# Release History

## 1.38.0 (2024-02-26)

### Features Added

* Add GetRehydrationToken to Operation for rehydration purpose.

### Other Changes

* Additional Azure data centers are now included in AzureLocation. The following were added:
  + China East 3
  + China North 3
  + Israel Central
  + Italy North
  + Poland Central
  + Sweden South

## 1.37.0 (2024-01-11)

### Bugs Fixed

* Fixed exponential retry behavior so that delay milliseconds greater than Int32.MaxValue do not trigger an exception.
* Fixed DelayStrategy behavior to no longer shift the delay to be used over by one attempt. Previously, the first delay would be what should have been used for the second, and the second was what should have been used for the third, etc. Note, this would only be observed when using DelayStrategy outside of a RetryPolicy or RetryOptions.
* Do not add the error.type attribute twice when tracing is enabled.
* Do not suppress nested activities when they occur in the context of Consumer/Server activities (e.g. BlobClient.Download is no longer suppressed under EventHubs.Process).

### Other Changes

* Remove targets for .NET Core 2.1 and .NET 5 since they are out of support. Azure.Core is no longer compatible with .NET Core 2.1 after removal of target. The remaining targets are unchanged.

## 1.36.0 (2023-11-10)

### Features Added

* Added RequiresUnreferencedCode attribute to RequestContent.Create(object) overloads that use reflection to serialize the input object. This provides support for native AOT compilation when Azure.Core is used for diagnostics.
* Use System.Text.Json source generation to deserialize the error response in RequestFailedException on net6.0 and above targets.

### Breaking Changes

* Updated tracing attributes names to conform to OpenTelemetry semantic conventions version 1.23.0.
* Suppress client activity creation by Azure clients if it happens in scope of another activity created by an Azure client.
* Changed how ActivitySource name is constructed for clients that use single-worded activity names (without dot). We now append provided activity name as is to the client namespace name. Previously, the provided activity name was omitted and the ActivitySource name matched the provided client namespace.
* Distributed tracing with ActivitySource for HTTP and REST-based client libraries is declared stable. [Experimental feature-flag](https://github.com/Azure/azure-sdk-for-net/blob/main/sdk/core/Azure.Core/samples/Diagnostics.md) is no longer required for most of the newly released libraries. Tracing for messaging libraries remains experimental.
* Added nullable annotation to ResourceIdentifier.TryParse parameter input.

## 1.35.0 (2023-09-07)

### Features Added

* Expand the set of supported DynamicData property types to included heterogeneous arrays of allowed types.

### Breaking Changes

* Added the nullability annotation to NullableResponse<T>.Value to indicate that it is a nullable type.

## 1.34.0 (2023-07-11)

### Features Added

* Added IsCaeEnabled property to TokenRequestContext to enabled per-request support for Continuous Access Evaluation
* Updated dependency on System.Diagnostics.DiagnosticSource
* Added ContentLengthLong property to ResponseHeaders

## 1.33.0 (2023-06-16)

### Features Added

* Added BinaryData.ToDynamicFromJson() extension method to enable dynamic access to JSON. See the [aka.ms/azsdk/net/dynamiccontent](https://aka.ms/azsdk/net/dynamiccontent) for further details.

### Other Changes

* Client redirects are now disabled by default and can be enabled by setting providing a custom transport in ClientOptions'. Client Authors can also enable redirects by setting HttpPipelineTransportOptions.IsClientRedirectEnabledtotrueon the transport options passed toHttpPipelineBuilder.Build`.

## 1.32.0 (2023-05-09)

### Features Added

* Added the GetRawResponse method to RequestFailedException.
* Added overloads of Operation<T>.WaitForCompletion and Operation.WaitForCompletionResponse that take a DelayStrategy.

## 1.31.0 (2023-04-10)

### Features Added

* Added the RetryPolicy type which can be used to create a custom retry policy.
* Added the DelayStrategy type which can be used to customize delays.

### Bugs Fixed

* Set the Activity status to Error on failed activity source activities.
* Mark the Azure.Core.Http.Request span as failed if the request fails with an exception thrown in the pipeline.
* Fixed equality comparison when comparing a string to a ContentType instance.
* Jitter is added when using a RetryMode of Fixed.

## 1.30.0 (2023-03-09)

### Bugs Fixed

* Fixed the issue with empty header names and values, caused by ArrayBackedPropertyBag keeping reference to the array after returning it to array pool [in Dispose method](https://github.com/Azure/azure-sdk-for-net/pull/34800).

## 1.29.0 (2023-03-02)

### Features Added

* ActivitySource activities that are used when using the [experimental OpenTelemetry support](https://devblogs.microsoft.com/azure-sdk/introducing-experimental-opentelemetry-support-in-the-azure-sdk-for-net/) will include the az.schema\_url tag indicating the OpenTelemetry schema version. They will also include the attribute names specified [here](https://github.com/Azure/azure-sdk/blob/main/docs/tracing/distributed-tracing-conventions.yml).
* "West US 3", "Sweden Central" and "Qatar Central" locations are added to Azure.Core.AzureLocation

### Improvements

* Azure.Core.ArrayBackedPropertyBag is used to store request headers before HttpRequestMessage is created instead of System.Net.Http.Headers.HttpContentHeaders
* Azure.HttpRange.ToString uses string.Create instead of FormattableString.Invariant in .NET 6.0+
* Azure.Core.Diagnostics.AzureCoreEventSource checks EventLevel before formatting data for the events
* Azure.Core.Pipeline.HttpClientTransport.JoinHeaderValues uses System.Runtime.CompilerServices.DefaultInterpolatedStringHandler to join header string values in .NET 6.0+

### Bugs Fixed

* ActivitySource activities will no longer be stamped with the kind attribute as this is redundant with the OpenTelemetry SpanKind attribute.
* The product information section of the UserAgent header is now validated for invalid parenthesis formatting and escaped, if necessary.

## 1.28.0 (2023-02-06)

### Bugs Fixed

* Fixed an issue with AzureSasCredential which resulted in messages to fail authentication if the SAS signature was updated while a message was in a retry cycle.

## 1.27.0 (2023-01-10)

### Features Added

* Made RedirectPolicy public to provide SetAllowAutoRedirect() method to library authors.
* Added RetryPolicy property to ClientOptions to allow library authors to set a custom retry policy.
* Added MessageProcessingContext type and ProcessingContext property to HttpMessage which contains information about the message as it traverses through the pipeline.
* Added SetProperty and TryGetProperty overloads to HttpMessage to allow setting property values using a Type as the key.

## 1.26.0 (2022-11-08)

### Features Added

* Introduced a new NullableResponse<T> type for scenarios where a service method may or may not return a value. One common example is Get\*IfExists methods. Response<T> also now inherits from NullableResponse<T>.
* Added TryParse method to the ResourceIdentifier type.
* Added AppendQuery and AppendPath overloads to RequestUriBuilder.

### Bugs Fixed

* Fixed issue where fixed delay was applied when the RetryMode was set to Exponential when retrying a request that resulted in an exception.

### Other Changes

* Azure.Core now targets .NET 6 in addition to the existing targets.

## 1.25.0 (2022-06-23)

### Features Added

* Added RequestFailedDetailsParser abstract class, which client libraries can implement to control customization of exception messages for failed responses.
* Added HttpPipelineOptions type which is accepted in a new overload to HttpPipelineBuilder.Build. This type contains all the properties from other overloads and adds a property to specify a RequestFailedDetailsParser.
* Added a property to HttpPipelineTransportOptions called ClientCertificates which is a collection of X509Certificate2. If populated, the certificates in the collection will be used by the client for TLS client certificate authentication.
* Added the MultipartResponse type, which can be used by clients to parse the sub-responses for multi-part responses.

## 1.24.0 (2022-04-04)

### Features Added

* Added the MessageContent type which represents a message containing a content type and data.
* Sub classes of ClientOptions are now able to create sub class implementations of DiagnosticsOptions and set it as the implementation for the Diagnostics property of ClientOptions via a new constructor overload.

## 1.23.0 (2022-03-21)

### Features Added

* Added the TelemetryDetails type which enables customization of UserAgent header values on a per-request basis based on a specified Assembly and an optional application Id string.
* Added AddClassifier methods to RequestContext. These methods allow callers to change the response classification behavior for a given method invocation.
* Added a new StatusCodeClassifier type that will be used as the default ResponseClassifier for some libraries.
* Added an extension method to BinaryData called ToObjectFromJson which converts the json value represented by BinaryData to an object of a specific type.
* Additional data center locations were added to AzureLocation.
* Added WaitUntil enum to allow callers to set whether a method invoking a long running operation should return when the operation starts or once it has completed.

### Breaking Changes

* Cookies are no longer set on requests by default. Cookies can be re-enabled for HttpClientTransport by either setting an AppContext switch named "Azure.Core.Pipeline.HttpClientTransport.EnableCookies" to true or by setting the environment variable, "AZURE\_CORE\_HTTPCLIENT\_ENABLE\_COOKIES" to "true". Note: AppContext switches can also be configured via configuration like below:

<ItemGroup>

<RuntimeHostConfigurationOption Include="Azure.Core.Pipeline.HttpClientTransport.EnableCookies" Value="true" />

</ItemGroup>

## 1.22.0 (2022-01-11)

### Features Added

* Added AddPolicies method to RequestContext. This allows policies to be added to the pipeline when calling protocol methods.
* Added IsError property to Response. This will indicate whether the message's ResponseClassifier considers the response to be an error.
* Added RequestFailedException constructor that takes a Response.
* Added AzureLocation. This class gives static references to known Azure regions.
* Added ResourceIdentifier. This class allows users to load an Azure resource identifier string and parse out the pieces of that string such as which SubscriptionId does the resource belong to.
* Added ResourceType. This class represents the ARM provider information for a given resource and is used by the ResourceIdentifier class.
* Added HttpPipelineTransportOptions type. This type contains a ServerCertificateCustomValidationCallback property that allows callers to set a Func<ServerCertificateCustomValidationArgs, bool> delegate. If set, the delegate will be called to validate the server side TLS certificate.
* Added a new static overload for HttpPipelineBuilder.Build that takes an HttpPipelineTransportOptions instance. This overload creates an HttpPipeline with the default transport configuration and the HttpPipelineTransportOptions applied. It returns a DisposableHttpPipeline that implements IDisposable. Note: The HttpPipelineTransportOptions will not be applied if a custom Transport has been set in the ClientOptions. In the case that transport options were provided but not applied, an event is logged (PipelineTransportOptionsNotApplied).

### Breaking Changes

* Added logging of api-version query parameter by default. In order to redact this, you can do the following:

options.Diagnostics.LoggedQueryParameters.Remove("api-version");

### Bugs Fixed

* Fixed a bug where requests were failing with NotImplementedException on Unity with .NET Framework scripting.

## 1.21.0 (2021-11-03)

### Features Added

* Added RequestContext and ErrorOptions types to aid in configuring requests.
* Added ContentType strongly-typed string to allow operation callers to specify the content type of a request.

## 1.20.0 (2021-10-01)

### Features Added

* Added the static DelegatedTokenCredential type with a Create method, which returns an instance of TokenCredential that uses the supplied delgates to produce an AccessToken. This would most typically be used when an token has previously been obtained from some other source and that token needs to be returned by a TokenCredential instance.
* Added ResponseError type to represent an Azure error type.
* Added an experimental ActivitySource support.

### Bugs Fixed

* Fixed an exception during EventSource creation on Xamarin.

## 1.19.0 (2021-09-07)

### Features Added

* Added HttpAuthorization to represent authentication information in Authorization, ProxyAuthorization, WWW-Authenticate, and Proxy-Authenticate header values.

## 1.18.0 (2021-08-18)

### Bugs Fixed

* Fixed a bug where a buffered error responses on .NET Framework were prematurely disposed
* Fixed relative redirect support.

## 1.17.0 (2021-08-10)

### Features Added

* Added ClientOptions.Default to configure defaults process-wide.
* Added HttpPipelinePosition.BeforeTransport to be able to add policies at the end of the pipeline before the transport.

### Fixed

* Fixed NotSupportedException when running in Unity.

## 1.16.0 (2021-06-30)

### Changed

* Added TenantId to the properties on TokenRequestContext to enable multi-tenant support in Azure.Identity.

## 1.15.0 (2021-06-08)

### Features Added

* Types to represent GeoJson primitives.

### Changed

* Response.Content no longer throws InvalidOperationException when the response is backed by a MemoryStream with a non publicly visible buffer.

## 1.14.0 (2021-05-11)

### Features Added

* Added additional methods to BearerTokenAuthenticationPolicy, which enables creation of authentication policies that can handle challenges.

## 1.13.0 (2021-04-07)

### Key Bug Fixes

* Fixed NotSupportedException when running using Blazor in the browser.
* Disable the response caching and enable the streaming when running using Blazor in the browser.

## 1.12.0 (2021-04-06)

### Features Added

* Added HttpPipeline.CreateHttpMessagePropertiesScope that can be used to inject scoped properties into HttpMessage.

## 1.11.0 (2021-03-22)

### Features Added

* Operation base class for operations that do not return a value.
* Added Content property to Response which returns the body of the response as a BinaryData if the body is buffered.
* AzureNamedKeyCredential has been implemented to cover scenarios where services require that a shared key name and the key value be used as a component of the algorithm to form the authorization token.

### Key Bug Fixes

* Check the JsonIgnoreAttribute.Condition property added in .NET 5 when discovering members with JsonObjectSerializer.
* ETag now returns string.Empty if it is constructed with a null value.
* Keep-Alive connections are recycled every 300 seconds to observe DNS changes.

## 1.10.0 (2021-03-09)

## Features Added

* Added CloudEvent type based on the [CloudEvent spec](https://github.com/cloudevents/spec/).

## 1.9.0 (2021-02-09)

## Features Added

* Added Serialize overloads on ObjectSerializer that serialize to BinaryData.
* Added AzureCoreExtensions containing extensions methods for BinaryData that allow deserializing with an ObjectSerializer.

### Key Bug Fixes

* Avoid ObjectDisposedException when the request is cancelled during content upload over HTTPS.
* Fix exception while setting If-Modified-Since header on .NET Framework.

## 1.8.1 (2021-01-11)

### Key Bug Fixes

* Include Microsoft.Bcl.AsyncInterfaces dependency on .NET 5 to avoid build issues in applications targeting .NET 5.

## 1.8.0 (2021-01-06)

### Features Added

* AzureSasCredential and its respective policy.

### Key Bug Fixes

* Avoid a causing and ignoring an exception when setting network stream timeout on .NET Core.

## 1.7.0 (2020-12-14)

### New Features

* System.Text.Json.JsonConverter implementation for the ETag
* Synchronous HttpClient support on .NET 5.

### Key Bug Fixes

* System proxy settings are correctly applied on .NET Framework

## 1.6.0 (2020-10-28)

### Features Added

* The HttpClientTransport(HttpMessageHandler) constructor overload.
* The JsonPatchDocument type.

### Key Bugs Fixed

* The race condition in AzureEventSourceListener class that sometimes resulted in a NullReferenceException in the EventSource.
* The overflow exception when content length is larger than int.MaxValue.

## 1.5.1 (2020-10-01)

### Changed

* ServicePointManager Connection limit is automatically increased to 50 for Azure endpoints.

## 1.5.0 (2020-09-03)

### Changed

* ETag now supports weak ETags and implements an overload for ToString that accepts a format string.

### Features Added

* HttpWebRequest-based transport implementation. Enabled by-default on .NET Framework. Can be disabled using AZURE\_CORE\_DISABLE\_HTTPWEBREQUESTTRANSPORT environment variable or Azure.Core.Pipeline.DisableHttpWebRequestTransport AppContext switch. To use the app context switch add the following snippet to your .csproj:

<ItemGroup>

<RuntimeHostConfigurationOption Include="Azure.Core.Pipeline.DisableHttpWebRequestTransport" Value="true" />

</ItemGroup>

When the environment variable or the switch are set the HttpClientTransport would be used by default instead.

## 1.4.1 (2020-08-18)

### Key Bugs Fixed

* Bug in TaskExtensions.EnsureCompleted method that causes it to unconditionally throw an exception in the environments with synchronization context

## 1.4.0 (2020-08-06)

### Features Added

* Added ObjectSerializer base class for serialization.
* Added IMemberNameConverter for converting member names to serialized property names.
* Added JsonObjectSerializer that implements ObjectSerializer for System.Text.Json.

### Key Bugs Fixed

* Connection leak for retried non-buffered requests on .NET Framework.

## 1.3.0 (2020-07-02)

### Features Added

* HttpPipeline.CreateClientRequestIdScope method to allow setting client request id on outgoing requests.

## 1.2.2 (2020-06-04)

### Key Bugs Fixed

* Retry server timeouts on .NET Framework.

## 1.2.1 (2020-04-30)

### Changed

* Read client request ID value used for logging and tracing off the initial request object if available.

### Key Bugs Fixed

* Fixed a bug when using Azure.Core based libraries in Blazor WebAssembly apps.

## 1.2.0 (2020-04-03)

### Features Added

* AzureKeyCredential and its respective policy.

### Changed

* Response trace messages are properly identified.
* Content type "application/x-www-form-urlencoded" is decoded in trace messages.

## 1.1.0 (2020-03-05)

### Fixes and improvements

* Add OPTIONS and TRACE HTTP request methods.
* Add NetworkTimeout property to RetryOptions and apply it to network operations like sending request or reading from the response stream.
* Implement serialization for RequestFailedException.

## 1.0.2 (2020-01-10)

* Block bearer token authentication for non TLS protected endpoints.
* Add support for retrying on request timeouts.
* Add support for retrying on 408, 500, 502, 504 status codes.
* Remove commit hash from User-Agent telemetry.

## 1.0.1

* Fix issues with log redaction where first query character was replaced with '?' character.
* Exclude EventCounter events from AzureEventSourceListener.
* Add AZURE\_TRACING\_DISABLED environment variable support.

## 1.0.0

* Updating versioning and packaging for general availability.
* Make types and namespace names consistent.

## 1.0.0-preview.9

* Added console and trace logger listener.
* Added additional content and header logging options.
* Moved commonly used types to Azure namespace.

## 1.0.0-preview.8

* Minor improvements and bug fixes.

## 1.0.0-preview.7

* Support for distributed tracing added.
* Support for TokenCredential in ASP.NET Core integration added. Shared types for long running operations and async collections added.
* .NET Core dependencies updated to preview7.
* Bug fixes.