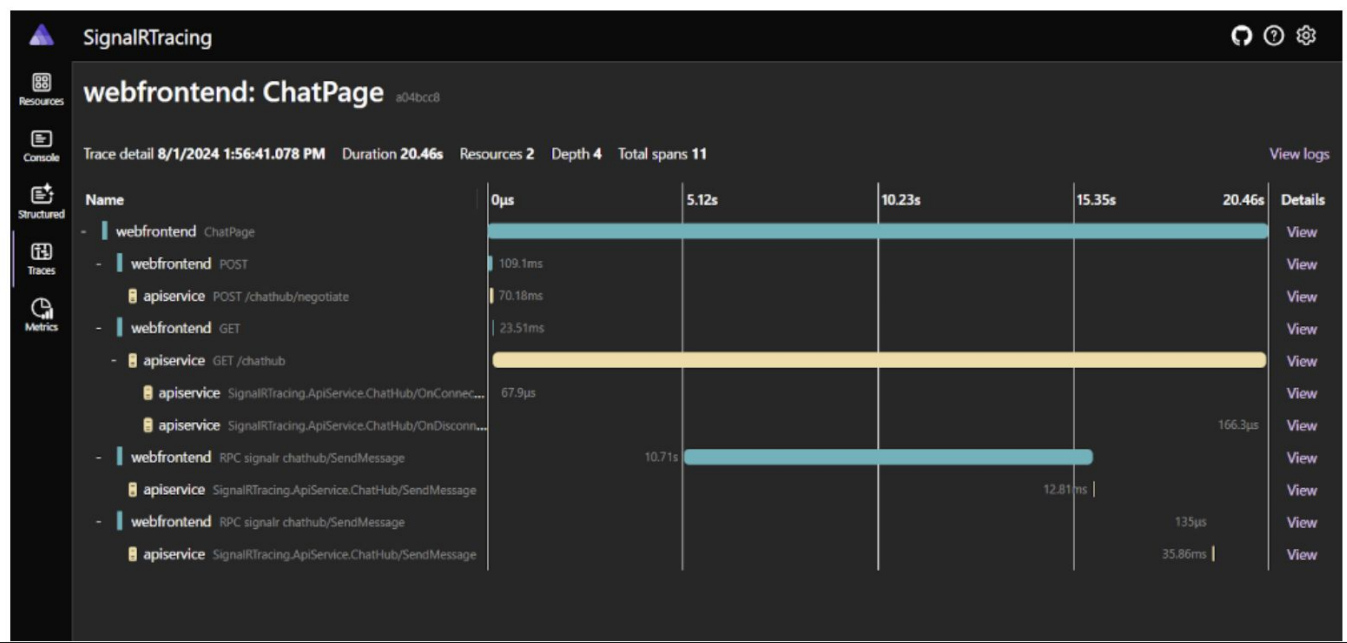
Timestamp	Name	Spans	Duration
7/5/2024 8:47:18.28...	unknown_service:WebApplication27: GET d3b7fd9	unknown_service:WebApplication27 (1)	80.64ms
7/5/2024 8:47:18.35...	unknown_service:WebApplication27: POST /default/negotiate 6d3ef74	unknown_service:WebApplication27 (1)	10.57ms
7/5/2024 8:47:18.39...	unknown_service:WebApplication27: GET /default 31dd5ac	unknown_service:WebApplication27 (1)	22.18s
7/5/2024 8:47:18.41...	unknown_service:WebApplication27: MyHub/OnConnectedAsync 5ec2ebf	unknown_service:WebApplication27 (1)	25.5µs
7/5/2024 8:47:22.47...	unknown_service:WebApplication27: MyHub/SendMessage e63a6ed	unknown_service:WebApplication27 (1)	7.83ms
7/5/2024 8:47:22.96...	unknown_service:WebApplication27: MyHub/SendMessage c80a8e6	unknown_service:WebApplication27 (1)	155.7µs
7/5/2024 8:47:23.13...	unknown_service:WebApplication27: MyHub/SendMessage 2ebf38b	unknown_service:WebApplication27 (1)	107.1µs
7/5/2024 8:47:40.55...	unknown_service:WebApplication27: GET 0b106a2	unknown_service:WebApplication27 (1)	4.82ms
7/5/2024 8:47:40.56...	unknown_service:WebApplication27: MyHub/OnDisconnectedAsync 7fa3a43	unknown_service:WebApplication27 (1)	16.5µs

https://github.com/dotnet/aspnetcore/pull/51557

Artsofte

Improved Activities for SignalR

Обращение на изменение

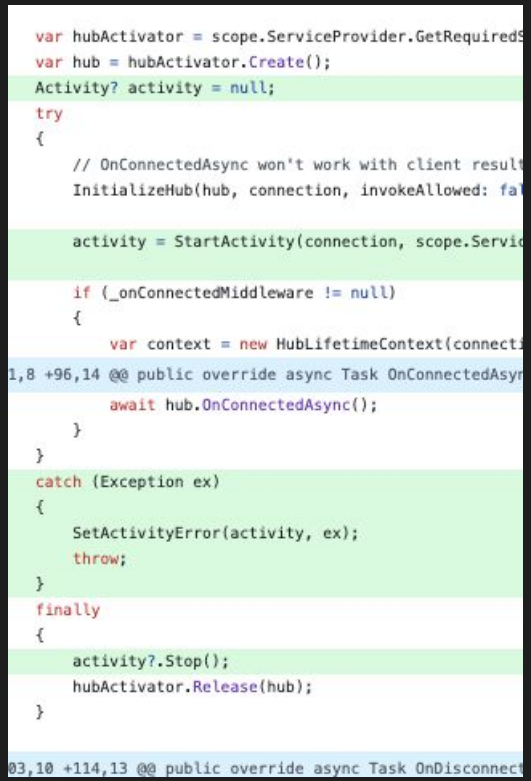


https://github.com/dotnet/aspnetcore/pull/51557

Artsofte

Improved Activities for SignalR

Начало работы

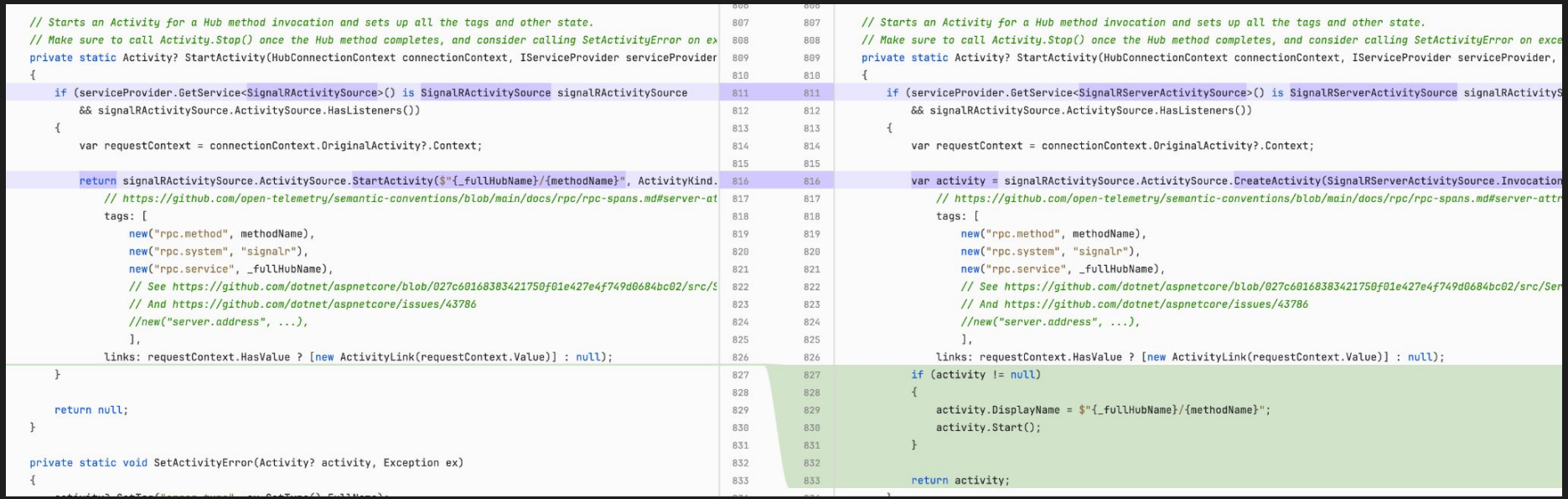


```
var hubActivator = scope.ServiceProvider.GetRequiredS
var hub = hubActivator.Create();
Activity? activity = null;
try
{
    // OnConnectedAsync won't work with client result
    InitializeHub(hub, connection, invokeAllowed: fa

    activity = StartActivity(connection, scope.Servic

    if (_onConnectedMiddleware != null)
    {
        var context = new HubLifetimeContext(connecti
1,8 +96,14 @@ public override async Task OnConnectedAsyn
        await hub.OnConnectedAsync();
    }
}
catch (Exception ex)
{
    SetActivityError(activity, ex);
    throw;
}
finally
{
    activity?.Stop();
    hubActivator.Release(hub);
}

03,10 +114,13 @@ public override async Task OnDisconnect
```

https://github.com/dotnet/aspnetcore/pull/55439

Artsofte

Посмотри git history у DefaultHubDispatcher



```
// Starts an Activity for a Hub method invocation and sets up all the tags and other state.
// Make sure to call Activity.Stop() once the Hub method completes, and consider calling SetActivityError on ex
private static Activity? StartActivity(HubConnectionContext connectionContext, IServiceProvider serviceProvider
{
    if (serviceProvider.GetService<SignalRActivitySource>() is SignalRActivitySource signalRActivitySource
        && signalRActivitySource.ActivitySource.HasListeners())
    {
        var requestContext = connectionContext.OriginalActivity?.Context;

        return signalRActivitySource.ActivitySource.StartActivity($"{_fullHubName}/{methodName}", ActivityKind.
            // https://github.com/open-telemetry/semantic-conventions/blob/main/docs/rpc/rpc-spans.md#server-at
            tags: [
                new("rpc.method", methodName),
                new("rpc.system", "signalr"),
                new("rpc.service", _fullHubName),
                // See https://github.com/dotnet/aspnetcore/blob/027c60168383421750f01e427e4f749d0684bc02/src/S
                // And https://github.com/dotnet/aspnetcore/issues/43786
                //new("server.address", ...),
                ],
            links: requestContext.HasValue ? [new ActivityLink(requestContext.Value)] : null);
    }

    return null;
}

private static void SetActivityError(Activity? activity, Exception ex)
{
```

```
// Starts an Activity for a Hub method invocation and sets up all the tags and other state.
// Make sure to call Activity.Stop() once the Hub method completes, and consider calling SetActivityError on exce
private static Activity? StartActivity(HubConnectionContext connectionContext, IServiceProvider serviceProvider,
{
    if (serviceProvider.GetService<SignalRServerActivitySource>() is SignalRServerActivitySource signalRActivityS
        && signalRActivitySource.ActivitySource.HasListeners())
    {
        var requestContext = connectionContext.OriginalActivity?.Context;

        var activity = signalRActivitySource.ActivitySource.CreateActivity(SignalRServerActivitySource.Invocation
            // https://github.com/open-telemetry/semantic-conventions/blob/main/docs/rpc/rpc-spans.md#server-attr
            tags: [
                new("rpc.method", methodName),
                new("rpc.system", "signalr"),
                new("rpc.service", _fullHubName),
                // See https://github.com/dotnet/aspnetcore/blob/027c60168383421750f01e427e4f749d0684bc02/src/Ser
                // And https://github.com/dotnet/aspnetcore/issues/43786
                //new("server.address", ...),
                ],
            links: requestContext.HasValue ? [new ActivityLink(requestContext.Value)] : null);
        if (activity != null)
        {
            activity.DisplayName = $"{_fullHubName}/{methodName}";
            activity.Start();
        }

        return activity;
    }
```

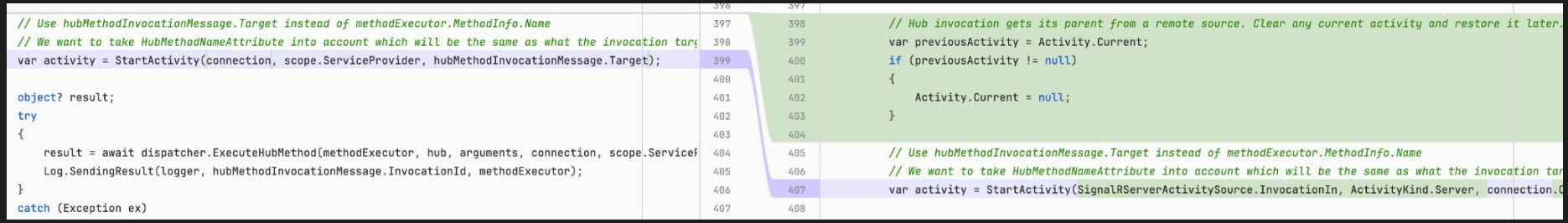
Начали фиксить
первые найденные
недоработки

https://github.com/dotnet/aspnetcore/pull/57118

Artsofte

Посмотри git history у DefaultHubDispatcher

- Вызовы хабов SignalR не сохраняли контекст трассировки.
- Трассировка обрывалась, и сложно было понять связь между запросами API и вызовами хабов.
- Телеметрия разрывалась, что затрудняло анализ производительности.



```
// Use hubMethodInvocationMessage.Target instead of methodExecutor.MethodInfo.Name
// We want to take HubMethodNameAttribute into account which will be the same as what the invocation targ
var activity = StartActivity(connection, scope.ServiceProvider, hubMethodInvocationMessage.Target);

object? result;
try
{
    result = await dispatcher.ExecuteHubMethod(methodExecutor, hub, arguments, connection, scope.ServiceP
    Log.SendingResult(logger, hubMethodInvocationMessage.InvocationId, methodExecutor);
}
catch (Exception ex)
```

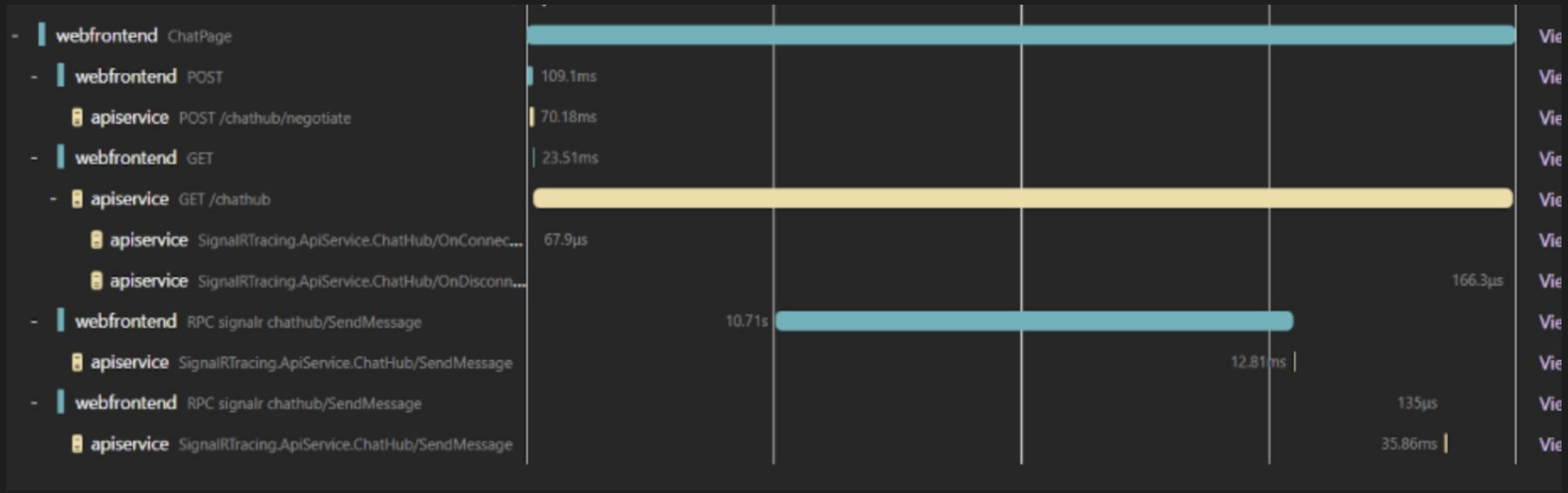
```
        // Hub invocation gets its parent from a remote source. Clear any current activity and restore it later.
        var previousActivity = Activity.Current;
        if (previousActivity != null)
        {
            Activity.Current = null;
        }

        // Use hubMethodInvocationMessage.Target instead of methodExecutor.MethodInfo.Name
        // We want to take HubMethodNameAttribute into account which will be the same as what the invocation tar
        var activity = StartActivity(SignalRServerActivitySource.InvocationIn, ActivityKind.Server, connection.C
```

Продолжаем фиксить недоработки

https://github.com/dotnet/aspnetcore/pull/57049

Artsofte

Посмотри git history у DefaultHubDispatcher



Продолжаем фиксить
недоработки

https://github.com/dotnet/aspnetcore/pull/57049

Artsofte

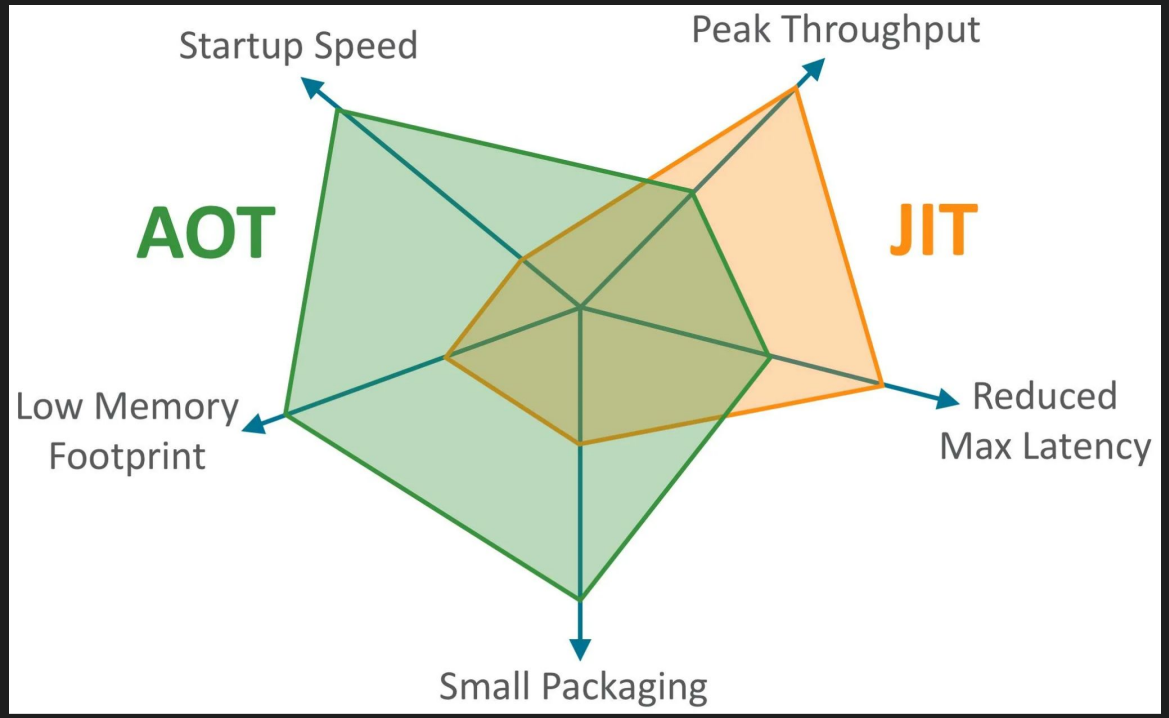
ВЫВОД “Improved Activities for SignalR“

1) Фича реально полезная и приятная
2) Вся фича заняла очень мало изменений, но не доделали до конца - можно войти в историю и помочь. (Клиенты SignalR, такие как клиент JavaScript, не поддерживают трассировку)
3) Забавно что на ходу меняется итоговая реализация, нет этапа проектирования

https://learn.microsoft.com/ru-ru/aspnet/core/release-notes/aspnetcore-9.0?view=aspnetcore-9.0#net-signalr-client-activitysource

Artsofte

SignalR supports trimming and Native AOT

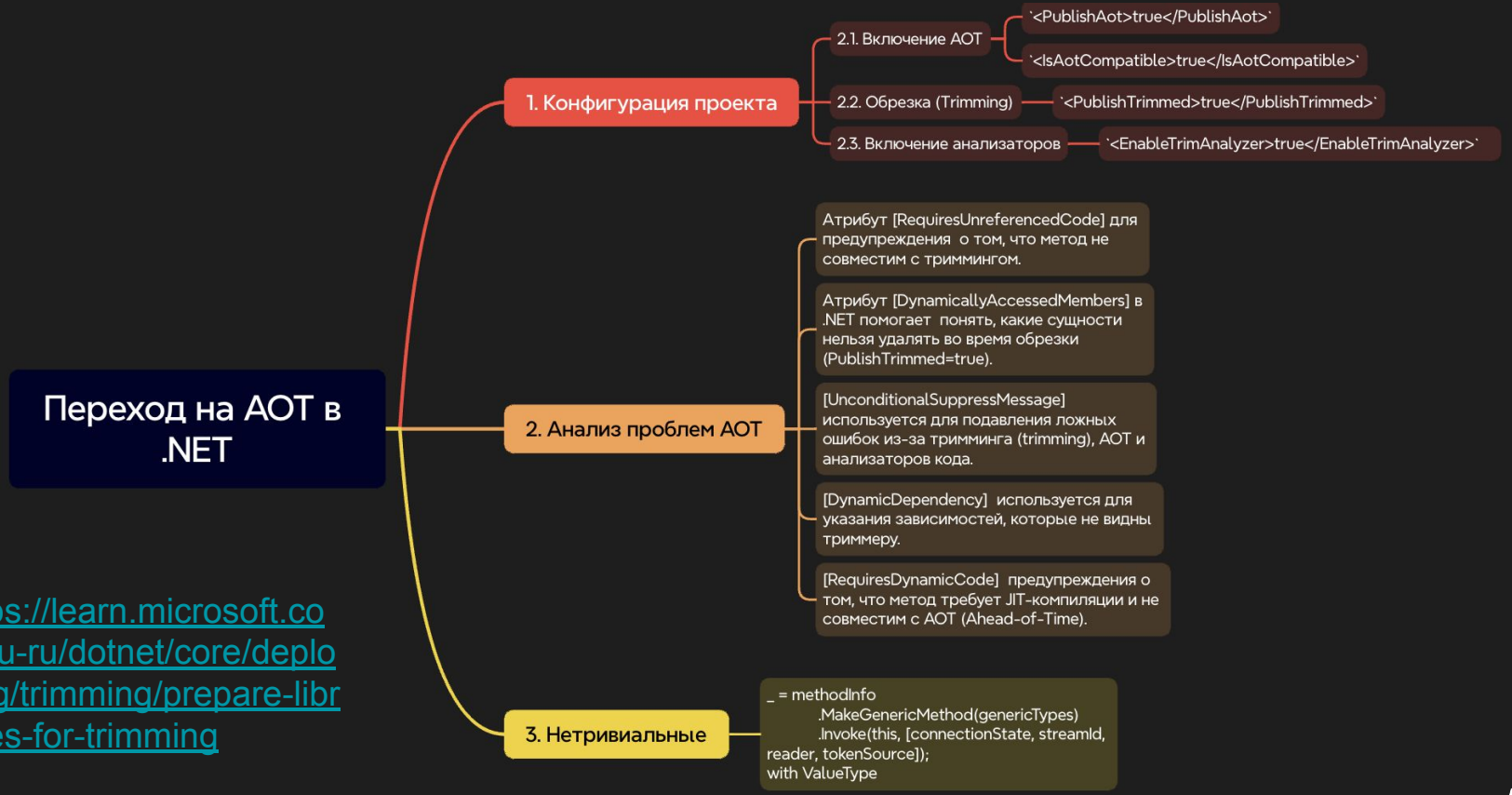


Server [56460]

Client [56079]

Artsofte

SignalR supports trimming and Native AOT



https://learn.microsoft.com/ru-ru/dotnet/core/deploying/trimming/prepare-libraries-for-trimming

Artsofte

SignalR supports trimming and Native AOT

```
/// <summary>
/// A base class for a SignalR hub.
/// </summary>
public abstract class Hub : IDisposable
{
    internal const DynamicallyAccessedMemberTypes DynamicallyAccessedMembers = DynamicallyAccessedMemberTypes.PublicConstructors | DynamicallyAccessedMemberTypes.PublicMethods;
```

```
using System.Diagnostics.CodeAnalysis;

namespace Microsoft.AspNetCore.SignalR.Internal;

[RequiresDynamicCode("Creating a proxy instance requires generating code at runtime")]
internal sealed class HubClients<THub, [DynamicallyAccessedMembers(DynamicallyAccessedMemberTypes.All)] T> : IHubClients<T> where THub : Hub
```

```
            var executor = IsCustomAwaitableSupported
                ? CreateObjectMethodExecutor(methodInfo, hubTypeInfo)
                : ObjectMethodExecutor.CreateTrimAotCompatible(methodInfo, hubTypeInfo);

            var authorizeAttributes = methodInfo.GetCustomAttributes<AuthorizeAttribute>(inherit: true);
            _methods[methodName] = new HubMethodDescriptor(executor, serviceProviderIsService, authorizeAttributes);
            _cachedMethodNames.Add(methodName);

            Log.HubMethodBound(_logger, hubName, methodName);
        }
    }

    [RequiresUnreferencedCode("Using SignalR with 'Microsoft.AspNetCore.SignalR.Hub.IsCustomAwaitableSupported=true' is not trim compatible.")]
    [RequiresDynamicCode("Using SignalR with 'Microsoft.AspNetCore.SignalR.Hub.IsCustomAwaitableSupported=true' is not native AOT compatible.")]
    private static ObjectMethodExecutor CreateObjectMethodExecutor(MethodInfo methodInfo, TypeInfo targetType)
        => ObjectMethodExecutor.Create(methodInfo, targetType);
```

Artsofte

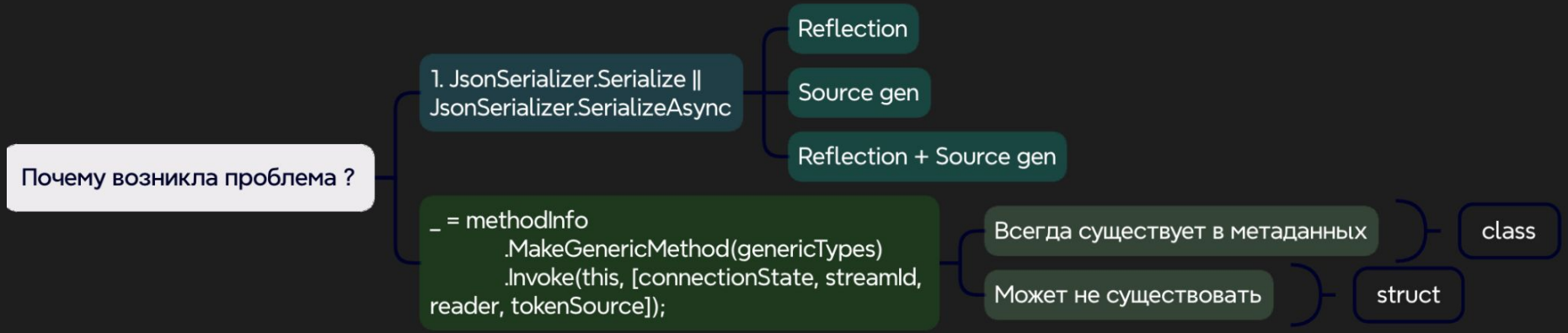
SignalR: Using IAsyncEnumerable<T> and ChannelReader<T> with ValueTypes in native AOT

```
    private void InvokeStreamMethod(MethodInfo methodInfo, Type[] genericTypes, ConnectionState
connectionState, string streamId, object reader, CancellationTokenSource tokenSource)
    {
#if NET
        Debug.Assert(genericTypes.Length == 1);

        if (!RuntimeFeature.IsDynamicCodeSupported && genericTypes[0].IsValueType)
        {
            // NativeAOT apps are not able to stream IAsyncEnumerable and ChannelReader of ValueTypes
            // since we cannot create SendStreamItems and SendIAsyncEnumerableStreamItems methods with
a generic ValueType.
            throw new InvalidOperationException($"Unable to stream an item with type
'{genericTypes[0]}' because it is a ValueType. Native code to support streaming this ValueType will not
be available with native AOT.");
        }
#endif

        _ = methodInfo
            .MakeGenericMethod(genericTypes)
            .Invoke(this, [connectionState, streamId, reader, tokenSource]);
    }
```

https://github.com/dotnet/aspnetcore/pull/56079

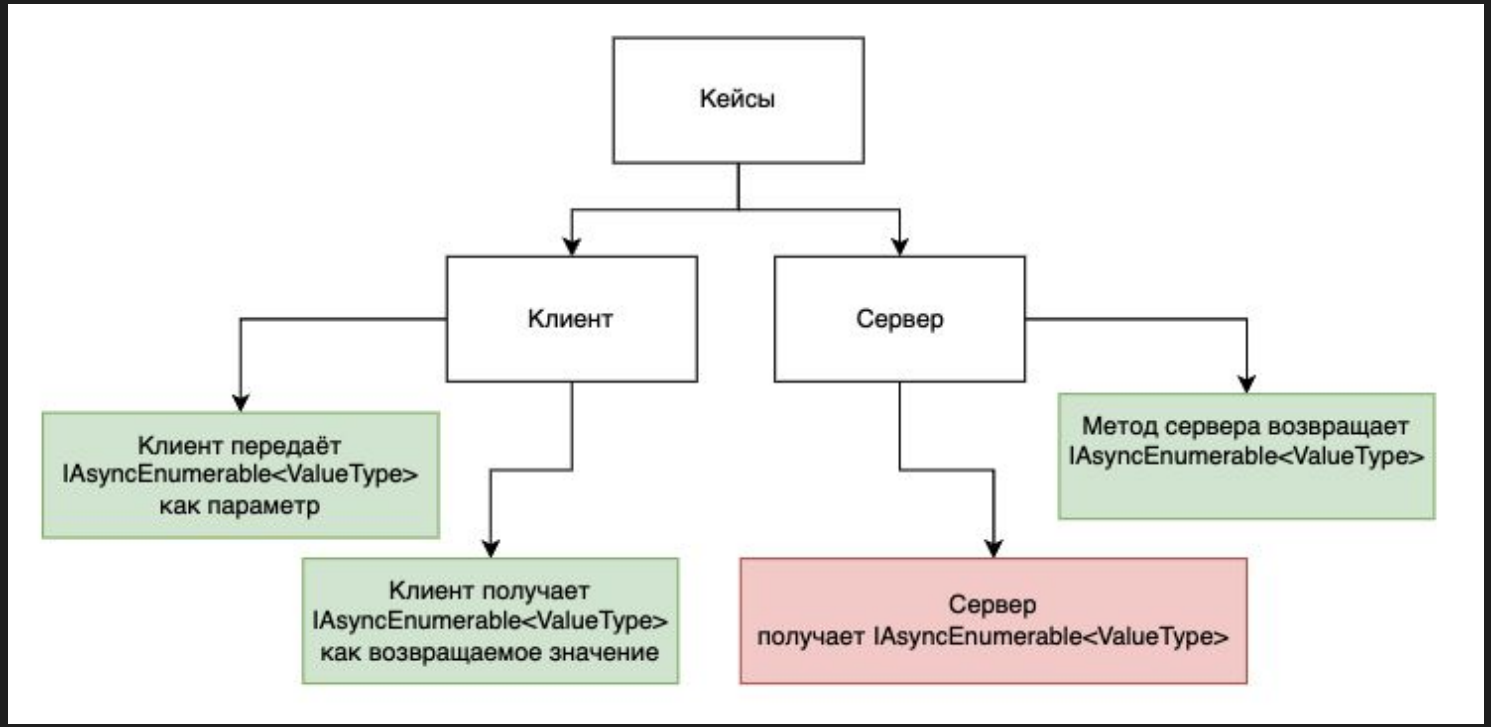
Artsofte

SignalR: Using IAsyncEnumerable<T> and ChannelReader<T> with ValueTypes in native AOT



Artsofte

SignalR: Using IAsyncEnumerable<T> and ChannelReader<T> with ValueTypes in native AOT



Artsofte

SignalR: Using IAsyncEnumerable<T> and ChannelReader<T> with ValueTypes in native AOT

```
private sealed class ReflectionAsyncEnumerator : IAsyncEnumerator<object?>
    {
        private static readonly MethodInfo _asyncEnumeratorMoveNextAsyncMethodInfo = typeof(IAsyncEnumerator<>).GetMethod("MoveNextAsync")!;
        private static readonly MethodInfo _asyncEnumeratorGetCurrentMethodInfo = typeof(IAsyncEnumerator<>).GetMethod("get_Current")!;

        private readonly object _enumerator;
        private readonly MethodInfo _moveNextAsyncMethodInfo;
        private readonly MethodInfo _getCurrentMethodInfo;

        public ReflectionAsyncEnumerator(object enumerator)
        {
            _enumerator = enumerator;

            var type = ReflectionHelper.GetIAsyncEnumeratorInterface(enumerator.GetType());
            _moveNextAsyncMethodInfo = (MethodInfo)type.GetMemberWithSameMetadataDefinitionAs(_asyncEnumeratorMoveNextAsyncMethodInfo)!;
            _getCurrentMethodInfo = (MethodInfo)type.GetMemberWithSameMetadataDefinitionAs(_asyncEnumeratorGetCurrentMethodInfo)!;
        }

        public object? Current => _getCurrentMethodInfo.Invoke(_enumerator, []);

        public ValueTask<bool> MoveNextAsync() => (ValueTask<bool>)_moveNextAsyncMethodInfo.Invoke(_enumerator, [])!;

        public ValueTask DisposeAsync() => ((IAsyncDisposable)_enumerator).DisposeAsync();
    }
```

https://github.com/dotnet/aspnetcore/pull/56583

Artsofte

Что еще сделали, но не сказали об этом в анонсе ?

Use new System.Net.ServerSentEvents package in SignalR

```
using HttpClient client = new();
using Stream stream = await client.GetStreamAsync("https://localhost:12345/sse");
await foreach (SseItem<string> item in SseParser.Create(stream).EnumerateAsync())
{
    Console.WriteLine(item.Data);
}
```

https://github.com/dotnet/aspnetcore/pull/56206

Что еще сделали, но не сказали об этом в анонсе ?

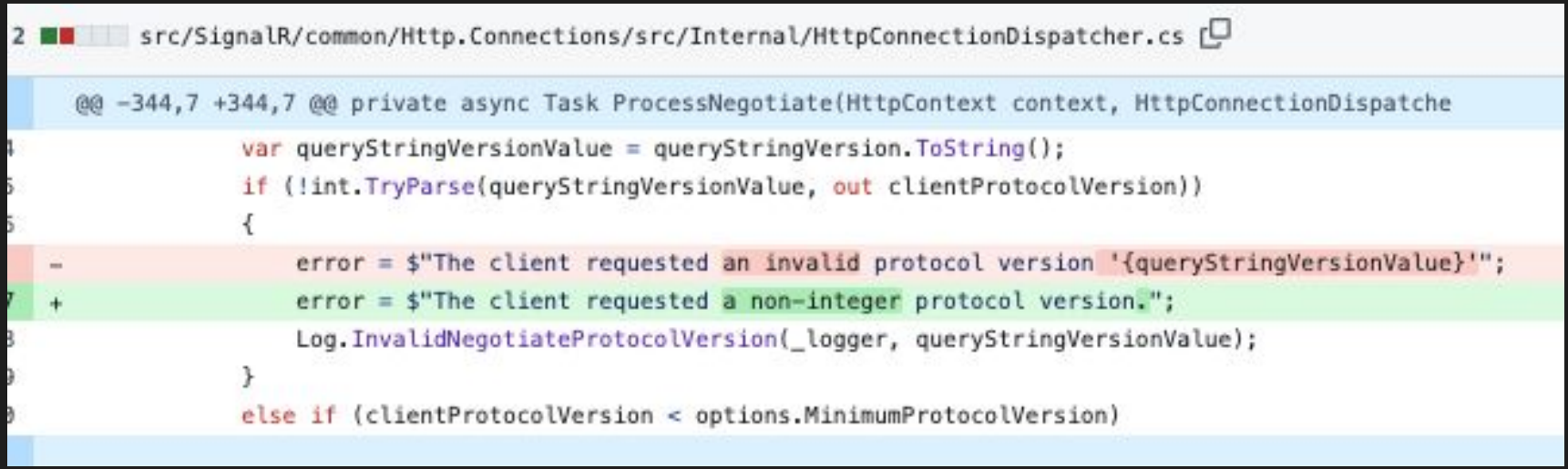
[release/9.0] Update dependencies from dotnet/runtime #57858

```
+ using System.Diagnostics.CodeAnalysis;
  using System.Dynamic;

  namespace Microsoft.AspNetCore.SignalR.Internal;
  @@ -9,6 +10,7 @@ internal sealed class DynamicClientProxy : DynamicObject
  {
      private readonly IClientProxy _clientProxy;

+     [RequiresDynamicCodeAttribute("This constructor requires dynamic code generation to construct a call site.")]
      public DynamicClientProxy(IClientProxy clientProxy)
      {
          _clientProxy = clientProxy;
```

https://github.com/dotnet/aspnetcore/pull/57858

Что еще сделали, но не сказали об этом в анонсе ?

Resolve potential XSS concerns from scanners in SignalR #57180



2 src/SignalR/common/Http.Connections/src/Internal/HttpConnectionDispatcher.cs

```
@@ -344,7 +344,7 @@ private async Task ProcessNegotiate(HttpContext context, HttpConnectionDispatche
            var queryStringVersionValue = queryStringVersion.ToString();
            if (!int.TryParse(queryStringVersionValue, out clientProtocolVersion))
            {
-               error = $"The client requested an invalid protocol version '{queryStringVersionValue}'";
+               error = $"The client requested a non-integer protocol version.";
                Log.InvalidNegotiateProtocolVersion(_logger, queryStringVersionValue);
            }
            else if (clientProtocolVersion < options.MinimumProtocolVersion)
```

https://github.com/dotnet/aspnetcore/pull/57180

Что еще сделали, но не сказали об этом в анонсе ?

Add partitioned to cookie for SignalR browser testing #57997

```
Set-Cookie: Beans=baked; SameSite=None; Secure; HttpOnly; Path=/; Partitioned;
```

/clients/ts/FunctionalTests/Startup.cs

```
84,11 @@ public void Configure(IApplicationBuilder app, IWebHostEnvironment env, ILog
    {
        cookieOptions.SameSite = Microsoft.AspNetCore.Http.SameSiteMode.None;
        cookieOptions.Secure = true;
        cookieOptions.Extensions.Add("partitioned"); // Required by Chromium

        expiredCookieOptions.SameSite = Microsoft.AspNetCore.Http.SameSiteMode.None;
        expiredCookieOptions.Secure = true;
        expiredCookieOptions.Extensions.Add("partitioned"); // Required by Chromium
    }
    context.Response.Cookies.Append("testCookie", "testValue", cookieOptions);
```

https://github.com/dotnet/aspnetcore/pull/57997

Что еще сделали, но не сказали об этом в анонсе ?

Annotate histograms with bucket boundaries #56500

```
internal static class MetricsConstants
{
    // Follows boundaries from http.server.request.duration/http.client.request.duration
    public static readonly IReadOnlyList<double> ShortSecondsBucketBoundaries = [0.005, 0.01, 0.025, 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1, 2.5, 5, 7.5, 10];

    // Not based on a standard. Larger bucket sizes for longer lasting operations, e.g. HTTP connection duration. See https://github.com/open-telemetry/semantic-conventions/issues/336
    public static readonly IReadOnlyList<double> LongSecondsBucketBoundaries = [0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10, 30, 60, 120, 300];
}
```

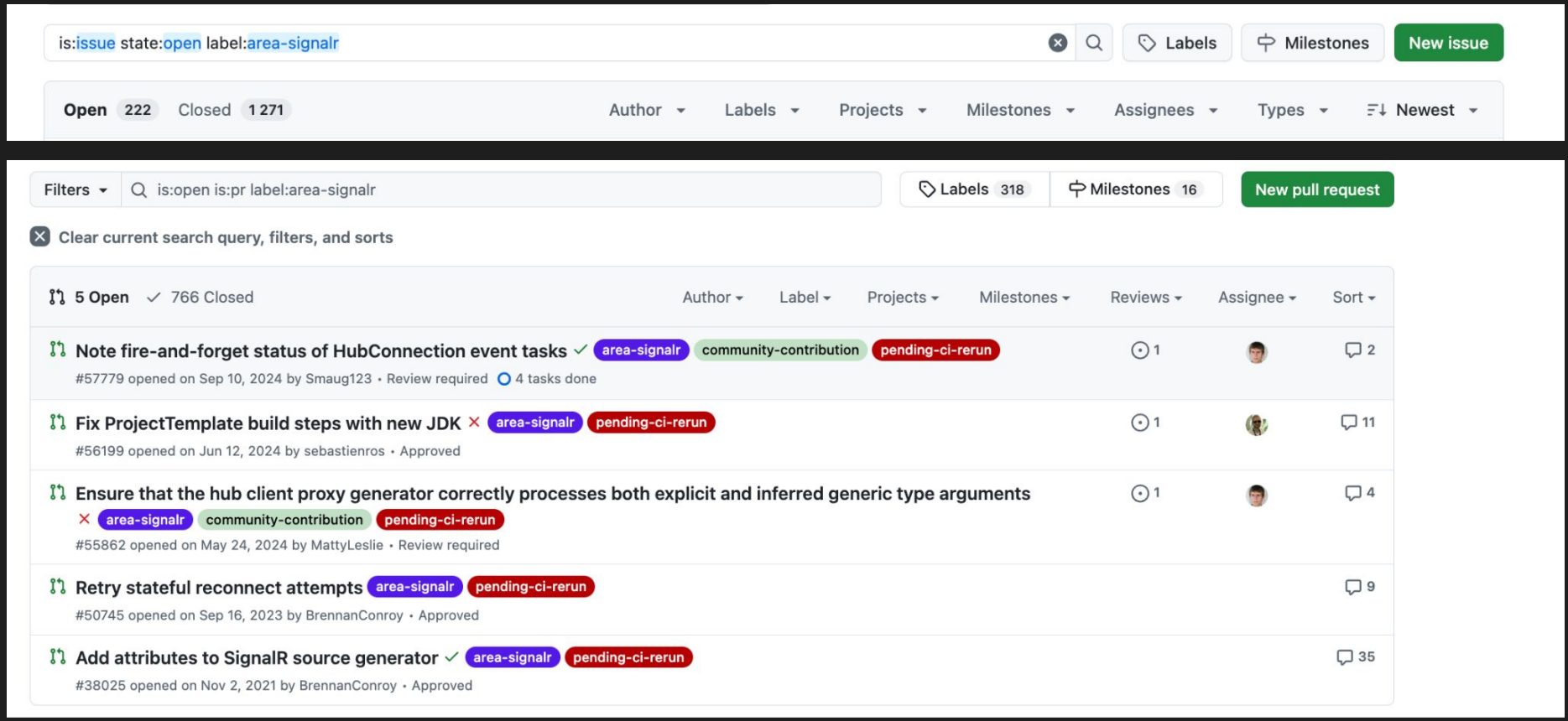
Почему это важно?

- Без бакетов: Гистограмма собирает только среднее значение, что теряет детализацию.
- С бакетами: Можно группировать метрики в разные временные диапазоны (0-1с, 1-5с, 5-10с и т. д.), улучшая детализацию мониторинга.

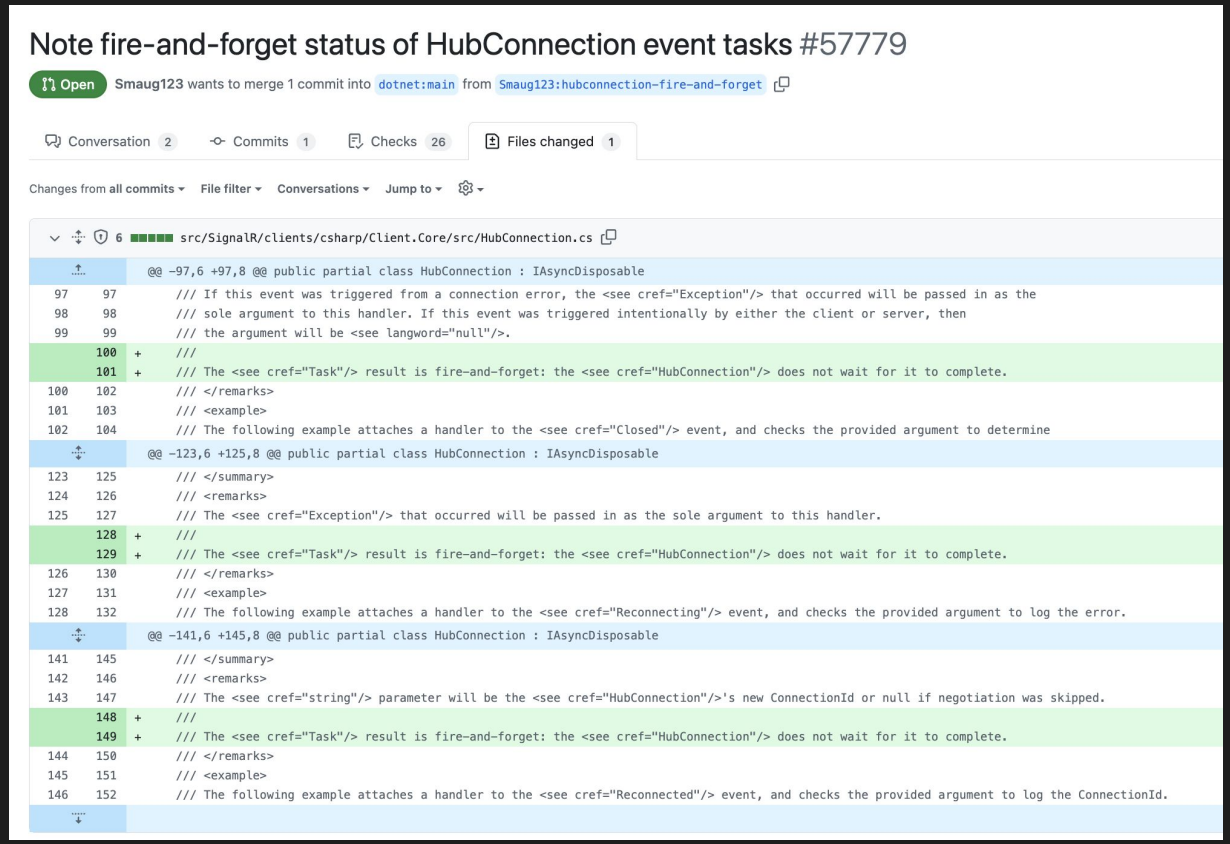
https://github.com/dotnet/aspnetcore/pull/56500

Artsofte

Что в работе ? 222 issue и 5 PR



Открытые PR



Note fire-and-forget status of HubConnection event tasks #57779

Open Smaug123 wants to merge 1 commit into dotnet:main from Smaug123:hubconnection-fire-and-forget

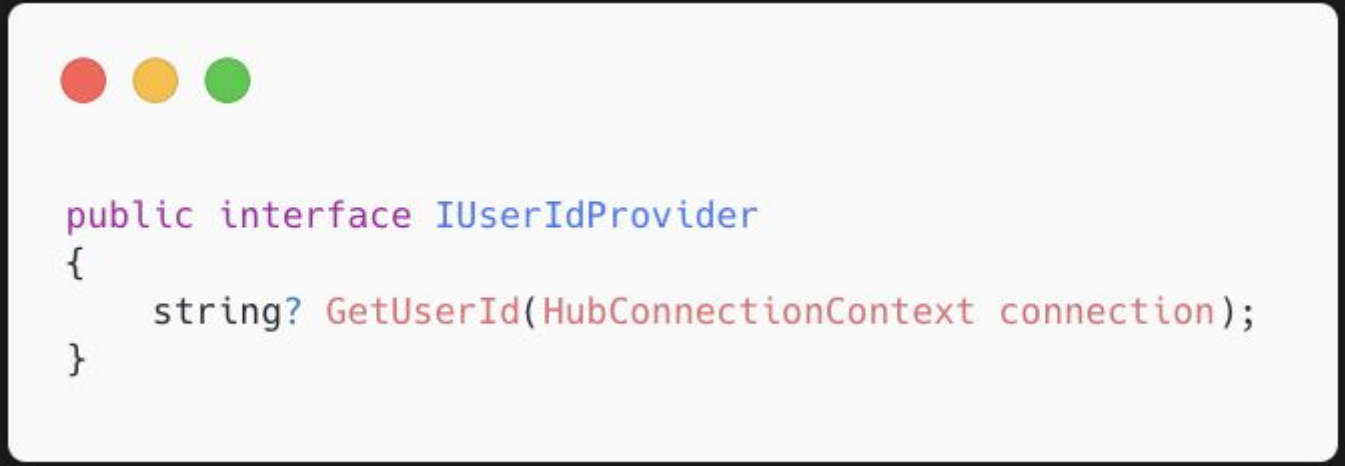
Conversation 2 Commits 1 Checks 26 Files changed 1

Changes from all commits File filter Conversations Jump to

6 src/SignalR/clients/csharp/Client.Core/src/HubConnection.cs

```
@@ -97,6 +97,8 @@ public partial class HubConnection : IAsyncDisposable
97  97      /// If this event was triggered from a connection error, the <see cref="Exception"/> that occurred will be passed in as the
98  98      /// sole argument to this handler. If this event was triggered intentionally by either the client or server, then
99  99      /// the argument will be <see langword="null"/>.
    100 +  ///
    101 +  /// The <see cref="Task"/> result is fire-and-forget: the <see cref="HubConnection"/> does not wait for it to complete.
100 102     /// </remarks>
101 103     /// <example>
102 104     /// The following example attaches a handler to the <see cref="Closed"/> event, and checks the provided argument to determine
@@ -123,6 +125,8 @@ public partial class HubConnection : IAsyncDisposable
123 125     /// </summary>
124 126     /// <remarks>
125 127     /// The <see cref="Exception"/> that occurred will be passed in as the sole argument to this handler.
    128 +  ///
    129 +  /// The <see cref="Task"/> result is fire-and-forget: the <see cref="HubConnection"/> does not wait for it to complete.
126 130     /// </remarks>
127 131     /// <example>
128 132     /// The following example attaches a handler to the <see cref="Reconnecting"/> event, and checks the provided argument to log the error.
@@ -141,6 +145,8 @@ public partial class HubConnection : IAsyncDisposable
141 145     /// </summary>
142 146     /// <remarks>
143 147     /// The <see cref="string"/> parameter will be the <see cref="HubConnection"/>'s new ConnectionId or null if negotiation was skipped.
    148 +  ///
    149 +  /// The <see cref="Task"/> result is fire-and-forget: the <see cref="HubConnection"/> does not wait for it to complete.
144 150     /// </remarks>
145 151     /// <example>
146 152     /// The following example attaches a handler to the <see cref="Reconnected"/> event, and checks the provided argument to log the ConnectionId.
```

Плохая поддержка multi-tenant

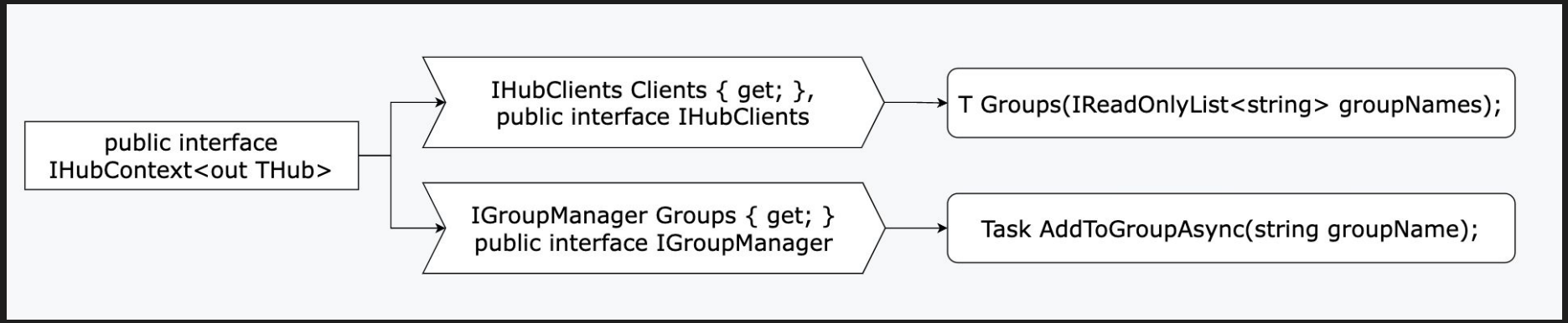


```
public interface IUserIdProvider
{
    string? GetUserId(HubConnectionContext connection);
}
```

```
connectionContext1.UserIdentifier = userIdProvider.GetUserId(connectionContext1);
IConnection
if ((keepAl
    connect
```

local variable HubConnectionContext **connectionContext1**

Плохая поддержка multi-tenant



Нет интерфейса, через который можно зарегистрировать перехват формирования имен групп

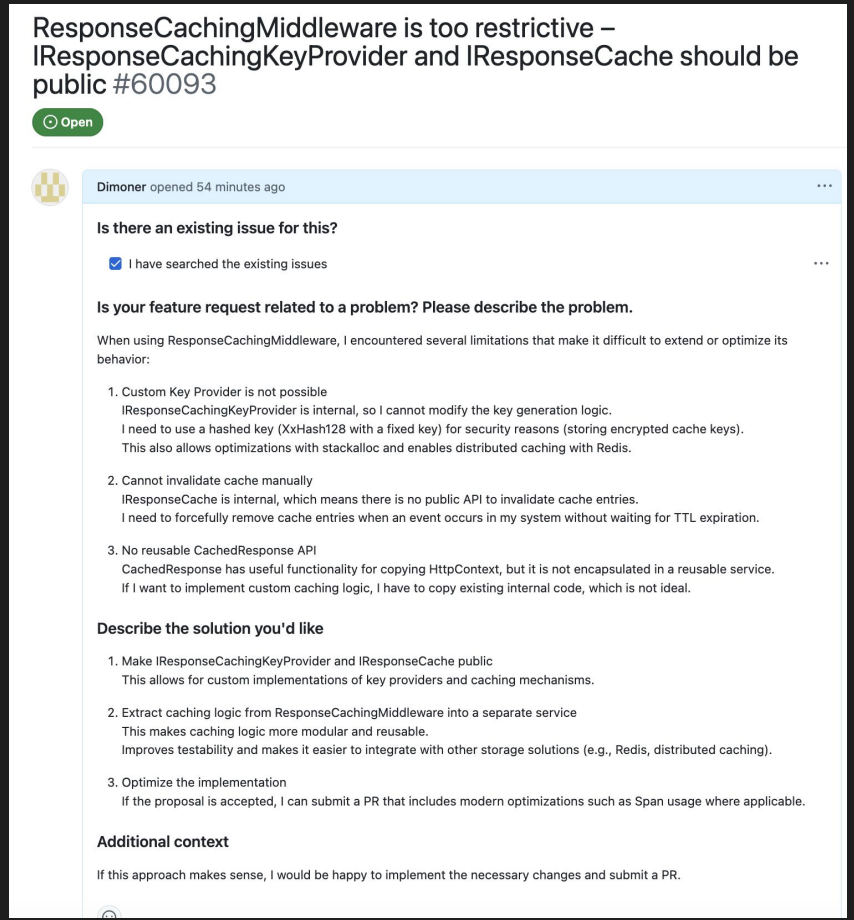
Нет встроенных счетчиков числа подключений

```
public class DefaultHubLifetimeManager<THub> : HubLifetimeManager<THub> where THub : Hub
  {
    #nullable disable
    private readonly HubConnectionStore _connections = new HubConnectionStore();
```

```
public class RedisHubLifetimeManager<THub> : HubLifetimeManager<THub>, IDisposable where THub : Hub
{
    private readonly HubConnectionStore _connections = new HubConnectionStore();
```

Что бы еще хотелось сделать ?

https://github.com/dotnet/aspnetcore/issues/60093



ResponseCachingMiddleware is too restrictive – IResponseCachingKeyProvider and IResponseCache should be public #60093

Open

Dimoner opened 54 minutes ago

Is there an existing issue for this?

- [x] I have searched the existing issues

Is your feature request related to a problem? Please describe the problem.

When using ResponseCachingMiddleware, I encountered several limitations that make it difficult to extend or optimize its behavior:

1. Custom Key Provider is not possible
   IResponseCachingKeyProvider is internal, so I cannot modify the key generation logic.
   I need to use a hashed key (XxHash128 with a fixed key) for security reasons (storing encrypted cache keys).
   This also allows optimizations with stackalloc and enables distributed caching with Redis.

2. Cannot invalidate cache manually
   IResponseCache is internal, which means there is no public API to invalidate cache entries.
   I need to forcefully remove cache entries when an event occurs in my system without waiting for TTL expiration.

3. No reusable CachedResponse API
   CachedResponse has useful functionality for copying HttpContext, but it is not encapsulated in a reusable service.
   If I want to implement custom caching logic, I have to copy existing internal code, which is not ideal.

Describe the solution you'd like

1. Make IResponseCachingKeyProvider and IResponseCache public
   This allows for custom implementations of key providers and caching mechanisms.

2. Extract caching logic from ResponseCachingMiddleware into a separate service
   This makes caching logic more modular and reusable.
   Improves testability and makes it easier to integrate with other storage solutions (e.g., Redis, distributed caching).

3. Optimize the implementation
   If the proposal is accepted, I can submit a PR that includes modern optimizations such as Span usage where applicable.

Additional context

If this approach makes sense, I would be happy to implement the necessary changes and submit a PR.

Artsofte