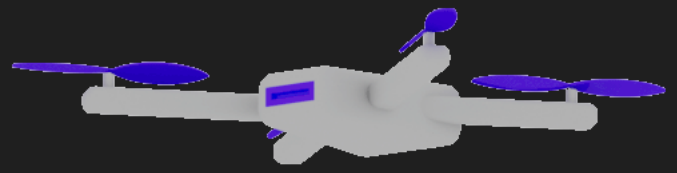


.NET Conf 2023 x Seoul

.NET Framework 기반 SaaS 솔루션 .NET 6 전환기

한재준 | 크레비스파트너스

오늘 이야기할 내용

- 도너스 소개
- .NET 6 업그레이드 준비
- .NET 6 업그레이드
- .NET 6 업그레이드 후

도너스 소개

크레비스파트너스

CREVISSE

“Awaken Your Potential
to Impact the World”

세상을 바꾸어 나가는
소셜 임팩트 벤처 그룹

브릭투웍스

BRICTOWORKS

크레비스의 기술제품 사업 본부

비즈니스 솔루션(Brick)을 만들어서
고객의 더 큰 성과(to Works)를 달성

도너스

DONUS

모금 관리 SaaS 솔루션

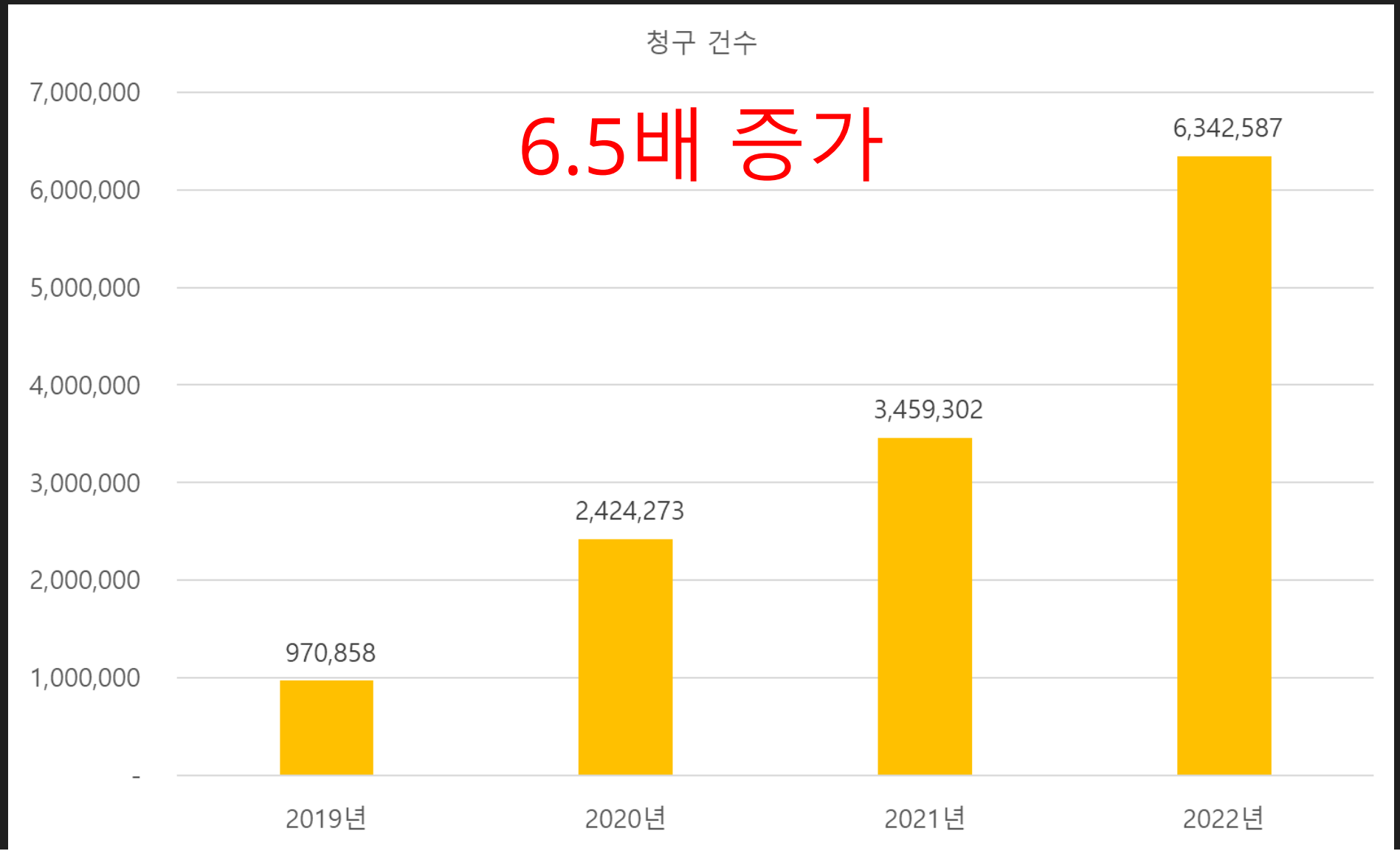
후원자 총 400만 명
연간 기부금 6,000억 원

도너스

분야별 주요 고객

- 국제구호: 국경없는의사회, 옥스팜코리아
- 아동/복지: 굿네이버스, 플랜코리아
- 인권/여성: 국제앰네스티, 한국여성의전화
- 환경: WWF 세계자연기금, 환경재단
- 대학: 서울대발전기금, 가톨릭대발전기금
- 의료/보건: 서울대병원, 연세의료원, 서울성모병원
- 문화/예술: 한국문화예술위원회, 세종문화회관
- 시민단체: 참여연대, 서울환경연합, 녹색연합
- 지역사회: 한국YWCA연합회, 아름다운가게
- 동물: 동물자유연대, 동물해방물결
- 종교: 대한성서공회

빠르게 성장하는 도너스


청구 건수
6.5배 증가
7,000,000
6,000,000
5,000,000
4,000,000
3,000,000
2,000,000
1,000,000
-
970,858
2,424,273
3,459,302
6,342,587
2019년
2020년
2021년
2022년

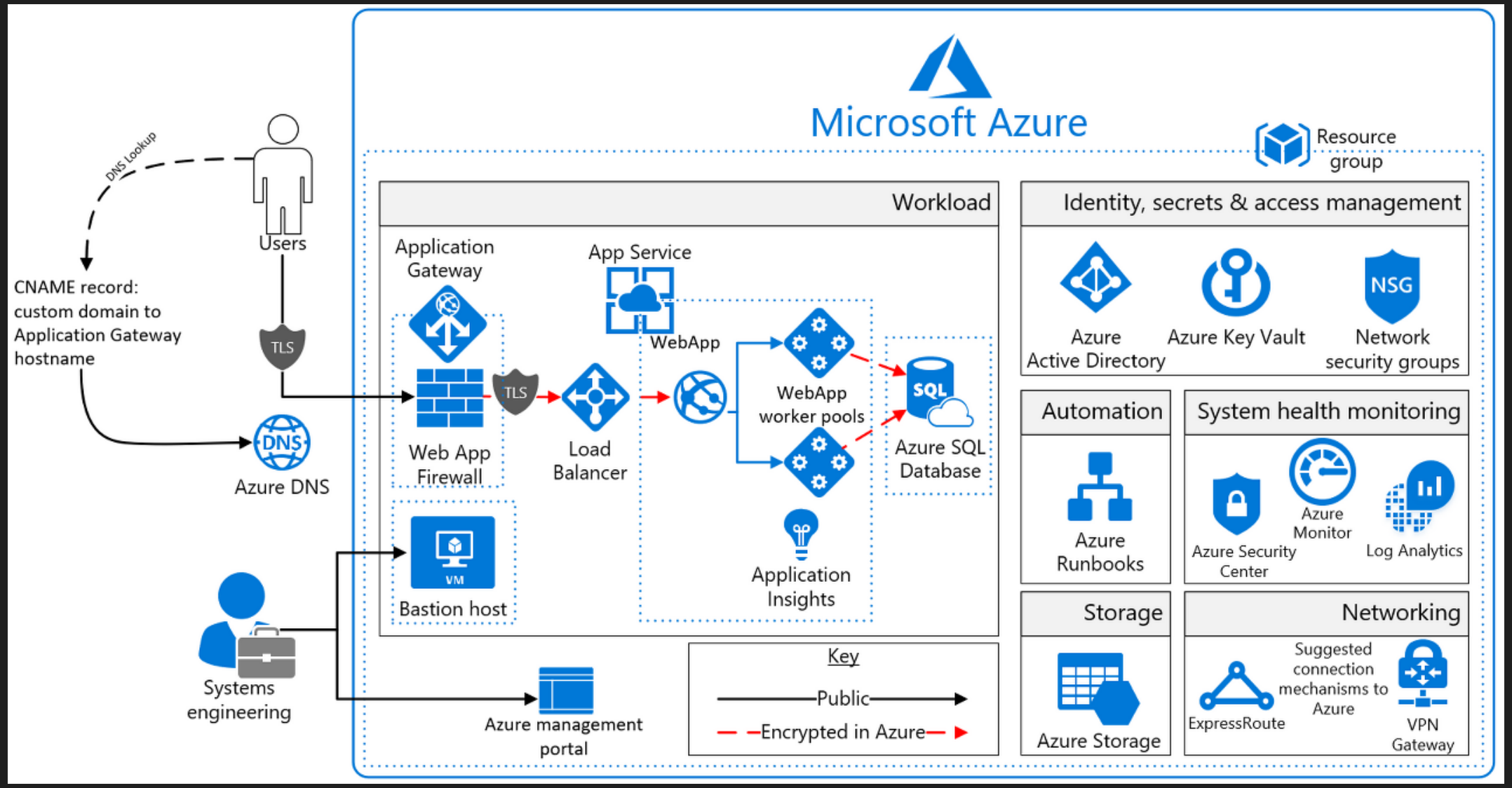
빠르게 성장하는 도너스


메시지 발송건수
13배 증가
30,000,000
25,000,000
20,000,000
15,000,000
10,000,000
5,000,000
-
2,132,068
15,146,143
22,887,819
27,903,712
2019년
2020년
2021년
2022년

도너스

- Multi Tenant SaaS
- .NET framework 4.7.2
- 관련 제품 중 80% 이상 .NET 사용
- Azure Cloud

Azure web application best practice: DONUS



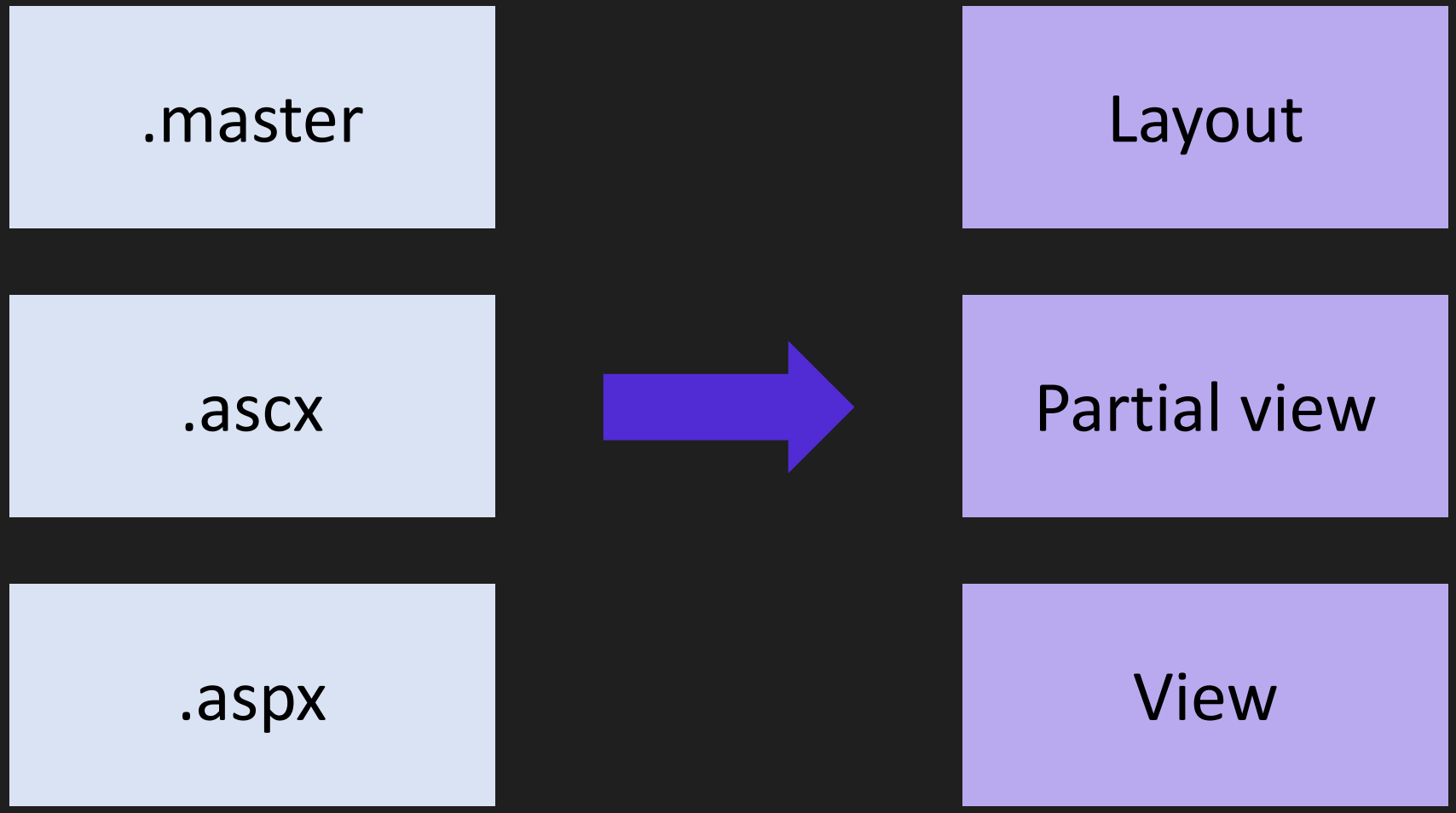
.NET 6 업그레이드 타임라인

		1월	2월	3월	4월	5월	6월	7월	8월	9월	10월	11월	12월
업그레이드 준비	Web Forms ⟹ MVC 전환						프로젝트						
	업그레이드 리서치 & 이슈 설계												
	기타 업그레이드 준비 작업												
업그레이드	업그레이드 리서치 & 이슈 설계												
	업그레이드, CI/CD 구축												
	테스트 케이스 작성 & 통합 테스트										배포		
업그레이드 후	DI 적용												
	임시로직 제거												

400 페이지 규모의 솔루션
개발자 3명, 7개월 진행

.NET 6 업그레이드 준비

Web Forms 페이지 MVC + Razor 전환



Web Forms **페이지** MVC + Razor **전환**

- Telerik razor-converter ⟹ 실패
  (https://github.com/telerik/razor-converter)
- 정규식 활용
- 400개의 Web Forms 페이지를 cshtml으로 전환

`@` Replace

- `@` ⇒ `@Html.Raw("@")`
- `"@"` ⇒ `"@Html.Raw("@")"` 변경

if, for, foreach 변경 => <% if %> => @if
(`<%\s` 로 찾으면 편함)

Replace 진행

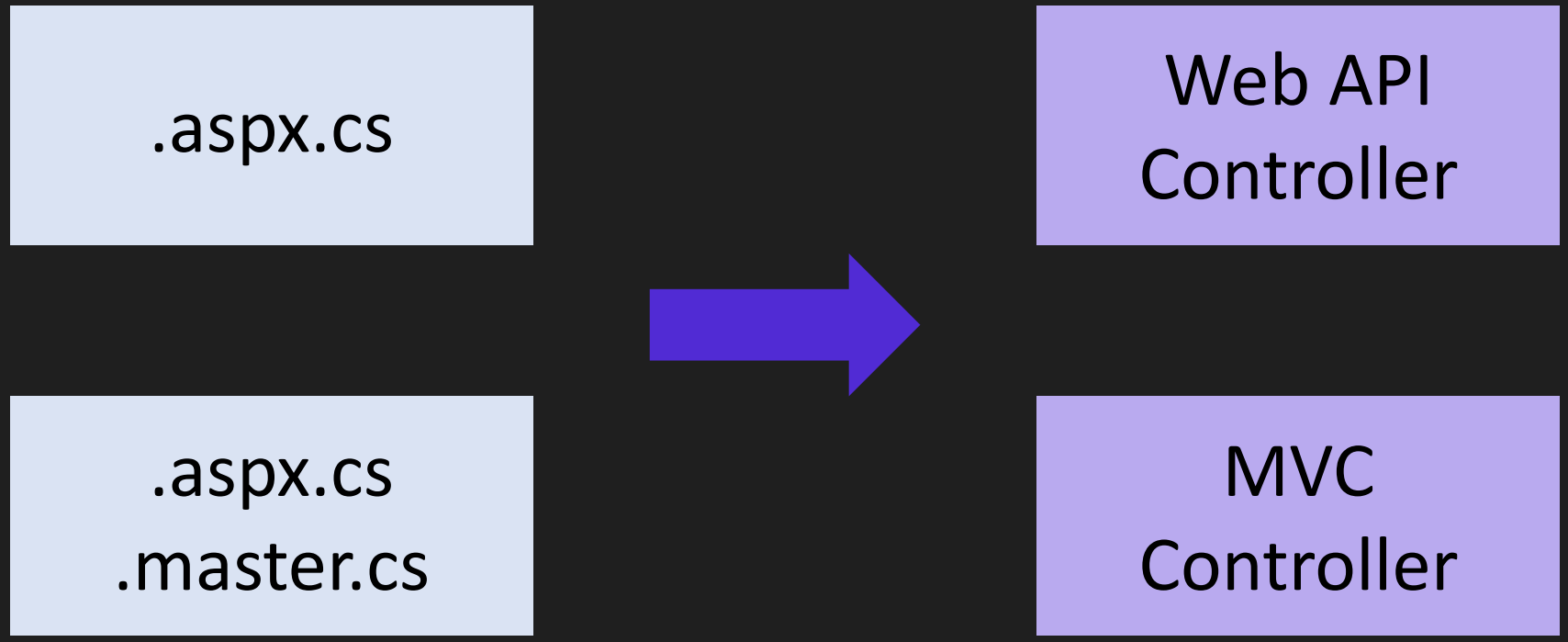
- `<%:\s*` => `@(`
- `<%=\s*` => `@Html.Raw(`
- `\s*%>` => `)`

aspx 파일 변환 시 주의사항

- `map[" xxxIdx"]` , `xxxIdx == " 1"` 처럼 공백이 들어간 경우가 발생할 수 있음.
  (`(\"\s\w*\"|\n\s*\")` 검색으로 쌍따옴표로 공백이 들어간 경우를 찾아서 확인 필요)
- ~~"앞에서 개행되는 경우 있음 `(\n *\")` 으로 검색~~ ⇒ 위 정규식으로 통합
- aspx 제거와 cshtml 생성이 하나의 commit 내에 이루어져야 rename으로 인식됨(gitkraken).
- `@if` 안에서 닫지않는 태그, js 등 syntax error는 `@:` 활용하기

```
@for (int index = 0; index < Model.MemberCustom.Count; index++)
{
    Map customField = (Map)Model.MemberCustom[index];

    if (index % totalCol == 0)
    {
        @:<tr>
    }
```

Web Forms code behind MVC 전환



기타 업그레이드 준비 작업

- Session에 저장하는 데이터 형식 json으로 변경

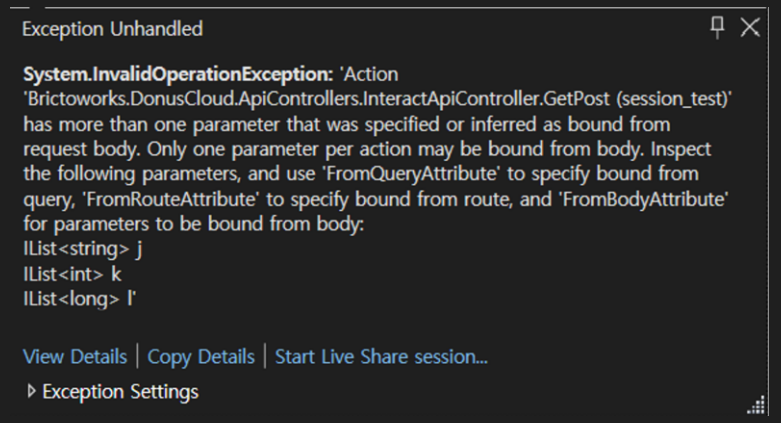
```
public static class SessionExtensions
{
    public static void Set<T>(this ISession session, string key, T value)
    {
        session.SetString(key, JsonSerializer.Serialize(value));
    }

    public static T? Get<T>(this ISession session, string key)
    {
        var value = session.GetString(key);
        return value == null ? default : JsonSerializer.Deserialize<T>(value);
    }
}
```

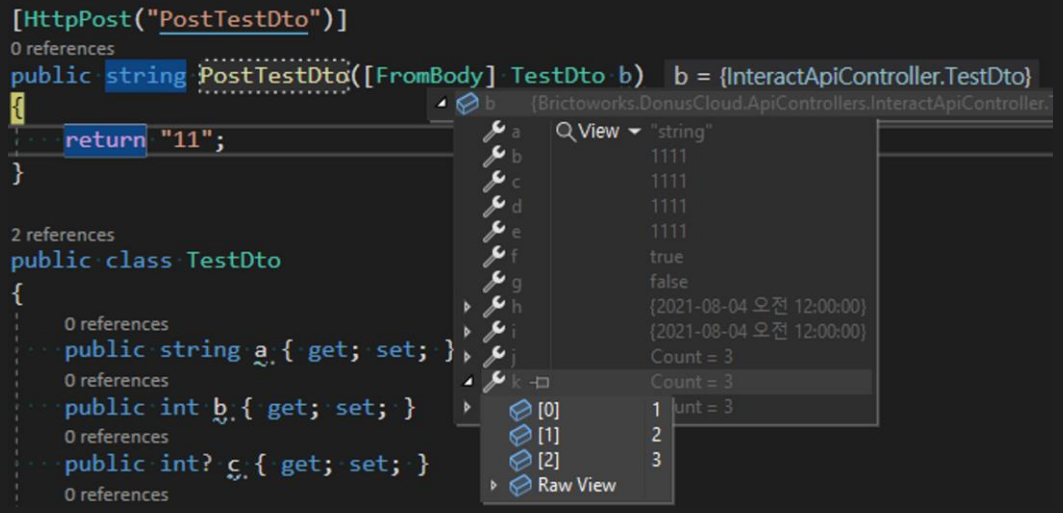
All session data must be serialized to enable a distributed cache scenario, even when using the in-memory cache. String and integer serializers are provided by the extension methods of ISession. Complex types must be serialized by the user using another mechanism, such as JSON.

https://learn.microsoft.com/en-us/aspnet/core/fundamentals/app-state?view=aspnetcore-6.0&source=docs#set-and-get-session-values

기타 업그레이드 준비 작업

- Web API Controller DTO(Data Transfer Object) 적용





Don't apply `[FromBody]` to more than one parameter per action method. Once the request stream is read by an input formatter, it's no longer available to be read again for binding other `[FromBody]` parameters.

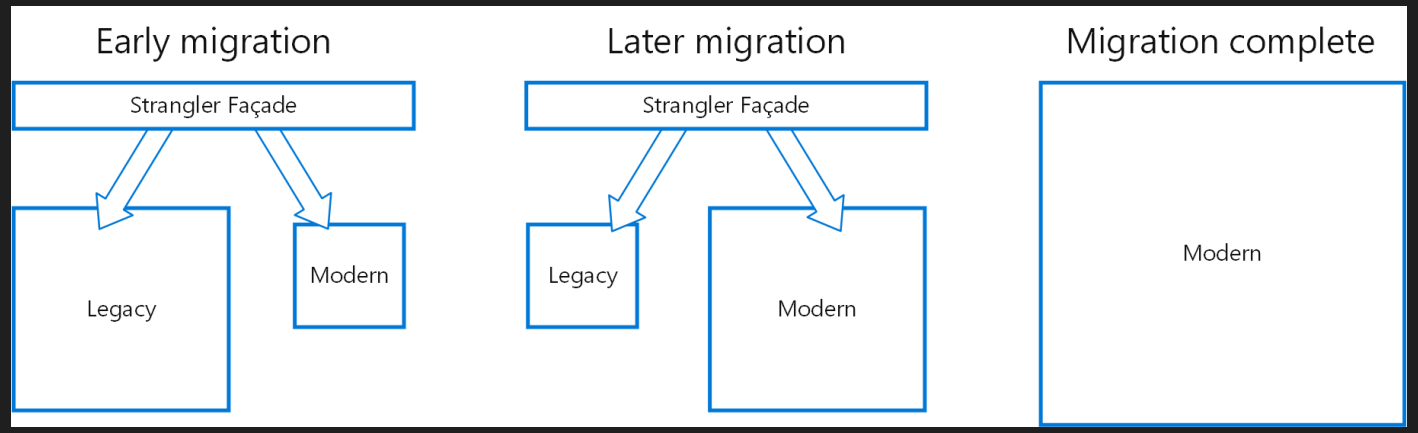
https://learn.microsoft.com/en-us/aspnet/core/mvc/models/model-binding?view=aspnetcore-6.0#frombody-attribute-1

.NET 6 업그레이드 방법 리서치

- Migration Tool
  - upgrade-assistant (https://github.com/dotnet/upgrade-assistant)
  - try-convert (https://github.com/dotnet/try-convert)

⟹ migration 실패

- MS Documentation(Porting Existing ASP.NET Apps to .NET 6)



Strangler pattern

⟹ 실패 (인프라 구축이 어렵고 history 유지가 어려움)

https://learn.microsoft.com/en-us/dotnet/architecture/porting-existing-aspnet-apps/

결론

그냥 하자.

.NET 6 업그레이드

Target Framework 변경

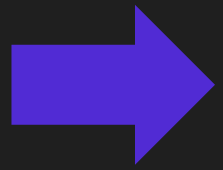
- csproj 파일 변경

```
<Project Sdk="Microsoft.NET.Sdk.Web">

    <PropertyGroup>
        <TargetFramework>net6.0</TargetFramework>
        <Nullable>enable</Nullable>
        <ImplicitUsings>enable</ImplicitUsings>
    </PropertyGroup>
```

- Web.config 파일 ⟹ appsettings.json 파일 변경 (upgrade-assistant 이용)
- Program.cs 파일 생성

.NET framework

- Global.asax.cs
  - Application Start



.NET 6

- Program.cs

수많은 Build Error 발생

.NET 6 업그레이드 이슈

~~DC-3006~~ [Archi] HttpRequestBase => HttpRequest 변동사항 적용	~~DC-3031~~ [Archi] RestSharp 관련 수정	~~DC-3047~~ [Archi] cshtml 관련 Web.Config 정리 RELEASED
~~DC-3017~~ [Archi] Target Framework .NET 6로 변경	~~DC-3032~~ [Archi] HttpPostedFileBase => IFormFiled로 변경	~~DC-3051~~ [Archi] LoginManager prototype 코드 제거에 따른 에러 수정 RELEASED
~~DC-3018~~ [Archi] 기존 ApiController 를 AspNetCore ApiController 로 변경	~~DC-3033~~ [Archi] WebApiConfig 정리	~~DC-3052~~ [Archi] Test 프로젝트 target .NET 6로 변경 RELEASED
~~DC-3019~~ [Archi] GlobalUsings.cs 추가	~~DC-3034~~ [Archi] Dlog, ApiTool, LogAttributes 정리	~~DC-3053~~ [Archi] Session.SetString 시 null로 set 하는 경우 에러 발생 RELEASED
~~DC-3020~~ [Archi] Helper class 적용	~~DC-3035~~ [Archi] PrepareApiLog MiddleWare 추가	~~DC-3054~~ [Archi] X-Donus-TenantCode 확인 못하는 문제 수정 RELEASED
~~DC-3021~~ [Archi] Bundling 라이브러리 적용	~~DC-3036~~ [Archi] MvcControllerBase 정리	~~DC-3055~~ [Archi] HostingEnvironmentHelper.MapPath 로직 변경 RELEASED
~~DC-3023~~ [Archi] ApiController NetCore Route 적용	~~DC-3037~~ [Archi] Server.ScriptTimeout, Server.Transfer 제거	~~DC-3056~~ [Archi] ajax_helper.js 수정, AddWebEncoders 추가, basic.css 수정 RELEASED
~~DC-3024~~ [Archi] Stripts, static 폴더 wwwroot로 이동	~~DC-3038~~ [Archi] namespace 정리, WebApiConfig 참조하는 미사용 파일 주석	~~DC-3057~~ [Archi] Error Handler - MVC Controller, Global RELEASED
~~DC-3026~~ [Archi] 기존 MvcController를 AspNetCoreMvcController로 전환	~~DC-3039~~ [Archi] CloudStorageAccount => BlobContainerClient 변경	~~DC-3058~~ [Archi] Error Handler - API Controller RELEASED
~~DC-3027~~ [Archi] Controller Attribute 수정	~~DC-3040~~ [Archi] Response.IsClientConnected, Response.End() 변경	~~DC-3059~~ [Archi] WebApiConfig migration 리서치 EO RELEASED
~~DC-3028~~ [Archi] ActionResult Return => NetCoreMvc Return 으로 변경	~~DC-3041~~ [Archi] AcceptVerbs attribute 수정	~~DC-3060~~ [Archi] Timeout 관련 구현 리서치 RELEASED
~~DC-3029~~ [Archi] HttpSessionState => ISession으로 변경	~~DC-3044~~ [Archi] ApiTools HttpRequestMessage => HttpRequest 변경 RELEASED	
~~DC-3030~~ [Archi] 기타 Attribute 관련 에러 수정(OutputCache, AsyncTimeout, RoutePr...	~~DC-3045~~ [Archi] Response.Cookies 사용법 변경 RELEASED	

HttpRequestBase ⟹ HttpRequest

- Request 관련 수정
  - `Request.Properties` ⇒ `Request.HttpContext.Items`
  - `Request.Headers.Contains` ⇒ `Request.Headers.Keys.Contains`
  - `Request.Headers.GetValues("s")` ⇒ `Request.Headers["s"]` or `Request.Headers.TryGetValue("s")`
  - `request.GetQueryNameValuePairs()` ⇒ `request.Query`
  - `Request.QueryString` ⇒ `Request.Query`
  - `request.Files.Get("zipfile")` ⇒ `Request.Form.Files.GetFile("zipfile")`
  - `request.Content.ReadAsStringAsync().Result` ⇒

```
using StreamReader reader = new(request.Body);
string json = await reader.ReadToEndAsync();
```

  - IFormFile 예시

```
using (BinaryReader binaryReader = new(zipFile.InputStream))
{
    zipFileBytes = binaryReader.ReadBytes(zipFile.ContentLength);
}
```

⇒

```
using (MemoryStream stream = new())
{
    zipFile.CopyTo(stream);
    zipFileBytes = stream.ToArray();
}
```

Helper static class 적용

Program.cs

```
HttpContextHelper.Initialize(app.Services.GetRequiredService<IHttpContextAccessor>());
```

HttpContextHelper.cs

```
22 usages  Jaejoon Han
public static class HttpContextHelper
{
    private static IHttpContextAccessor _httpContextAccessor;

    ◇  2 usages  Jaejoon Han
    public static void Initialize(IHttpContextAccessor httpContextAccessor)
    {
        _httpContextAccessor = httpContextAccessor;
    }

    20 usages  Jaejoon Han
    public static HttpContext Current => _httpContextAccessor.HttpContext;
}
```

Helper static class 적용

.NET framework

- HttpContext
  - Current
- ConfigurationManager
  - AppSettings
- HostingEnvironment
  - MapPath
  - QueueBackgroundWorkItem()

.NET 6

- HttpContextHelper
  - Current
- ConfigurationHelper
  - AppSettings
- HostingEnvironmentHelper
  - MapPath
  - QueueBackgroundWorkItem()

Custom Middleware 개발

.NET framework

- WebApiConfig.cs
  - IExceptionLogger
  - IExceptionHandler
  - ApiExceptionFilterAttribute

- Global.asax.cs
  - Application Error

.NET 6

- API Logging Middleware
- API Exception Handling Middleware

- MVC Exception Handling Middleware
- Not Found Handling Middleware

Background Task queue

.NET framework

- HostingEnvironment
  - QueueBackgroundWorkItem()

.NET 6

- IBackgroundTaskQueue
- BackgroundService

```
usages  overrides  Jaejoon Han  ext methods  exposing APIs
private async Task BackgroundProcessing(CancellationToken stoppingToken)
{
    while (!stoppingToken.IsCancellationRequested)
    {
        Func<CancellationToken, ValueTask> workItem = await TaskQueue.DequeueAsync(stoppingToken);

        try
        {
            await workItem(stoppingToken);
        }
        catch (Exception ex)
        {
            DLog.LogException(ex, $"Error occurred executing {nameof(workItem)}.", level: LogLevel.Error);
        }
    }
}
```

https://learn.microsoft.com/en-us/aspnet/core/fundamentals/host/hosted-services?view=aspnetcore-7.0&tabs=visual-studio#queued-background-tasks

Background Task queue

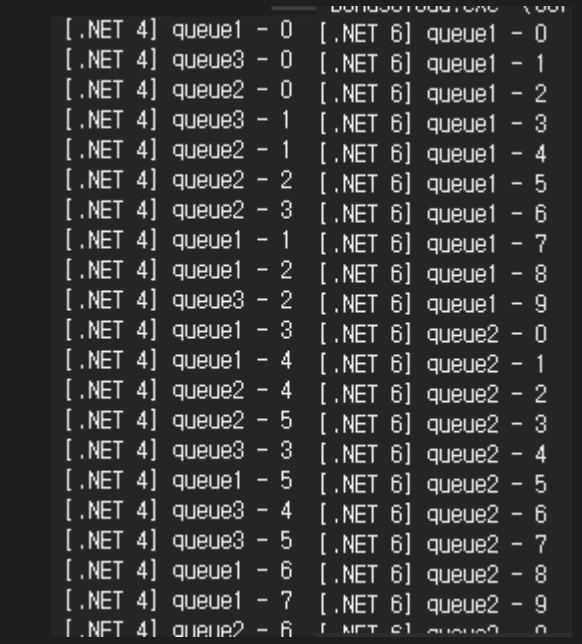
[주의사항]

.NET framework

```
HostingEnvironment.QueueBackgroundWorkItem(ct ⟹ Test(ct, 1));
HostingEnvironment.QueueBackgroundWorkItem(ct ⟹ Test(ct, 2));
HostingEnvironment.QueueBackgroundWorkItem(ct ⟹ Test(ct, 3));
```

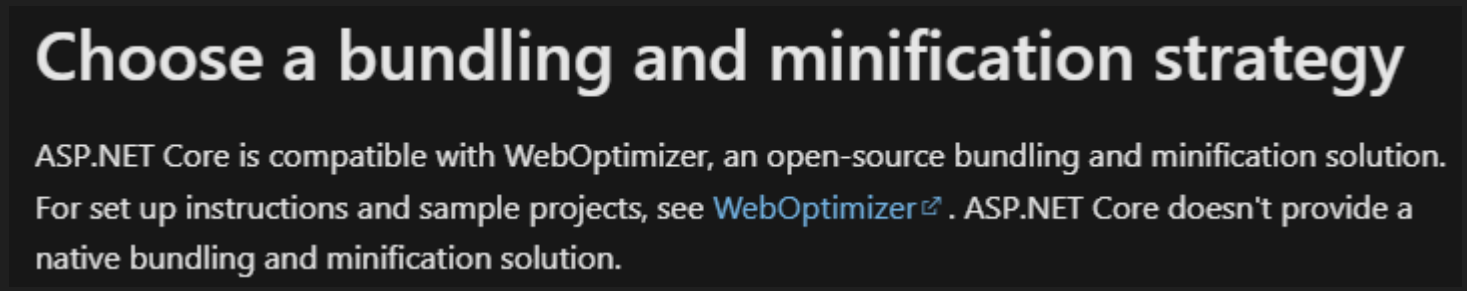
.NET 6

```
_taskQueue.QueueBackgroundWorkItemAsync(async ct ⟹ Test(ct, 1));
_taskQueue.QueueBackgroundWorkItemAsync(async ct ⟹ Test(ct, 2));
_taskQueue.QueueBackgroundWorkItemAsync(async ct ⟹ Test(ct, 3));
```



WebOptimizer 적용

- BundleConfig.cs 제거



- WebOptimizer middleware 추가

```
services.AddWebOptimizer(
    pipeline =>
    {
        pipeline.AddBundle("/bundles/basic.js", "text/javascript", basicJsContents).Concatenate();
        pipeline.AddBundle("/bundles/donus.js", "text/javascript", donusJsContents).Concatenate();
        pipeline.AddBundle("/bundles/basic.css", "text/css", basicCssContents).Concatenate();
        pipeline.AddBundle("/bundles/bootstrap.js", "text/javascript", bootstrapJsContents).Concatenate();
        pipeline.AddBundle("/bundles/bootstrap.css", "text/css", bootstrapCssContents).Concatenate();
        pipeline.AddBundle("/bundles/cti.js", "text/javascript", ctiJsContents).Concatenate();
        pipeline.AddBundle("/bundles/cti.css", "text/css", ctiCssContents).Concatenate();
        pipeline.AddBundle("/bundles/statistics.js", "text/javascript", statisticsContents).Concatenate();
        pipeline.AddBundle("/bundles/link-config.css", "text/css", linkCssContents).Concatenate();
    }
);
```

.NET 6 CI/CD 구축

.NET 6

- IIS에서 프로세스를 사용 중일 때 배포 불가능

해결방안

- App Offline File: 배포 중에는 잠시 빈 페이지로 이동

⇒ UX가 좋지 않음

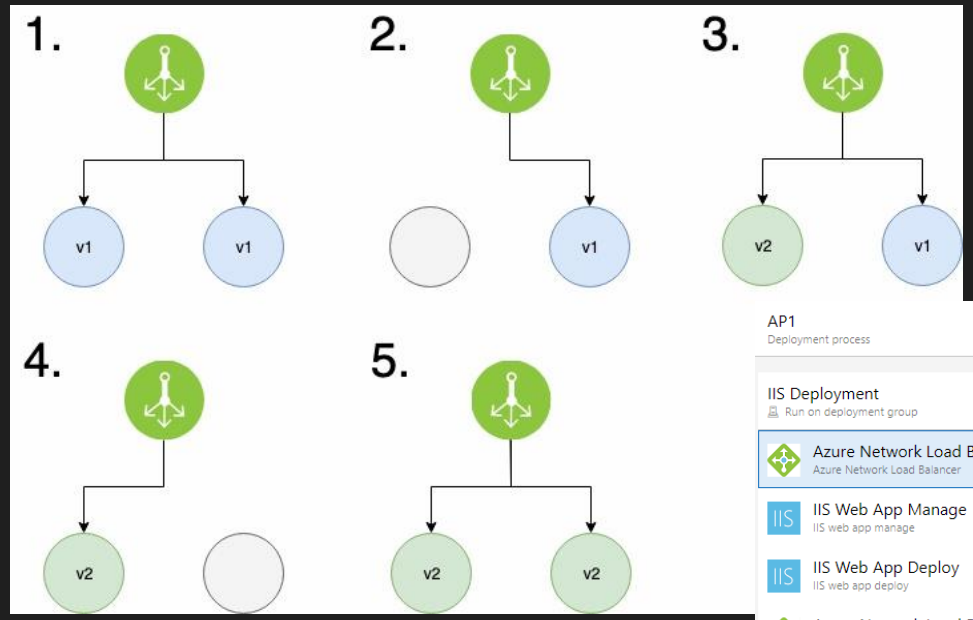
- Shadow Copying: 임시 폴더에 배포 후 IIS에서 변경을 감지해서 적용(Experimental)

⇒ 해결되지 않음

결론

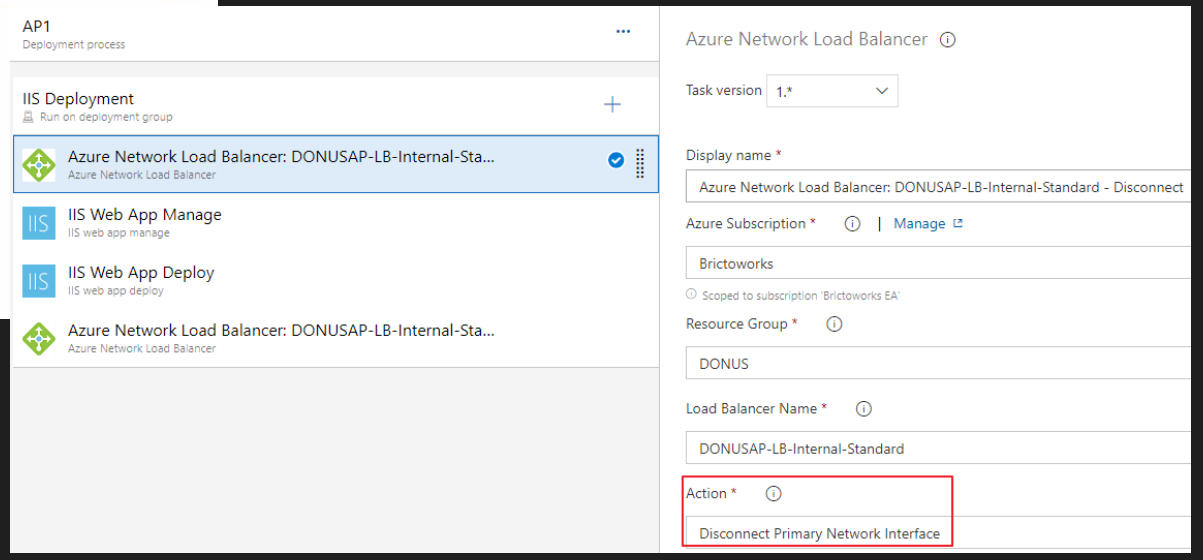
- Rolling 배포 도입

.NET 6용 CI/CD 구축



SKU(Stock Keeping Unit)

- Azure Load Balancer: standard 이상
- Azure Application Gateway: v2 이상



.NET 4.7.2 ⟹ .NET 6

무중단 배포 성공

.NET 6 배포의 순간

.NET 6 업그레이드 후

.NET 6 업그레이드 후 좋아진 것들

- 성능
- C# 9.0, C# 10.0
- Built-in DI(Dependency Injection)

성능 측정

BenchmarkDotNet

- https://github.com/dotnet/BenchmarkDotNet

```
[SimpleJob(RuntimeMoniker.Net472, baseline: true)]
[SimpleJob(RuntimeMoniker.NetCoreApp30)]
[SimpleJob(RuntimeMoniker.NativeAot70)]
[SimpleJob(RuntimeMoniker.Mono)]
[RPlotExporter]
public class Md5VsSha256
{
    private SHA256 sha256 = SHA256.Create();
    private MD5 md5 = MD5.Create();
    private byte[] data;

    [Params(1000, 10000)]
    public int N;

    [GlobalSetup]
    public void Setup()
    {
        data = new byte[N];
        new Random(42).NextBytes(data);
    }

    [Benchmark]
    public byte[] Sha256() => sha256.ComputeHash(data);

    [Benchmark]
    public byte[] Md5() => md5.ComputeHash(data);
}
```

```
BenchmarkDotNet=v0.12.0, OS=Windows 10.0.17763.805 (1809/October2018Update/Redstone5)
Intel Core i7-7700K CPU 4.20GHz (Kaby Lake), 1 CPU, 8 logical and 4 physical cores
  [Host]       : .NET Framework 4.7.2 (4.7.3468.0), X64 RyuJIT
  Net472       : .NET Framework 4.7.2 (4.7.3468.0), X64 RyuJIT
  NetCoreApp30 : .NET Core 3.0.0 (CoreCLR 4.700.19.46205, CoreFX 4.700.19.46214), X64 RyuJIT
  NativeAot70  : .NET 7.0.0-preview.4.22172.7, X64 NativeAOT
  Mono         : Mono 6.4.0 (Visual Studio), X64


| Method |      Runtime |     N |       Mean |     Error |    StdDev | Ratio |
|------- |------------- |------ |-----------:|----------:|----------:|------:|
| Sha256 |     .NET 4.7.2 |  1000 |   7.735 us | 0.1913 us | 0.4034 us |  1.00 |
| Sha256 |  .NET Core 3.0 |  1000 |   3.989 us | 0.0796 us | 0.0745 us |  0.50 |
| Sha256 | NativeAOT 7.0 |  1000 |   4.091 us | 0.0811 us | 0.1562 us |  0.53 |
| Sha256 |          Mono |  1000 |  13.117 us | 0.2485 us | 0.5019 us |  1.70 |
|        |               |       |            |           |           |       |
|    Md5 |     .NET 4.7.2 |  1000 |   2.872 us | 0.0552 us | 0.0737 us |  1.00 |
|    Md5 |  .NET Core 3.0 |  1000 |   1.848 us | 0.0348 us | 0.0326 us |  0.64 |
|    Md5 | NativeAOT 7.0 |  1000 |   1.817 us | 0.0359 us | 0.0427 us |  0.63 |
|    Md5 |          Mono |  1000 |   3.574 us | 0.0678 us | 0.0753 us |  1.24 |
|        |               |       |            |           |           |       |
| Sha256 |     .NET 4.7.2 | 10000 |  74.509 us | 1.5787 us | 4.6052 us |  1.00 |
| Sha256 |  .NET Core 3.0 | 10000 |  36.049 us | 0.7151 us | 1.0025 us |  0.49 |
| Sha256 | NativeAOT 7.0 | 10000 |  36.253 us | 0.7076 us | 0.7571 us |  0.49 |
| Sha256 |          Mono | 10000 | 116.350 us | 2.2555 us | 3.0110 us |  1.58 |
|        |               |       |            |           |           |       |
|    Md5 |     .NET 4.7.2 | 10000 |  17.308 us | 0.3361 us | 0.4250 us |  1.00 |
|    Md5 |  .NET Core 3.0 | 10000 |  15.726 us | 0.2064 us | 0.1930 us |  0.90 |
|    Md5 | NativeAOT 7.0 | 10000 |  15.627 us | 0.2631 us | 0.2461 us |  0.89 |
|    Md5 |          Mono | 10000 |  30.205 us | 0.5868 us | 0.6522 us |  1.74 |
```

성능

- 주민등록번호 검증 로직
  - Regex
  - Array
  - Parse
  - LINQ

Method	Job	Runtime	N	Mean	Error	StdDev	Ratio	RatioSD	Gen0	Gen1	Allocated	Alloc Ratio
ValidateRegno	.NET 6.0	.NET 6.0	1000	6.671 ms	0.1087 ms	0.1116 ms	0.72	0.02	1953.1250	15.6250	8.79 MB	0.64
ValidateRegno	.NET Framework 4.7.2	.NET Framework 4.7.2	1000	9.189 ms	0.1818 ms	0.2232 ms	1.00	0.00	3062.5000	15.6250	13.84 MB	1.00

성능 37% 향상

메모리 할당 36% 감소

성능

- DB 조회 + DTO mapping
  - 조회(SELECT) 200건
  - SqlClient, Dapper Micro ORM

```
using (var connection = new SqlConnection(connectionString))
{
    var list = connection.Query<Dto>(sql).ToList();
}
```

Method	Job	Runtime	Mean	Error	StdDev	Ratio	RatioSD	Allocated	Alloc Ratio
Select	.NET 6.0	.NET 6.0	17.08 ms	0.342 ms	0.479 ms	1.00	0.00	109.62 KB	1.00
Select	.NET Framework 4.7.2	.NET Framework 4.7.2	16.98 ms	0.309 ms	0.258 ms	1.00	0.02	125.1 KB	1.14

속도는 큰 차이 없음.

메모리 할당은 약간 감소.

성능

- JSON Serialize/Deserialize
  - Newtonsoft.Json (.NET 6, .NET 4.7.2)

Method	Job	Runtime	Mean	Error	StdDev	Ratio	RatioSD	Gen0	Gen1	Allocated	Alloc Ratio
SerializeJson	.NET 6.0	.NET 6.0	4.054 us	0.0235 us	0.0209 us	0.65	0.00	0.7401	-	3.41 KB	0.98
SerializeJson	.NET Framework 4.7.2	.NET Framework 4.7.2	6.192 us	0.0281 us	0.0249 us	1.00	0.00	0.7477	-	3.48 KB	1.00
DeserializeJson	.NET 6.0	.NET 6.0	6.110 us	0.1183 us	0.1620 us	0.82	0.02	0.8240	0.0076	3.82 KB	0.98
DeserializeJson	.NET Framework 4.7.2	.NET Framework 4.7.2	7.592 us	0.0635 us	0.0563 us	1.00	0.00	0.8392	0.0076	3.89 KB	1.00

- System.Text.Json (.NET 6)

Job=.NET 6.0 Runtime=.NET 6.0

Method	Mean	Error	StdDev	Ratio	Gen0	Allocated	Alloc Ratio
SerializeJson	2.402 us	0.0258 us	0.0241 us	1.00	0.3433	1.58 KB	1.00
DeserializeJson	3.628 us	0.0342 us	0.0320 us	1.00	0.2518	1.17 KB	1.00

성능

- JSON Serialize/Deserialize

	.NET 4.7.2 Newtonsoft.Json	.NET 6 Newtonsoft.Json	.NET 6 System.Text.Json
Serialize	6.192µs	**4.054µs**	2.402µs
Deserialize	7.592µs	**6.110µs**	3.628µs

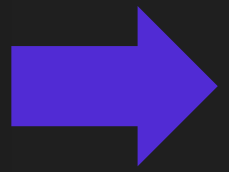
현재까지 약 53%, 24% 성능 향상

System.Text.Json 사용 시 158%, 110% 성능 향상 예상

C# 9.0 record, init

- record + init: Immutable DTO

```
public class Person
{
    ◇
    public Person(string firstName, string lastName)
    {
        FirstName = firstName;
        LastName = lastName;
    }

    8 1 usage
    public string FirstName { get; }
    8 1 usage
    public string LastName { get; }
}
```



```
public record Person(string FirstName, string LastName);

public record Person
{
    public string FirstName { get; init; }
    public string LastName { get; init; }
}
```

Data를 수정할 수 없기 때문에

데이터에 대한 신뢰도 향상 & thread safe한 객체

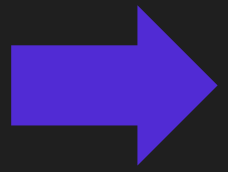
C# 10.0 global using, file-scoped namespace

- GlobalUsing.cs

```
global using Brictoworks.DonusCloud.Core;
global using LogLevel = Brictoworks.DonusCloud.Core.LogLevel;
```

- file-scoped namespace declaration

```
namespace Brictoworks.DonusCloud.Core
{
    new *
    public class Test
    {
    }
}
```



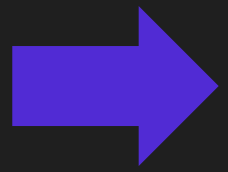
```
namespace Brictoworks.DonusCloud.Core;
```

코드 가독성 향상

Dependency Injection – **진행중**

- 매번 인스턴스 생성해서 사용

```
[HttpPost]
◇  Jaejoon Han +1 *
public void InsertFee([FromBody] InsertFeeCommand command)
{
    using UserContext context = new(User);

    new FeeCommandService().InsertFee(context, command);
}

[HttpPost]
◇  Jaejoon Han +1 *
public void UpdateFee([FromBody] UpdateFeeCommand command)
{
    using UserContext context = new(User);

    new FeeCommandService().UpdateFee(context, command);
}

[HttpPost]
◇  Jaejoon Han +1 *
public void DeleteFee([FromBody] DeleteFeeCommand command)
{
    using UserContext context = new(User);

    new FeeCommandService().DeleteFee(context, command);
}
```



의존성 제거
유닛 테스트 쉬워짐

- 주입 받은 인스턴스 사용

```
private readonly FeeListService _feeListService;
private readonly FeeCommandService _feeCommandService;

◇  Jaejoon Han
public FeeApiController(FeeListService feeListService, FeeCommandService feeCommandService)
{
    _feeListService = feeListService;
    _feeCommandService = feeCommandService;
}

[HttpPost]
◇  Jaejoon Han +1
public void InsertFee([FromBody] InsertFeeCommand command)
{
    using UserContext context = new(User);

    _feeCommandService.InsertFee(context, command);
}

[HttpPost]
◇  Jaejoon Han +1
public void UpdateFee([FromBody] UpdateFeeCommand command)
{
    using UserContext context = new(User);

    _feeCommandService.UpdateFee(context, command);
}

[HttpPost]
◇  Jaejoon Han +1
public void DeleteFee([FromBody] DeleteFeeCommand command)
{
    using UserContext context = new(User);

    _feeCommandService.DeleteFee(context, command);
}
```

Dependency Injection – 진행중

- IOC Container에 200개의 Service 등록

```
◇  1 usage  Jaejoon Han *
public static void RegisterAllTypes<T>(this IServiceCollection services, IEnumerable<Assembly> assemblies,
    ServiceLifetime lifetime = ServiceLifetime.Transient)
{
    IEnumerable<TypeInfo> typesFromAssemblies = assemblies.SelectMany(a => a.DefinedTypes.Where(x => x.GetInterfaces().Contains(typeof(T))));

    foreach (TypeInfo type in typesFromAssemblies)
    {
        services.Add(new ServiceDescriptor(type, type, lifetime));
    }
}
```

열심히 적용 중!

C# Nullable Reference Type 적용 – 진행 예정

- C# Nullable Reference Type
  - Null Reference Exception 방지

```
string warning = null;

string? ok1 = null;

string ok2 = string.Empty;
```

Converting null literal or possible null value into non-nullable type

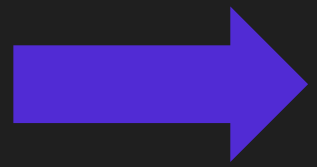
  - C# 8.0에 추가, .NET 6부터 enable이 default

> Null-state analysis and variable annotations are disabled by default for existing projects—meaning that all reference types continue to be nullable. Starting in .NET 6, they're enabled by default for *new* projects. For information about enabling these features by declaring a *nullable annotation context*, see Nullable contexts.

C# Nullable Reference Type 적용 – 진행 예정

- csproj 파일 수정

```
<PropertyGroup>
    <TargetFramework>net6.0</TargetFramework>
    <Nullable>disable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
</PropertyGroup>
```

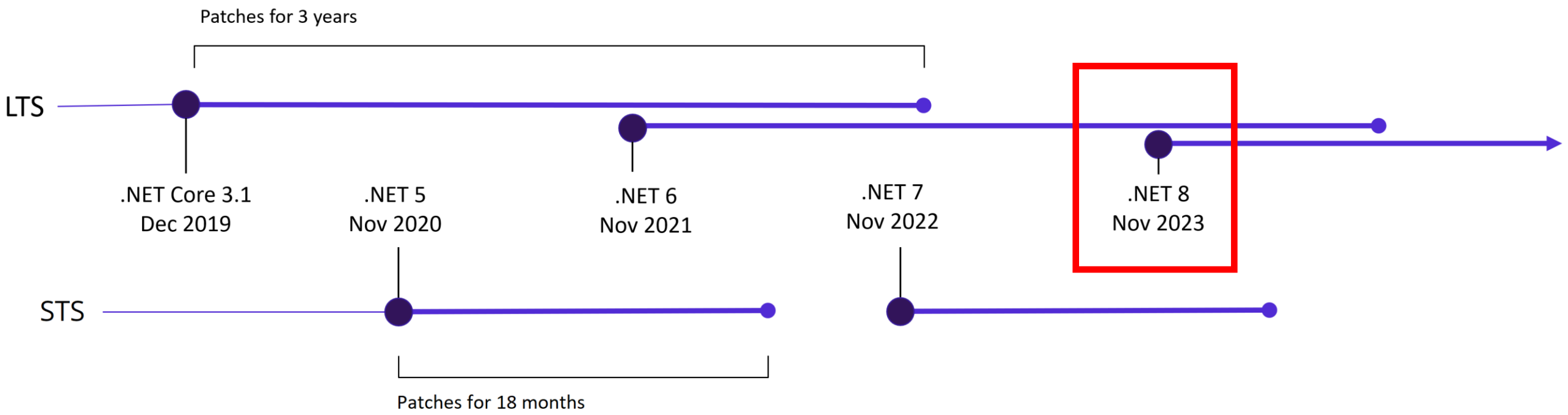


```
<PropertyGroup>
    <TargetFramework>net6.0</TargetFramework>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
</PropertyGroup>
```

수많은 build warnings

```
0>DonusCloud -> D:\Source_Company\DonusCloud_1\DonusCloud\bin\Debug\net6.0\DonusCloud.dll
0>-------- Finished building project: DonusCloud. Succeeded: True. Errors: 0. Warnings: 21813
```

.NET 8 LTS


Patches for 3 years
LTS
.NET Core 3.1
Dec 2019
.NET 5
Nov 2020
.NET 6
Nov 2021
.NET 7
Nov 2022
.NET 8
Nov 2023
STS
Patches for 18 months

함께할 동료를 찾습니다.

https://www.crevisse.com/careers

감사합니다.

jaejoon.han@crevisse.com
@HanJaeJoon