

Retry strategy CHEAT SHEET



Basics

This reactive resilience strategy **allows you to re-perform the same action** if the execution fails.

You can configure the behaviour of the strategy via the **RetryStrategyOptions{<T>}** object.

N maximum retries means at most N+1 attempts. The plus one is the original attempt.

If all attempts fail, then the strategy will return with the *last* result or exception.

Specify unconditional instant retries

```
new ResiliencePipelineBuilder()
    .AddRetry(new RetryStrategyOptions()
    {
        ShouldHandle =
            _ => PredicateResult.True(),
        Delay = TimeSpan.Zero
    })
```

Specify retry in case of exception or failure result

```
new ResiliencePipelineBuilder<string>()
    .AddRetry(new RetryStrategyOptions<string>()
    {
        ShouldHandle = new PredicateBuilder<string>()
            .Handle<CustomException>()
            .HandleResult(string.IsNullOrEmpty)
    })
```

Specify indefinite retries with asynchronous notification

```
new ResiliencePipelineBuilder()
    .AddRetry(new RetryStrategyOptions()
    {
        MaxRetryAttempts = int.MaxValue,
        OnRetry = async args => await NotifyAsync(args.AttemptNumber)
    })
```

Specify sleep duration dynamically based on HTTP Retry-After header

```
new ResiliencePipelineBuilder<HttpResponseMessage>()
    .AddRetry(new RetryStrategyOptions<HttpResponseMessage>()
    {
        DelayGenerator = args => ValueTask.FromResult<TimeSpan?>(
            args.Outcome.Result.Headers.RetryAfter.Delta
            ?? TimeSpan.FromSeconds(2))
    })
```

Specify exponential backoff with capped delays

```
new ResiliencePipelineBuilder()
    .AddRetry(new RetryStrategyOptions()
    {
        BackoffType = DelayBackoffType.Exponential,
        UseJitter = true,
        MaxRetryAttempts = 10,
        MaxDelay = TimeSpan.FromSeconds(10)
    })
```