Building .NET Core Worker Services

Steve Gordon
MICROSOFT DEVELOPER TECHNOLOGIES MVP

@stevejgordon www.stevejgordon.co.uk

Overview

What are worker services?

Creating a new worker service project

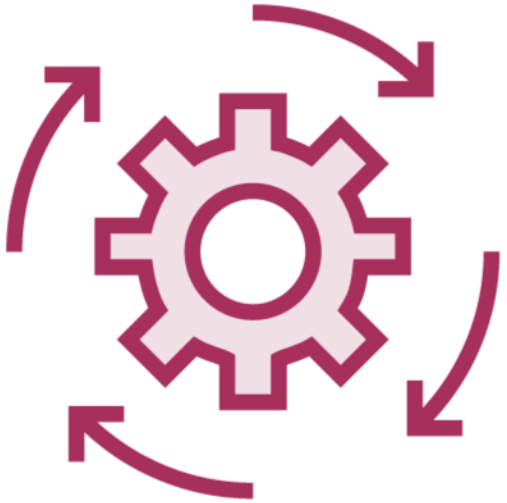
Migrate result processing to a worker service

- Poll a message queue
- Load and process the result file
- Refactoring the web application

What Are Worker Services?

Worker Services

Console application



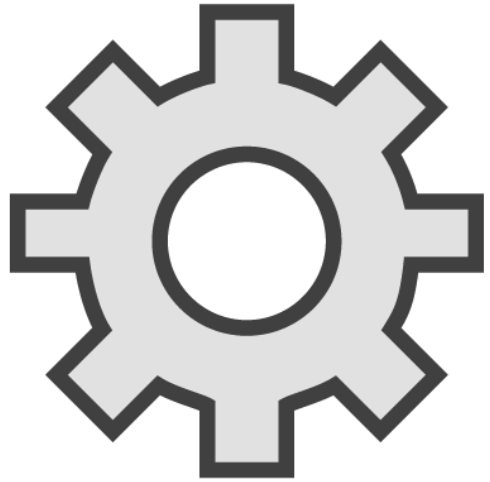
Hosting supports long-running operations

Scheduled workloads

.NET Core Hosting

Logging

Dependency Injection



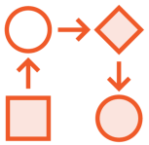
Configuration

Common Workloads

Processing messages/events from a queue, service bus or event stream

Reacting to file changes in a object/file store

Aggregating data from a data store



Enriching data in data ingestion pipelines

Formatting and cleansing of AI/ML datasets

Worker Service Architecture

Objectives

- Remove data processing from web app
- Break off responsibilities to microservices
- Design for cloud hosting in AWS

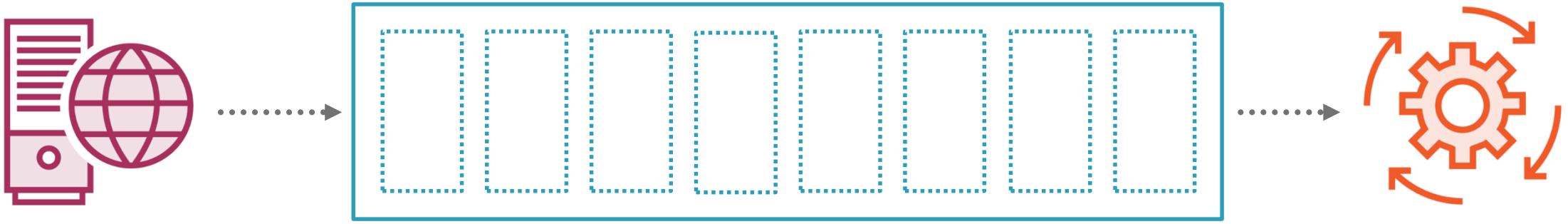
Amazon Web Services

Simple Storage Service

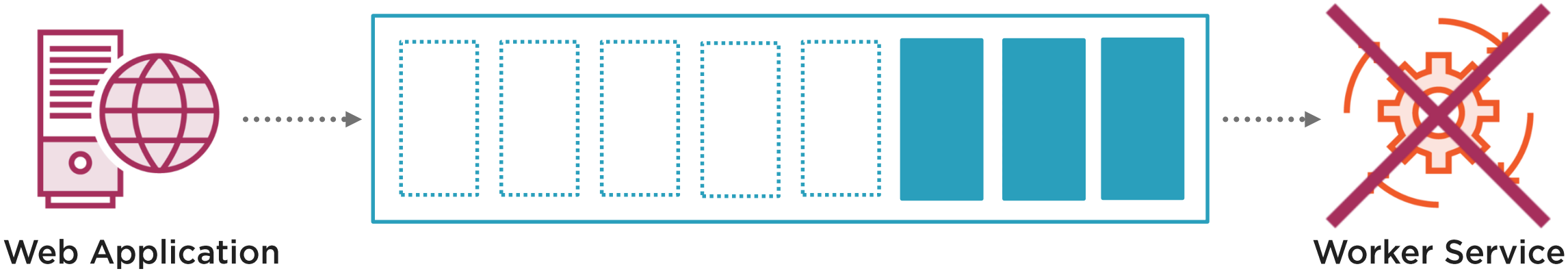
Simple Notification Service

Simple Queue Service

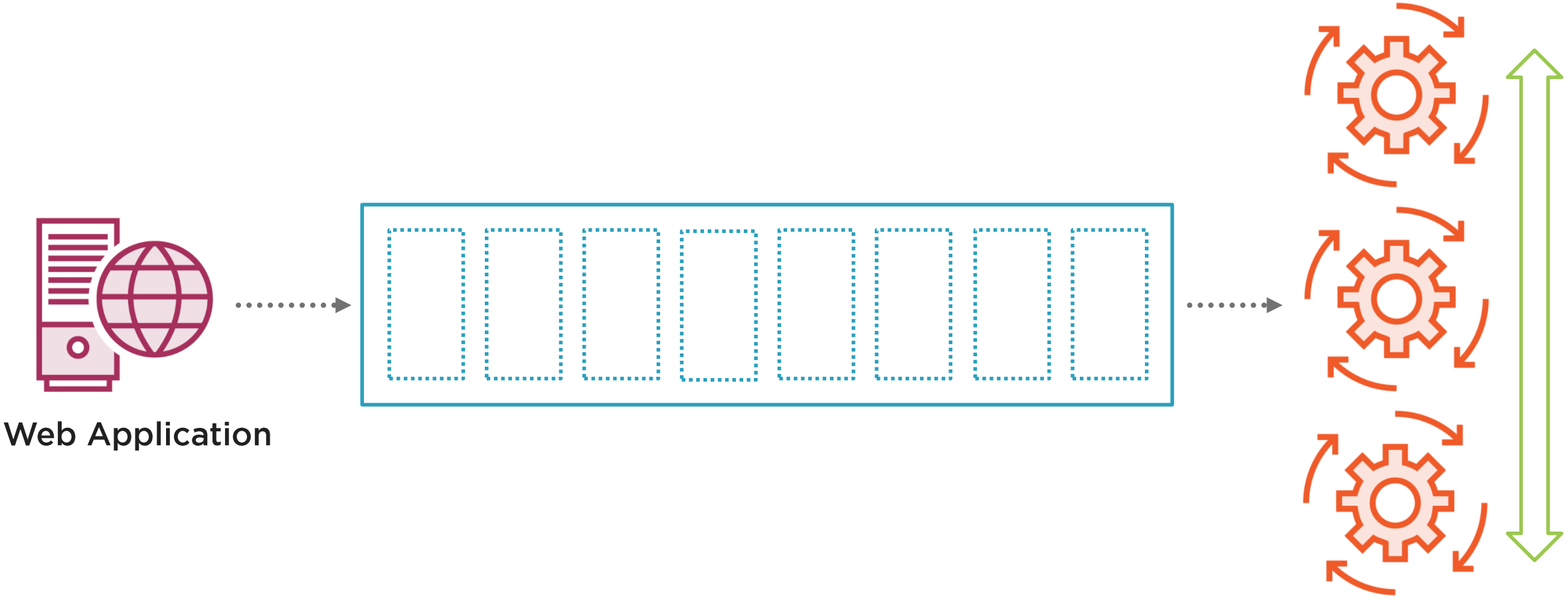
Decoupling Services With Queues



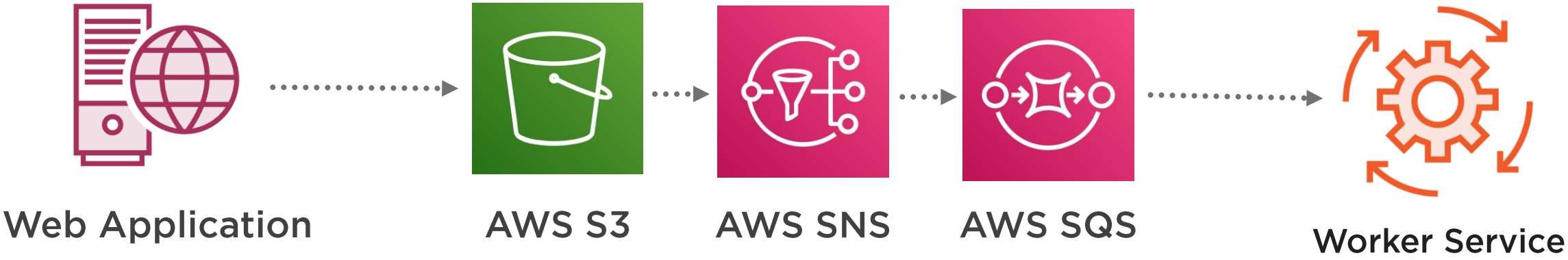
Decoupling Services With Queues



Decoupling Services With Queues



AWS Architecture



Demo

Create a worker service project

Explore the default project structure

Aside

```
dotnet new worker -n "TennisBookings.ScoreProcessor"
```

.NET CLI

Hosting in .NET Core

Host

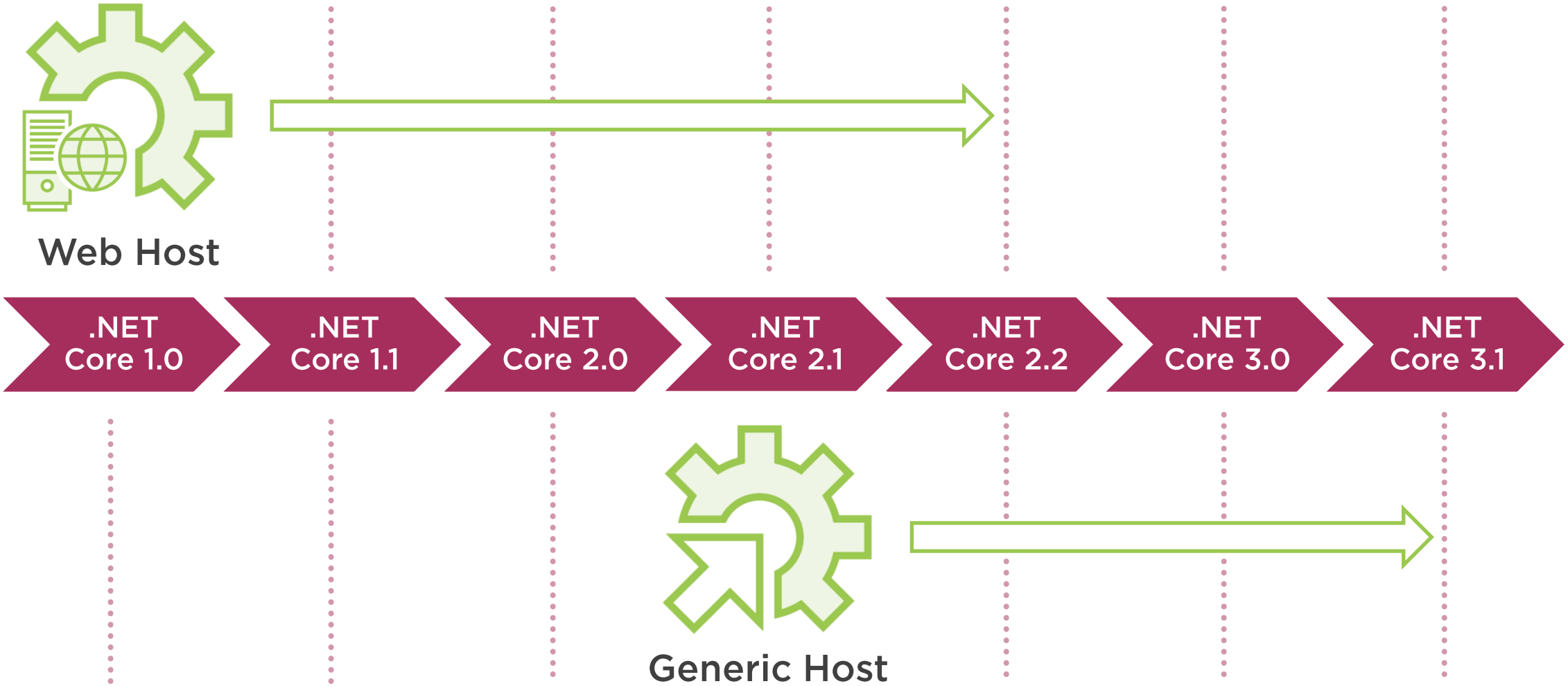
Manages application lifetime

Provides components such as logging, configuration and dependency injection

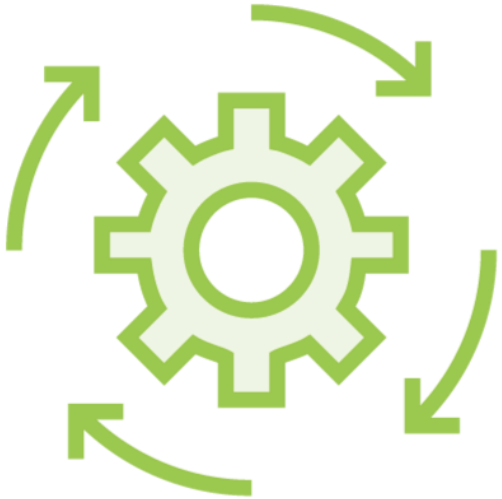
Turns a console application into a long-running service

Starts and stops hosted services

Hosting History



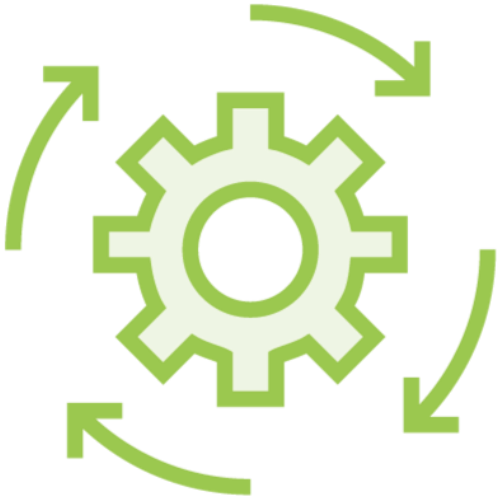
.NET Core 2.1



Generic Host
Worker Services

Web Host
ASP.NET Core Web Applications

.NET Core 3.0



Generic Host
Worker Services and
ASP.NET Core Web Applications

The Kestrel web server is started as a hosted service

WebHost

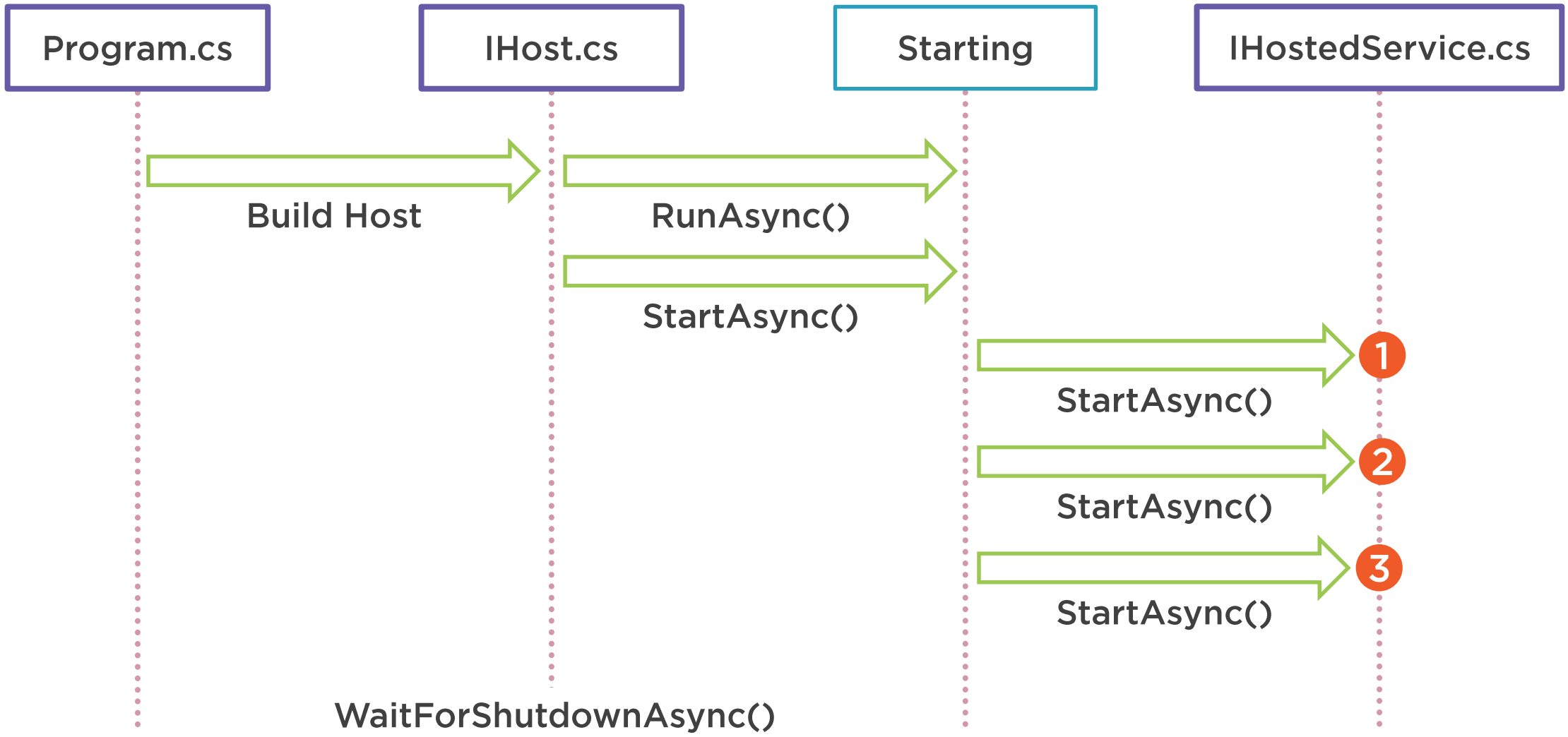
Should no longer be used for new web applications since .NET Core 3.0.

Program.cs

```
public class Program
{
    public static void Main(string[] args)
    {
        CreateHostBuilder(args).Build().Run();
    }

    public static IHostBuilder CreateHostBuilder(string[] args) =>
        Host.CreateDefaultBuilder(args)
            .ConfigureServices((hostContext, services) =>
            {
                services.AddHostedService<Worker>();
            });
}
```

Host Startup



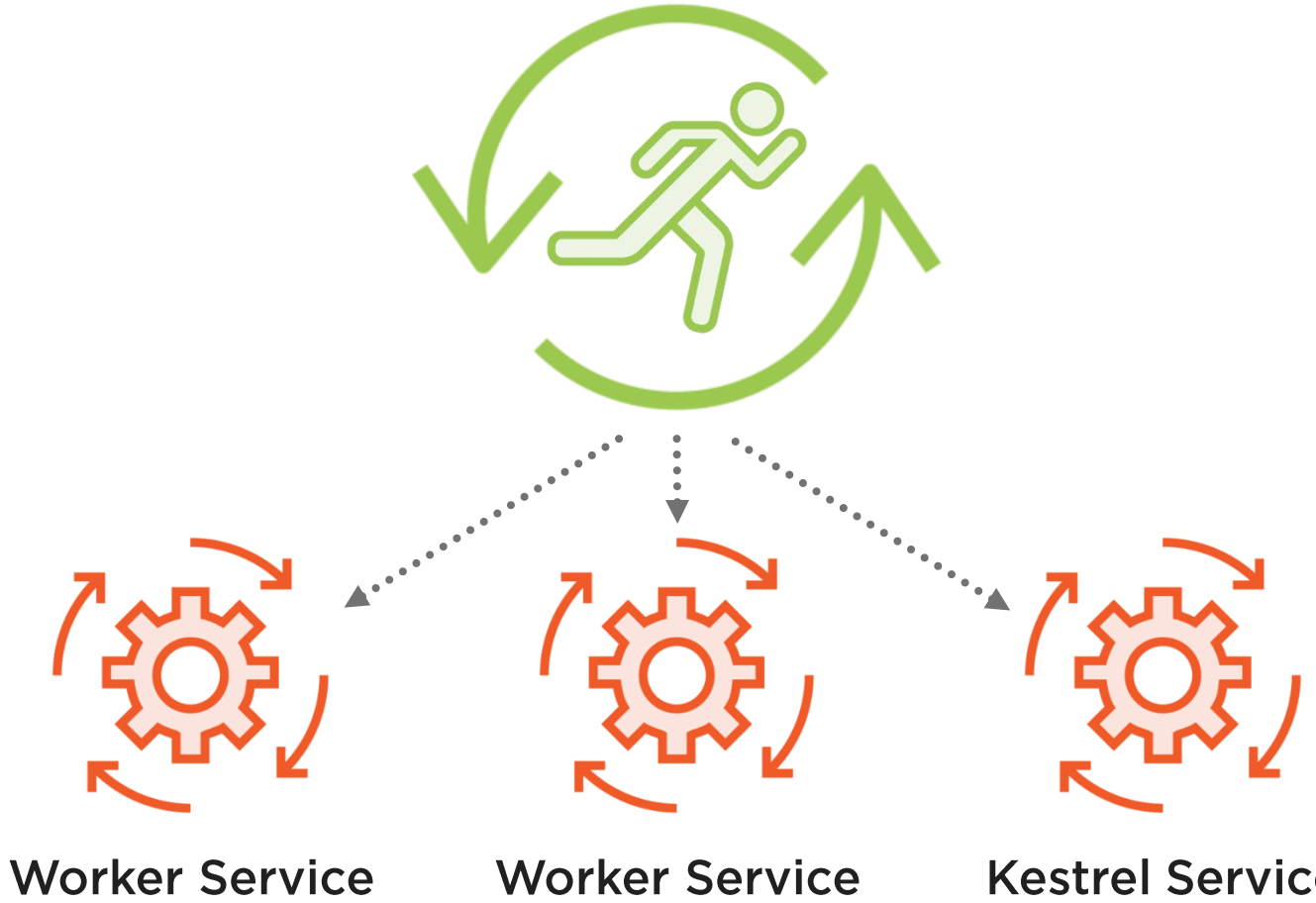
Triggering Shutdown

CTRL + C

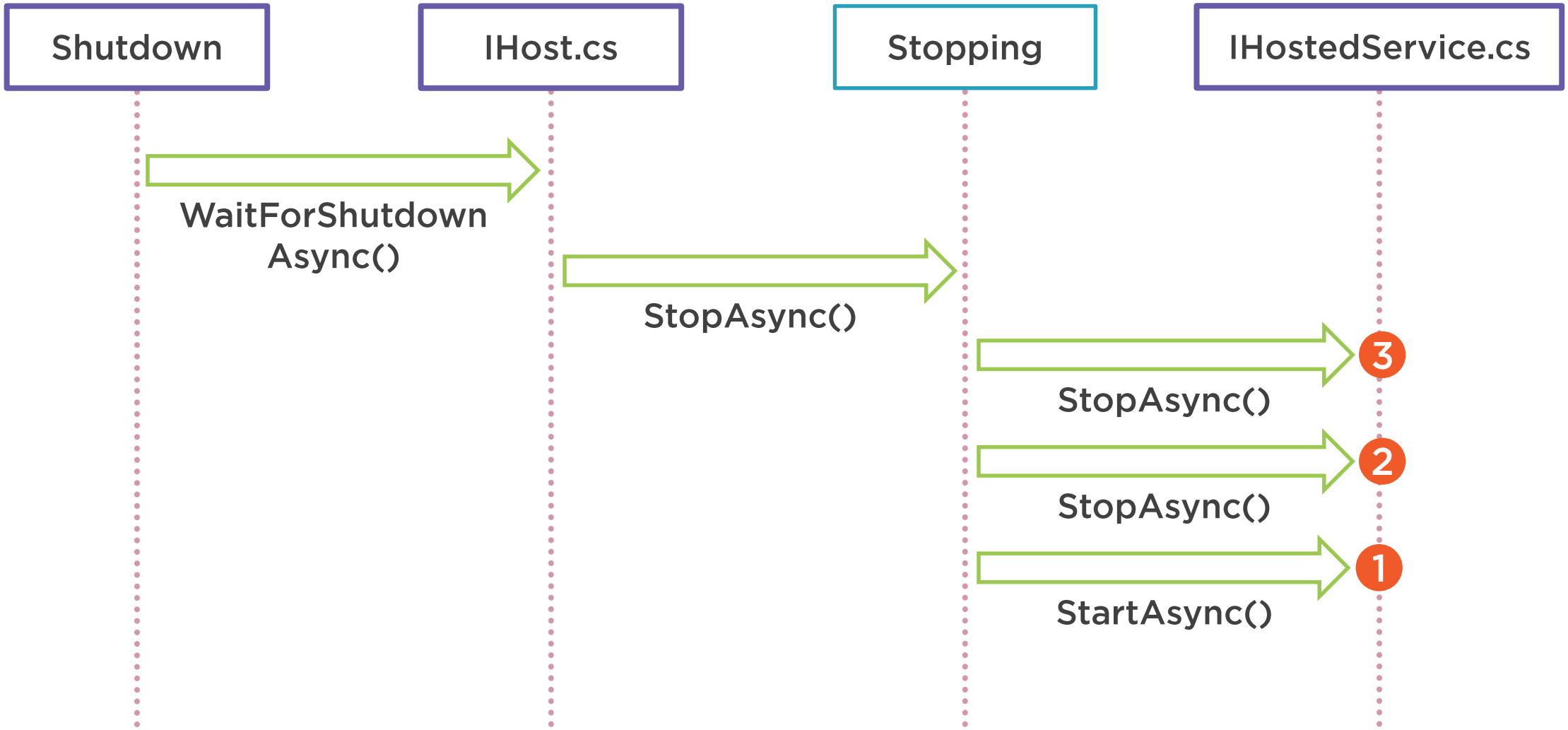
Process termination

Programmatic shutdown

Running Host



Host Shutdown



Demo

Set up Amazon Web Services

Set up LocalStack

Demo

Create the first background service

- Read messages from a queue
- Write S3 filename to a channel

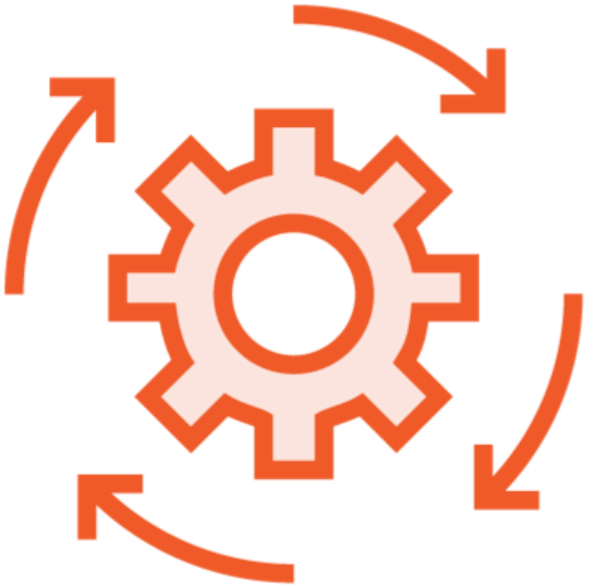
Demo

Create a second background service

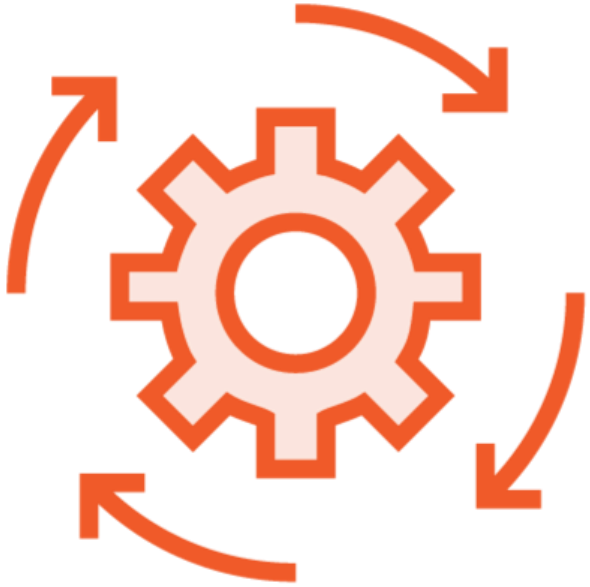
- Read filename from a channel
- Load the results file from S3
- Process the results

Concurrent Processing

QueueReadingService



ScoreProcessingService



Advanced Data and Stream Processing with Microsoft TPL Dataflow

by Szymon Warda

CPUs have more and more cores, but writing parallel programs is tricky. In this course, you will learn how the data flow programming model combined with the actor model makes writing high performance, large data-processing systems easy.

Resume Course | Bookmark | Add to Channel | Download Course

Expand All

Course author

Szymon Warda

With .NET from version 1.1 and with web development from the time when IE6 was the "better" browser. By day, a leader of an R&D department, by night a proud developer of cookit.pl - a pet project...

Course info

Level	Advanced
Rating	★★★★★ (28)
My rating	★★★★★
Duration	2h 54m
Released	15 Oct 2018

Share course

Building Concurrent Applications with the Actor Model in Akka.NET

by Jason Roberts

Easily build concurrent .NET applications using the high level abstractions of the Actor Model that automatically recovers from errors and that can be distributed across multiple computers with little additional effort.

Start Course | Bookmark | Add to Channel | Download Course

Table of contents | Description | Transcript | Exercise files | Discussion | Learning Check | Related Courses

Expand All

Introducing Actor Models and Akka.NET	38m 2s	

Course author

Jason Roberts

With over 15 years of experience in both frontend and backend software development, Jason Roberts is a freelance developer, trainer, and author. He holds a Bachelor of Science degree in computing,...

Course info

Level	Beginner
Rating	★★★★½ (364)
My rating	★★★★★
Duration	3h 23m
Released	5 Aug 2015

Share course

Demo

Refactoring the web application

Summary

Worker service template

Built a worker service

- Replaced web application processing
- Created a .NET Core microservice

Host provides features such as dependency injection, logging and configuration

Up Next:
Advanced Hosted Service Concepts