

# Fault Tolerant Web Service Requests with Polly

---

LEARNING THE BASICS OF THE POLLY FRAMEWORK



**Bryan Hogan**

SOFTWARE ARCHITECT

@bryanjhogan NoDogmaBlog.bryanhogan.net



# Introduction



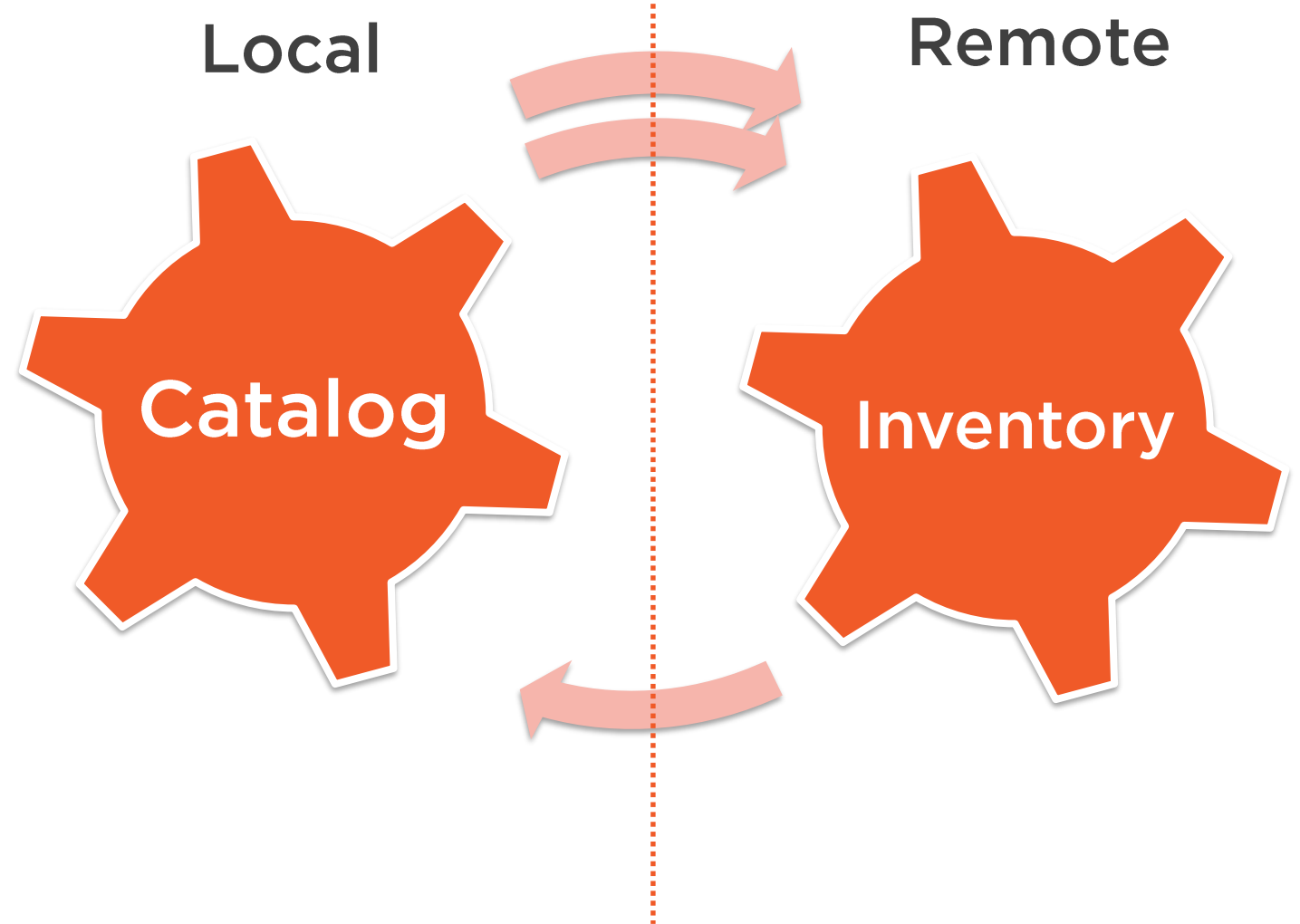
Resiliency

Transient failures

Longer outages



# Calling Remote Services



Why you need Polly

Retries, Fallbacks,  
Timeouts

Reusing policies

Unit testing

Combining policies

Finer control of policy

Caching

Stopping all requests

Application stability

## Course Overview



# Why Web Service Requests Fail

23094832	23355647	94643456	48355551	23094832	46478126	55280457	50845634
98564875	65861111	48916498	11001018	98564875	17823156	56788832	32535466
56732149	87484647	68456343	64578956	56732149	73213321	13321672	89355644
0-932476	81261782	25354668	87776886	0-932476	67223154	23154312	57577658
98345656	31115673	93556476	65478516	98345656	31221453	21453201	61111874
00874768	21332167	58647869	55844551	00874768	20113156	18831565	84647812
55647016	22315431	68866874	61494643	55647016	54345874	43577458	61743122
89355647	22145321	84610478	45648964	89355647	98645789	74986457	14532131
65864786	31500065	12617823	98684563	65864786	56877768	74457458	50006565
96880006	65434587	15673213	43253546	96880006	86065478	89568777	43458749
68748464	49861111	15673213	43253546	68748464	49861111	15673213	43253546
78126178	15673213	43253546	96880006	78126178	15673213	43253546	96880006
23156732	87484647	68456343	64578956	23156732	87484647	68456343	64578956
13321672	78951655	47851655	15673213	13321672	78951655	47851655	15673213
23154312	76880006	65434587	15673213	23154312	76880006	65434587	15673213
21453201	78516558	47851655	15673213	21453201	78516558	47851655	15673213
13156543	44551614	84455161	32110672	13156543	44551614	84455161	32110672
45874986	94643456	49464345	64564868	45874986	94643456	49464345	64564868
45789568	48916498	64896498	57875867	45789568	48916498	64896498	57875867
77768860	68456343	68456343	88944334	77768860	68456343	68456343	88944334
65478516	25354668	25354668	21010011	65478516	25354668	25354668	21010011
55844551	93556476	93556476	56456475	55844551	93556476	93556476	56456475
06149464	58647869	58647869	61249765	06149464	58647869	58647869	61249765
34564896	68866874	68866874	91093485	34564896	68866874	68866874	91093485
49860845	84610478	84647812	39201010	49860845	84610478	84647812	39201010
63432535	12617823	61782315	10048924	63432535	12617823	61782315	10048924

**SYSTEM FAILURE**





# Why Web Service Requests Fail



# Why Web Service Requests Fail

## Server Error in '/' Application.

*Object reference not set to an instance of an object.*

**Description:** An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

**Exception Details:** System.NullReferenceException: Object reference not set to an instance of an object.

**Source Error:**

An unhandled exception was generated during the execution of the current web request. Information regarding the origin and location of the exception can be identified using the exception stack trace below.

**Stack Trace:**

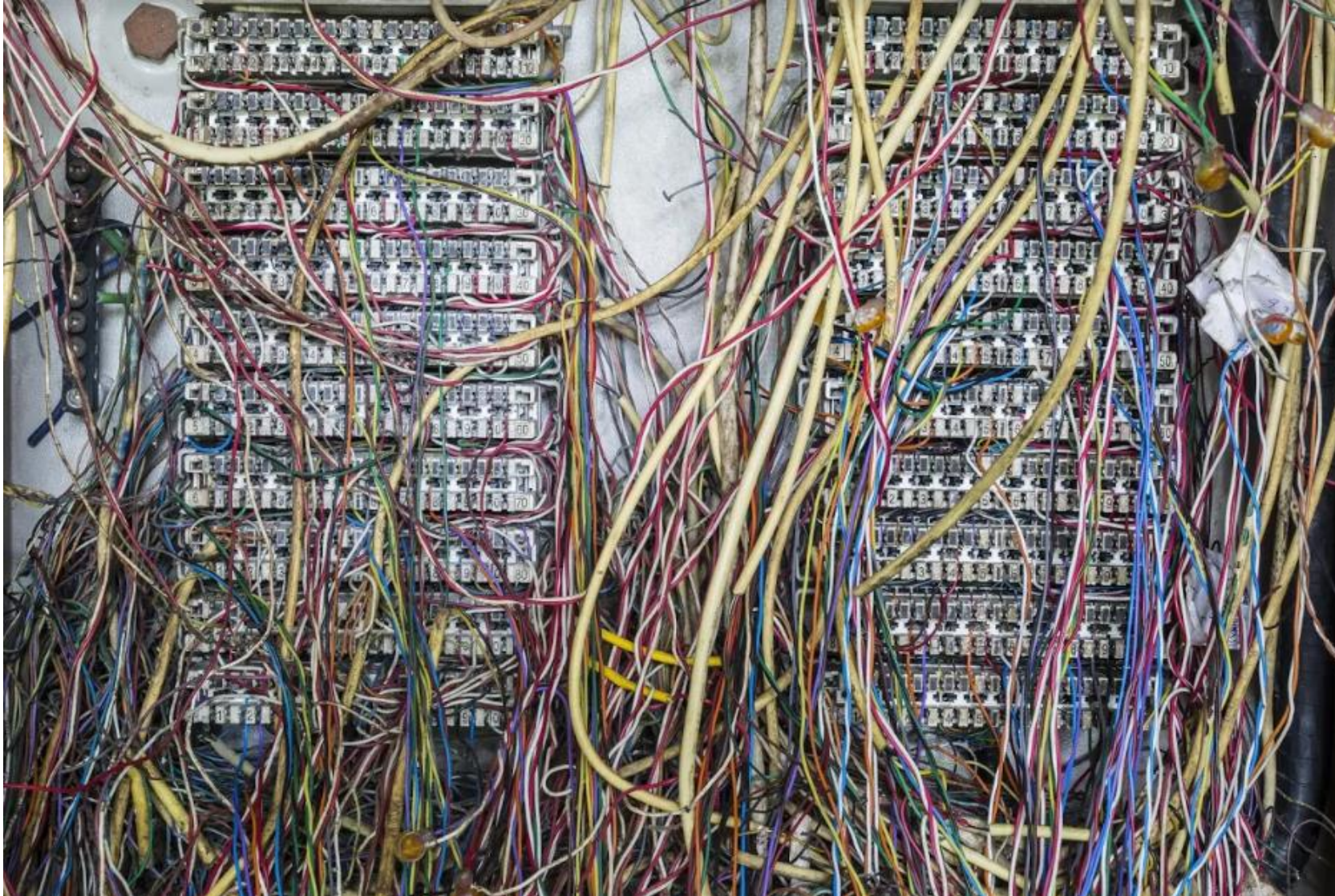
```
[NullReferenceException: Object reference not set to an instance of an object.]
  WebApplication1.Contact.Page_Load(Object sender, EventArgs e) +59
  System.Web.Util.CalliEventHandlerDelegateProxy.Callback(Object sender, EventArgs e) +51
  System.Web.UI.Control.OnLoad(EventArgs e) +95
  System.Web.UI.Control.LoadRecursive() +59
  System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean includeStagesAfterAsyncPoint) +678
```

**Version Information:** Microsoft .NET Framework Version:4.0.30319; ASP.NET Version:4.7.2046.0





# Why Web Service Requests Fail





# Why Web Service Requests Fail



# Why Web Service Requests Fail

```
Pinging 192.168.1.2 with 32 bytes of data:  
Reply from 192.168.1.7: Destination host unreachable.  
Reply from 192.168.1.1: Destination host unreachable.  
Reply from 192.168.1.1: Destination host unreachable.  
Reply from 192.168.1.1: Destination host unreachable.  
  
Ping statistics for 192.168.1.2:  
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```





# Polly Strategies

## **Reactive**

Responds to a current  
problem

## **Proactive**

Monitors for potential  
problems and acts  
ahead





# Reactive Strategies

## Retry

Retries Immediately

## Wait and Retry

Wait before resending  
request

## Circuit Breaker

Stops all request to a  
faulty service

## Fallback

Return a default value



# Proactive Strategies

## Timeout

End a request when you want

## Caching

Store a previous response

## Bulkhead isolation

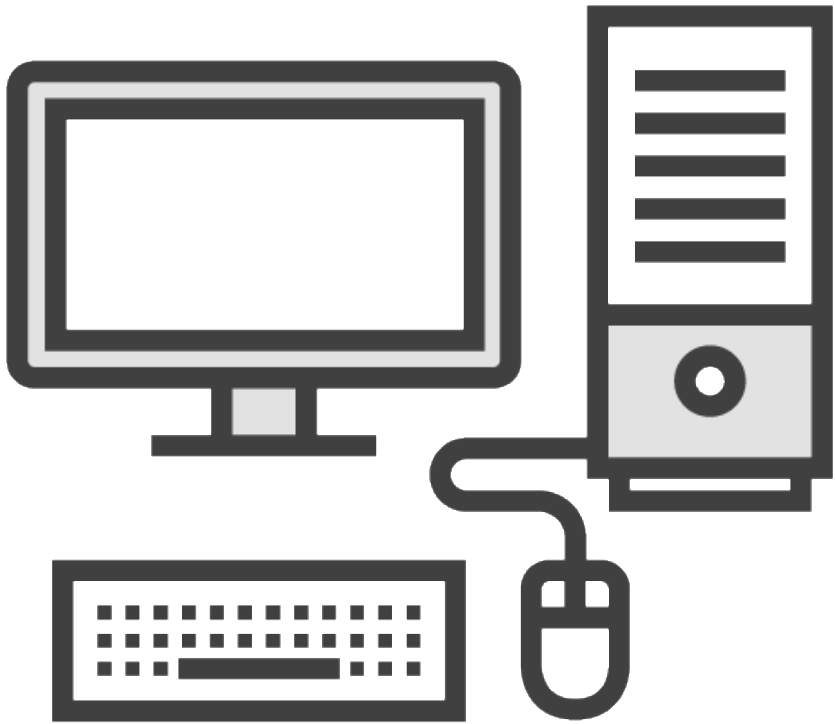
Prevent faults from bringing whole system down



# Quick Demo



# Prerequisites and Tools to Follow Along



Visual Studio 2017

Polly  $\geq 5.5$

.NET Core 2

Fiddler



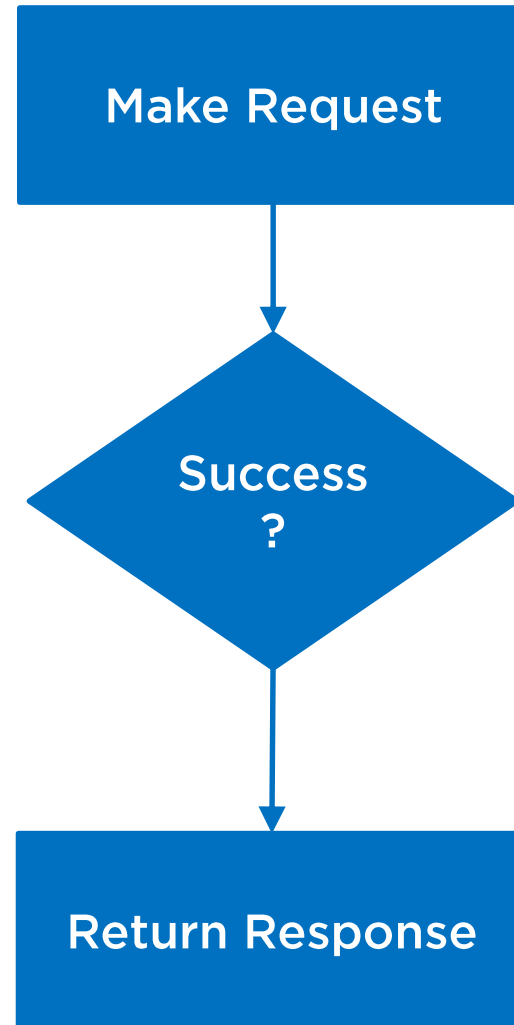
# What is a Resilience Framework?

**Recovery if possible**

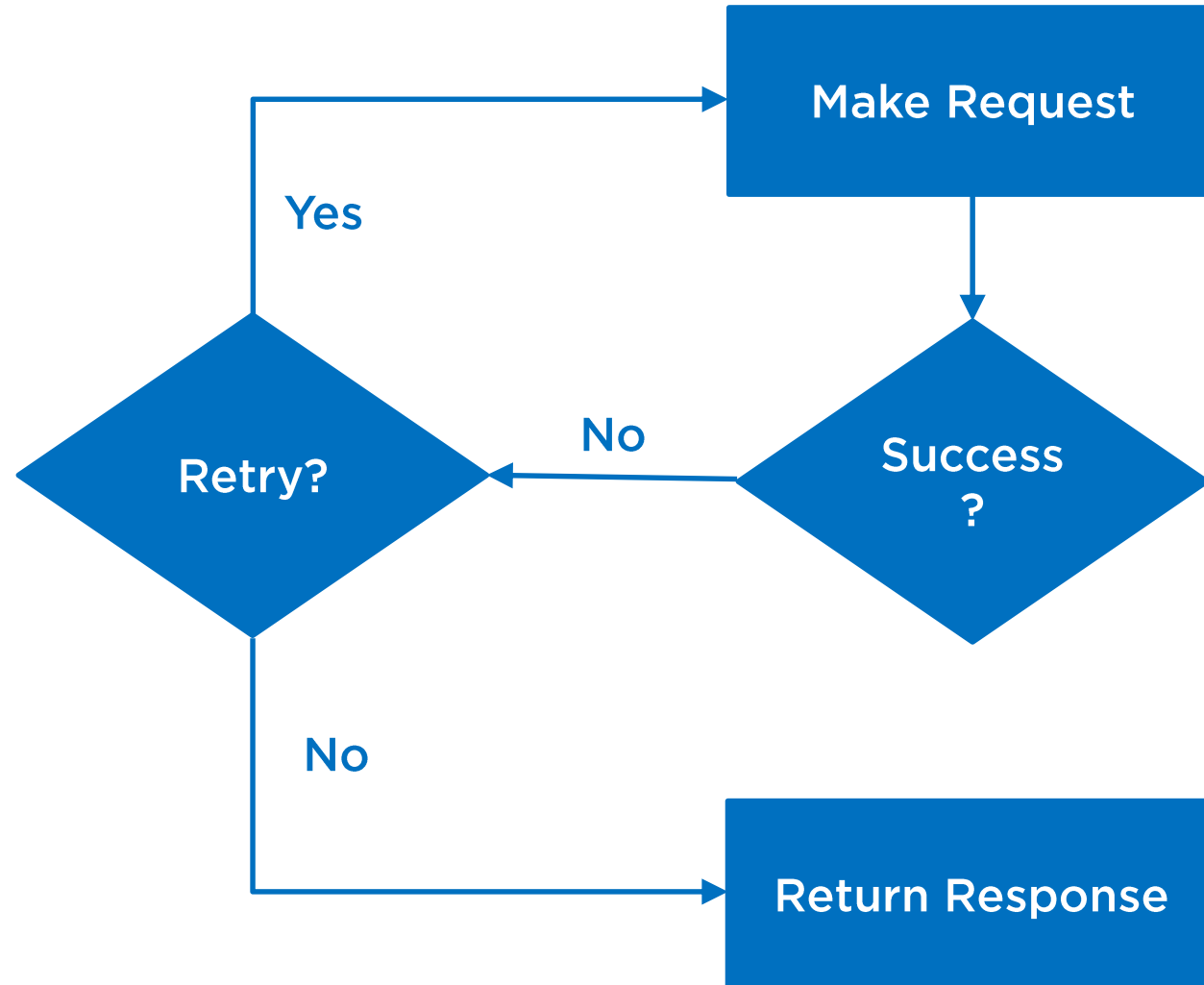
**Graceful degradation if no  
other option**



# Retry Policy in Action



# Retry Policy in Action



# A Quick Note on Policy Interfaces

**IAsyncPolicy**

**VS.**

**RetryPolicy**





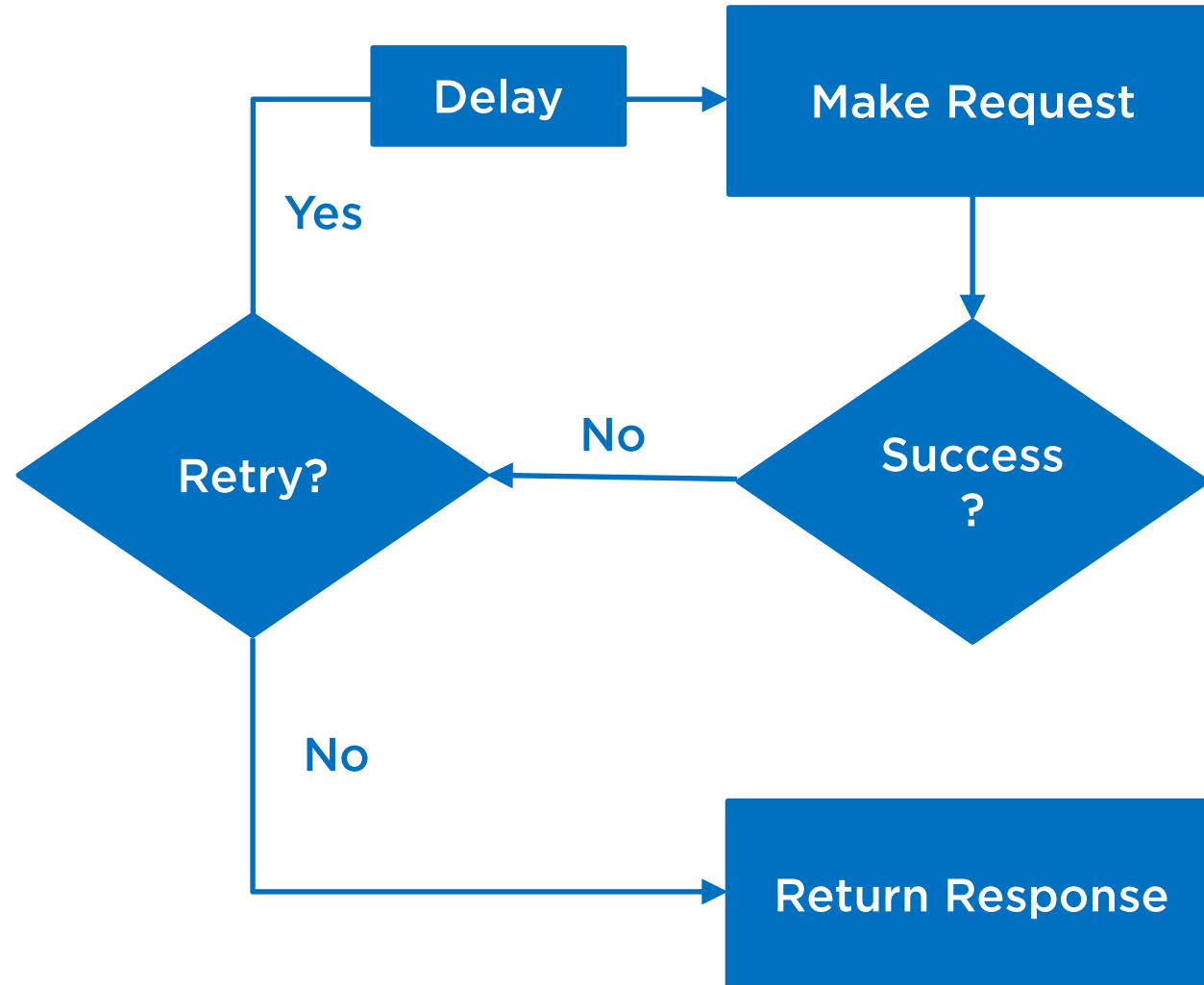
# Retry, Retry, Retry, Retry...



# Wait and Retry - Backoffs



# Wait and Retry Policy in Action



# Idempotence





Delegates

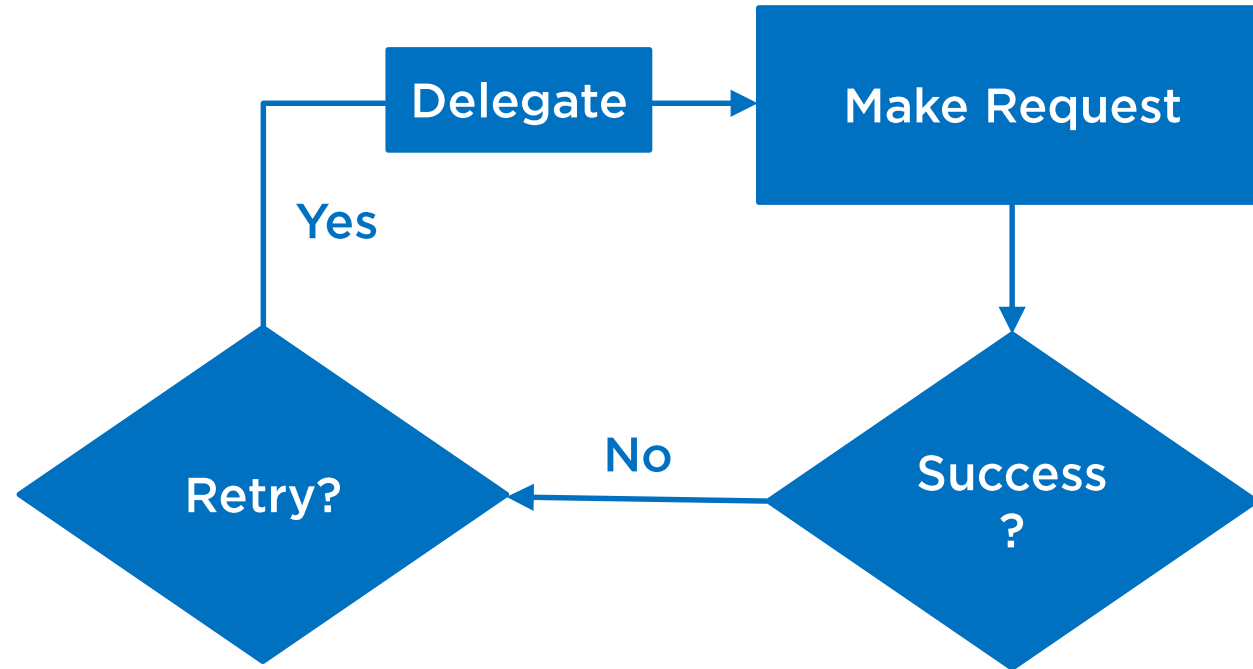
Retry Policy

Wait and  
Retry Policy

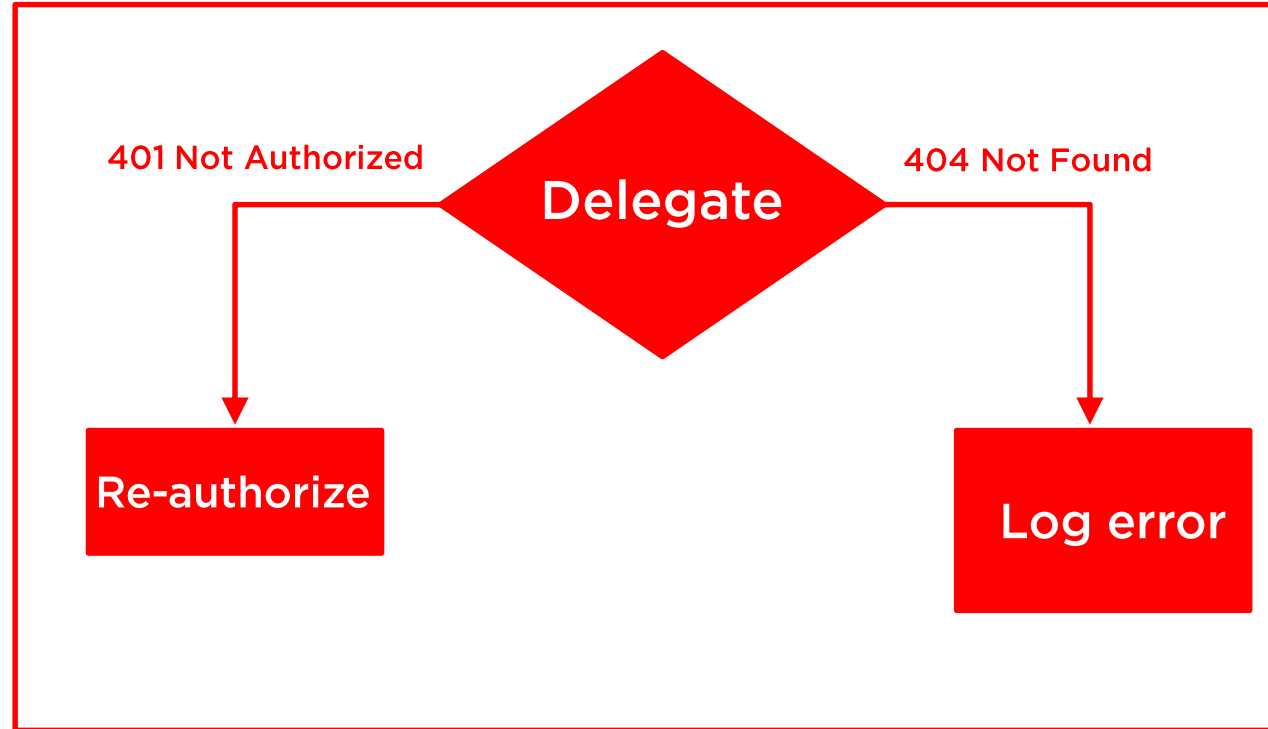
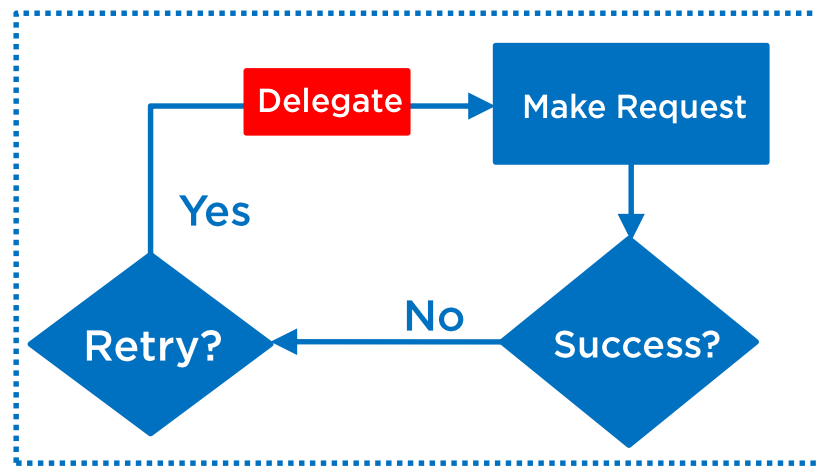
OnRetry  
delegate



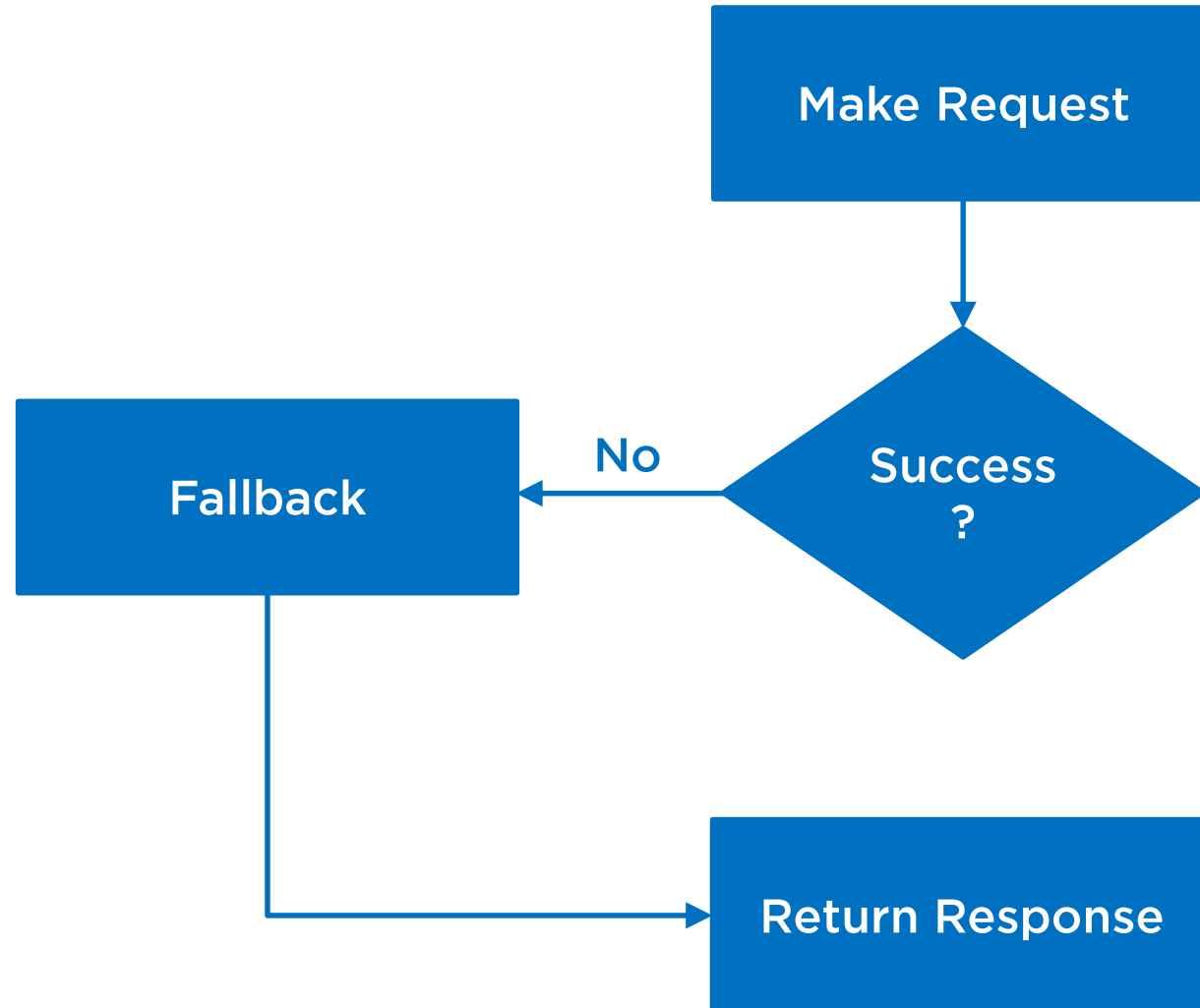
# Delegates



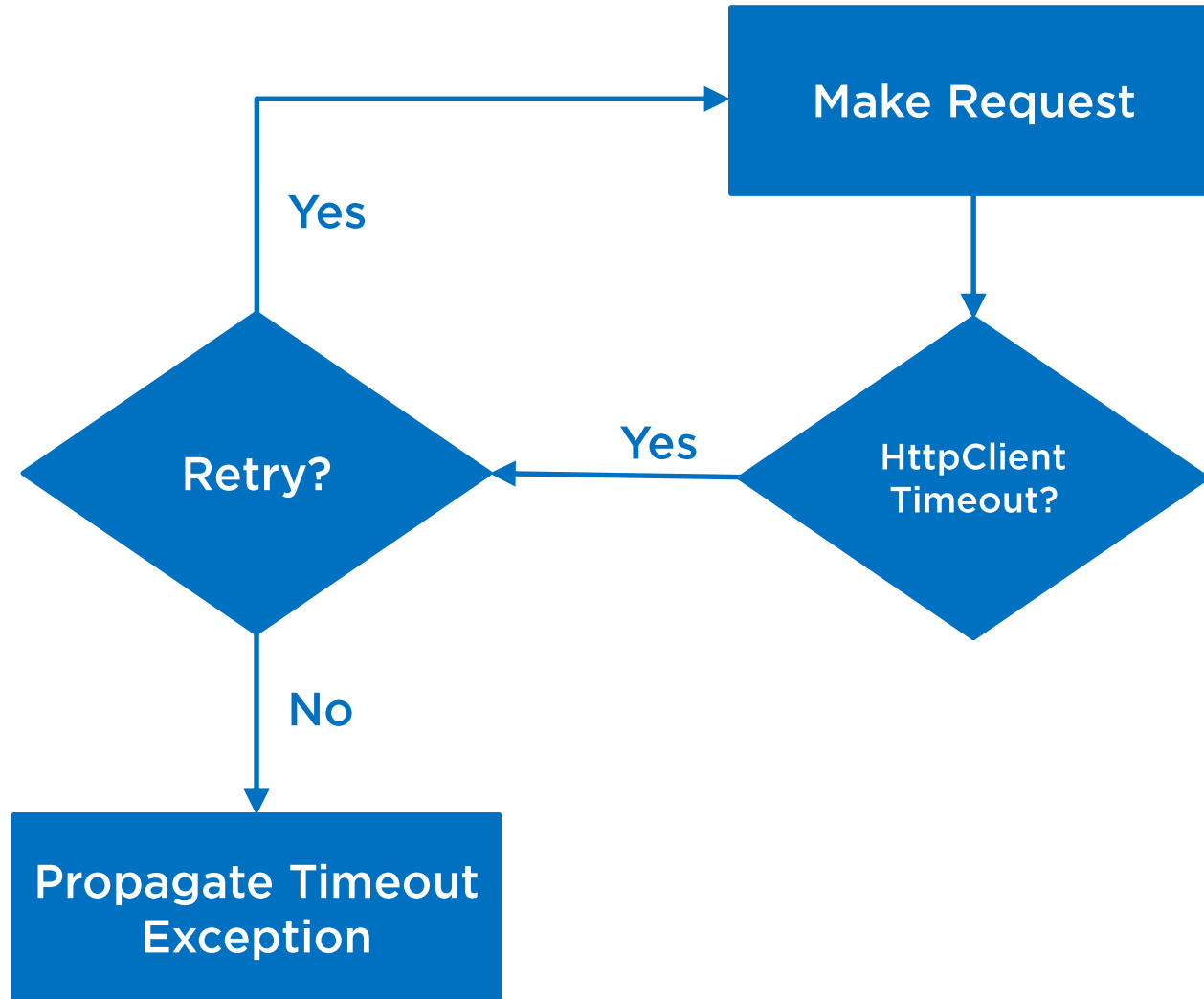
# Reauthorization



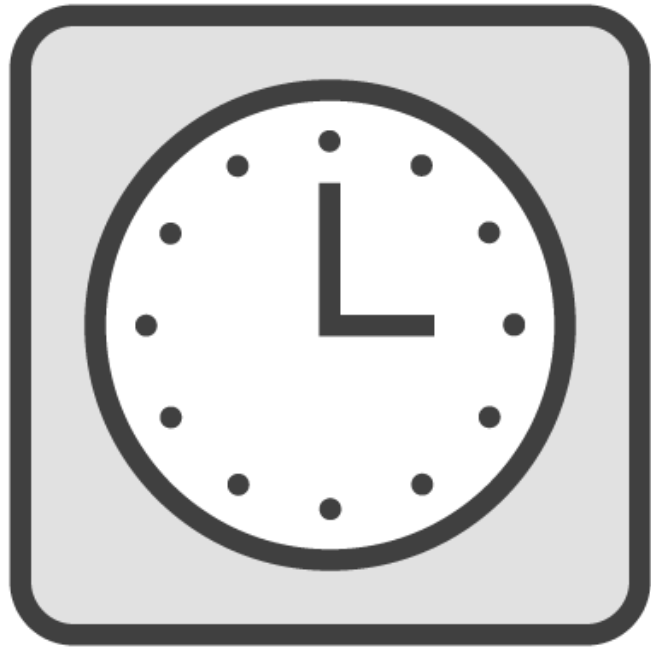
# Fallback



# HttpClient Timeout



# HttpClient Timeout





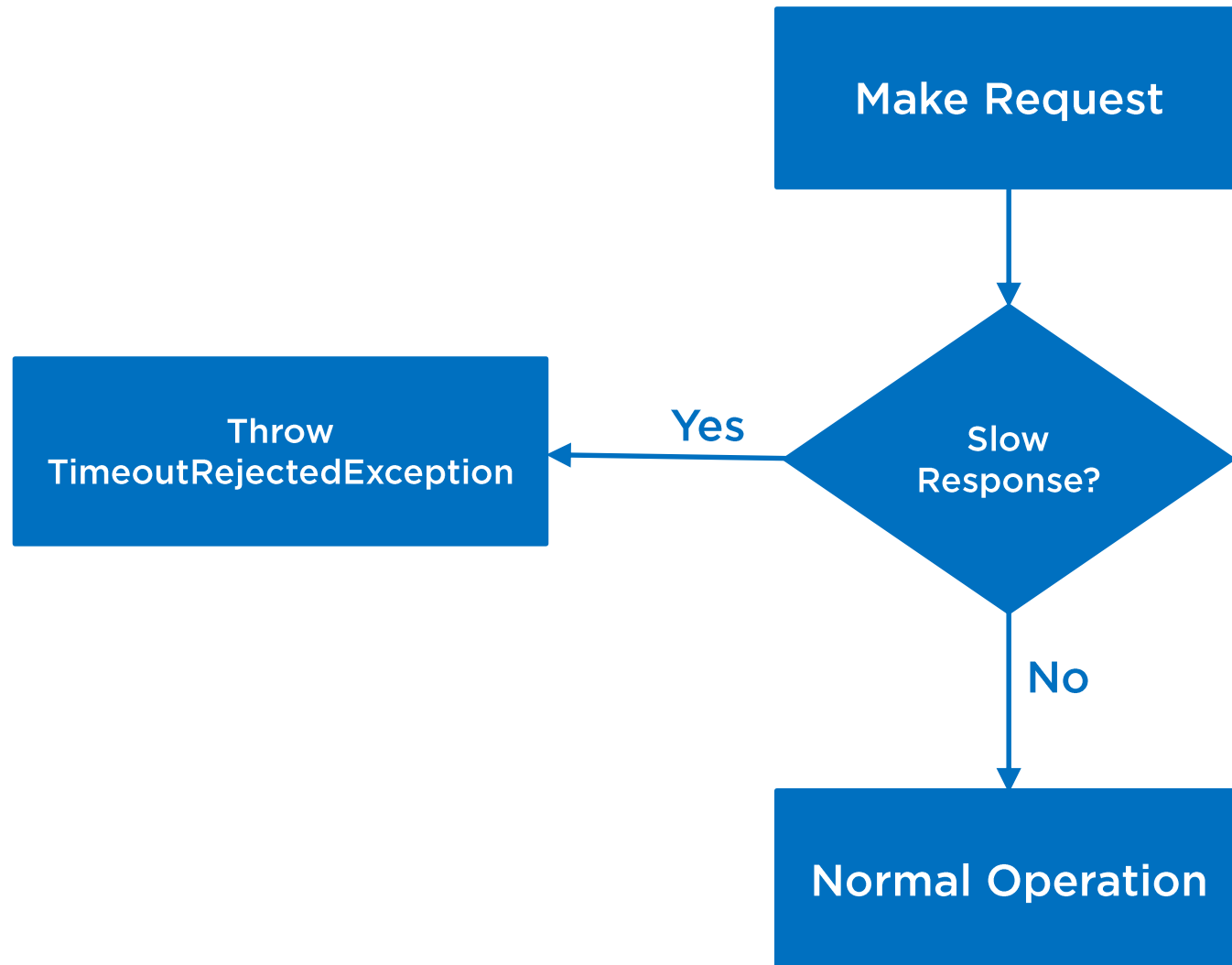
# Polly Timeout

HttpClient

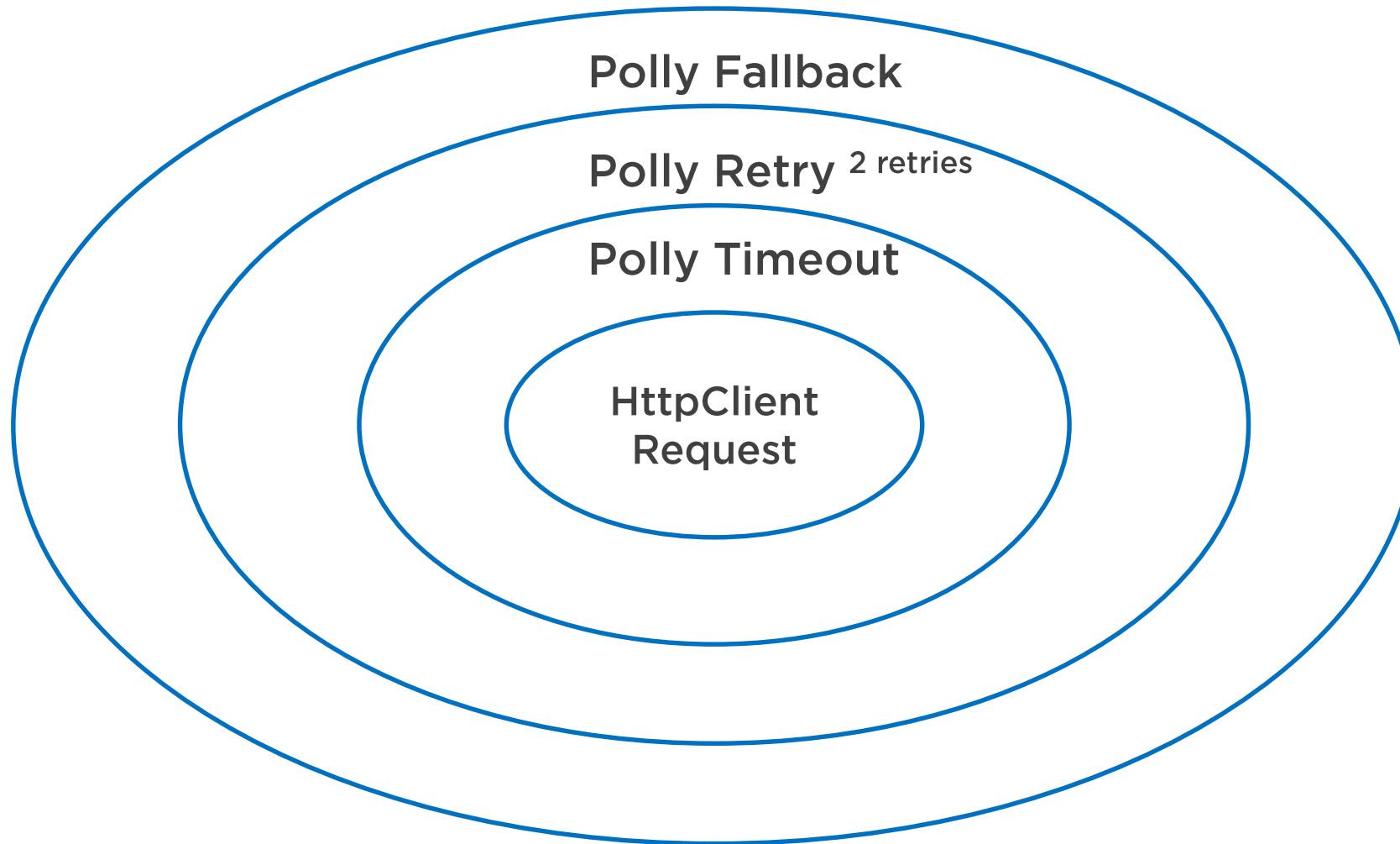
Preset  
Timeout  
100 Seconds



# Polly Timeout



# Polly Timeout with Retry and Fallback



# Polly Timeout with Retry and Fallback



# Summary



Why requests fail

Resilience strategies

Resilience framework

Policies – retry, wait & retry, fallback, timeout

Demos

