Workshop

Microsoft Azure
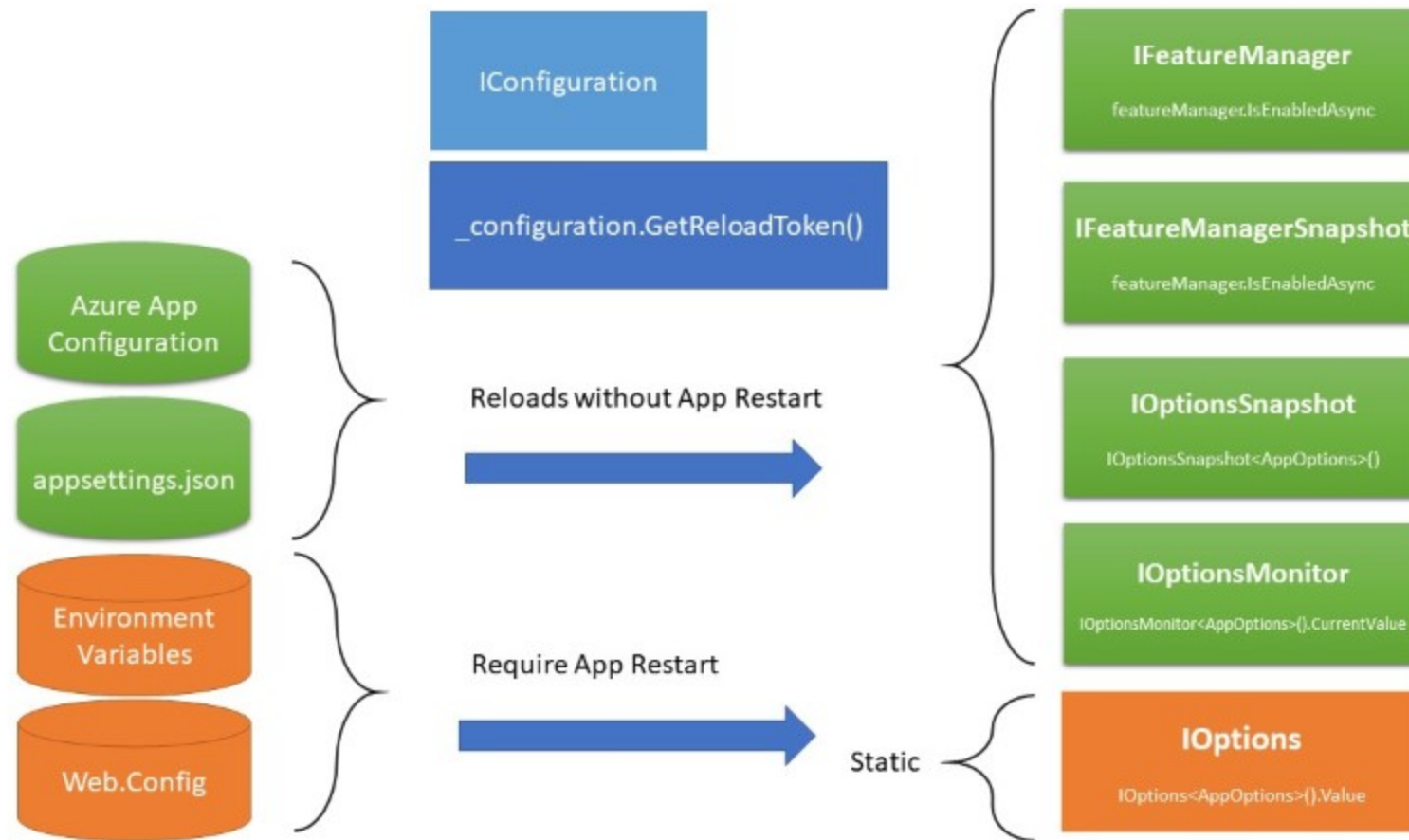
Feature Management And Azure
AppConfiguration

Release Features/Toggles

- **Code branch management**

- **Instant kill switch**

- **Test in production**

- **Flighting**

- **Selective activation**

Overview

Nuget packages

Features

- `Microsoft.FeatureManagement` (Console)
- `Microsoft.FeatureManagement.AspNetCore`

Features and Config (Storage)

- `Microsoft.Extensions.Configuration.AzureAppConfiguration` (Console)
- `Microsoft.Azure.AppConfiguration.AspNetCore`

Demos

- Worker: console app can be windows or linux service or K8s cron/process
- AspNetCore: web applications

`IConfiguration` Infrastructure

Configuration providers implement `IConfiguration` interface and thus raise an event if source has changed via `IConfiguration.GetReloadToken()`.

```
_changeSubscription = ChangeToken.OnChange(
            () => _configuration.GetReloadToken(),
            () => _stale = 1);
```

`FeatureManagement` library utilizes that to raise events and update Feature Flags when the value in the provider changes; thus there is no need to restart the application.

`Microsoft.FeatureManagement`

The infrastructure library that manages creation and execution of Features.

- `IFeatureFilter` creating new Feature Filters. (`PercentageFilter`, `TimeWindowFilter` built-in filters)

- `ISessionManager` (Sessions for Scoped) or `IFeatureManagerSnapshot` can be used for state per user request.

- `ITargetingContext` (User, Groups)

Configuration

```json
"FeatureManagement": {
  "Beta": true,
  "Alpha": false,
  "Cached": {
    "EnabledFor": [
      {
        "Name": "Microsoft.Percentage",
        "Parameters": {
          "Value": 50
        }
      }
    ]
  }
}
```

IFeatureManager

```
using var scope = _serviceProvider.CreateScope();
var featureManager = scope.ServiceProvider.GetRequiredService<IFeatureManager>();
_logger.LogInformation("Alpha Flag is: {alphaFlag}", await featureManager.IsEnabledAsync(nameof(FeatureFlags.Alpha)));
```

The code about can be used to enable to disable feature

Microsoft.FeatureManagement.AspNetCore

- `FeatureGate` on MVC and Api Controllers

- `FeatureTagHelper` for MVC UI

- `IDisabledFeaturesHandler` disabled feature must gracefully return results back

FeatureGate

```csharp
[HttpGet]
[FeatureGate(FeatureFlags.Alpha)]
public IEnumerable<WeatherForecast> Get()
{
    var rng = new Random();
    return Enumerable.Range(1, 5).Select(index => new WeatherForecast
    {
        Date = DateTime.Now.AddDays(index),
        TemperatureC = rng.Next(-20, 55),
        Summary = Summaries[rng.Next(Summaries.Length)]
    })
    .ToArray();
}
```

FeatureTagHelper

Add `@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers` to `_ViewImports.cshtml`

```html
<!-- Check if the feature is enabled using FeatureTagHelper -->
<feature name="@FeatureFlags.Beta">
    <li class="nav-item">
        <a class="nav-link text-dark" asp-area="" asp-controller="Beta" asp-action="Index">Beta</a>
    </li>
</feature>
```

AzureAppConfiguration

Key concepts:

- Authentication `Azure.Identity` new library (ConnectionString or Microsoft Managed Identity/MSI)
- Specify path to key value
- Specify label as filter for environments
- Refresh calls to the storage

Enable AppConfigurations

Add the following to `IHost`

```
return Host.CreateDefaultBuilder(args)
                .UseAzureAppConfiguration(
                "WorkerApp:WorkerOptions",
                "WorkerApp:WorkerOptions:Message",
                options =>
                {
                        options.UseFeatureFlags(flags =>
                        {
                                flags.CacheExpirationTime = TimeSpan.FromSeconds(1);
                        });
                })
```

Questions 😆?