# Outcome chaos strategy **CHEAT SHEET**

## Basics

This reactive chaos **strategy injects outcome(s)** (result and/or Exception) to simulate an unexpected response.

You can configure the behaviour of the strategy via the **ChaosOutcomeStrategyOptions<T>** object.

## Specify single result

```
new ResiliencePipelineBuilder<HttpResponseMessage>()
  .AddChaosOutcome(0.1,
    () => new HttpResponseMessage(
      HttpStatusCode.InternalServerError))
```

## Specify a delegate for injection notification

```
new ResiliencePipelineBuilder<HttpResponseMessage>()
  .AddChaosOutcome(new
ChaosOutcomeStrategyOptions<HttpResponseMessage>
  {
    OnOutcomeInjected = static async args =>
      await NotifyAsync(args.Outcome)
  })
```

## Specify multiple results with switch expression

```
new ResiliencePipelineBuilder<HttpResponseMessage>()
  .AddChaosOutcome(new ChaosOutcomeStrategyOptions<HttpResponseMessage>
  {
    OutcomeGenerator = static _ =>
    {
      var rnd = Random.Shared.NextDouble();
      HttpStatusCode statusCode = rnd switch
      {
        < 0.4 => HttpStatusCode.InternalServerError,
        _ => HttpStatusCode.RequestTimeout
      };
      var outcome = Outcome.FromResult(new HttpResponseMessage(status));
      return ValueTask.FromResult<Outcome<HttpResponseMessage>?>(outcome);
    }
  })
```

## Specify multiple results with OutcomeGenerator

```
new ResiliencePipelineBuilder<HttpResponseMessage>()
  .AddChaosOutcome(new ChaosOutcomeStrategyOptions<HttpResponseMessage>
  {
    OutcomeGenerator = new OutcomeGenerator<HttpResponseMessage>()
      .AddResult(() =>
        new HttpResponseMessage(HttpStatusCode.InternalServerError),
        weight: 40)
      .AddResult(() =>
        new HttpResponseMessage(HttpStatusCode.RequestTimeout),
        weight: 60)
})
```