# UnifiedOdds Feed SDK .NET library (.NET Standard 2.0)

Notice: before starting `DemoProject` make sure to enter your bookmaker access token in app.config file and restore nuget packages by right-clicking the solution item and selecting "Restore NuGet Packages".

Below are basic examples that can help you start using sdk.

## A basic way to use the UofSdk

To receive sdk events/messages subscribe to all `Sportradar.OddsFeed.SDK.Api.IUofSdk` and `Sportradar.OddsFeed.SDK.Api. IEntityDispatcher` events.

Note that there is one thread handling message reception and calling your event handler per session, so the processing within that method should be as quick as possible to not prevent following messages from being processed. It is recommended that all `Sportradar.OddsFeed.SDK.Entities.Rest.ISportEvent` processing is done in separate thread.

Below example is the minimum setup to start receiving messages. Note that you open only once, process messages for as long as you want, and then close the feed.

```
var config = UofSdk.GetConfigurationBuilder().BuildFromConfigFile();

var uofSdk = new UofSdk(config);

var session =
uofSdk.GetSessionBuilder().SetMessageInterest(MessageInterest.AllMessages).Build();

uofSdk.ProducerUp += OnProducerUp;
uofSdk.ProducerDown += OnProducerDown;
uofSdk.Disconnected += OnDisconnected;
uofSdk.Closed += OnClosed;

session.OnUnparsableMessageReceived += SessionOnUnparsableMessageReceived;
session.OnBetCancel += SessionOnBetCancel;
session.OnBetSettlement += SessionOnBetSettlement;
session.OnBetStop += SessionOnBetStop;
session.OnFixtureChange += SessionOnFixtureChange;
session.OnOddsChange += SessionOnOddsChange;
session.OnRollbackBetCancel += SessionOnRollbackBetCancel;
session.OnRollbackBetSettlement += SessionOnRollbackBetSettlement;
```

```
uofSdk.Open();
```

# UOF .NET SDK - Migration Guide

The `Sportradar.OddsFeed.SDKCore` library target framework was downgraded from .NET Standard 2.1 to .NET Standard 2.0, so this package can be used in any .NET Core and .NET project and also in old .NET Framework 4.6.2 or newer.

No upgrades are planned for the nuget package Sportradar.OddsFeed.SDK in the future and customer solutions should be upgraded to use `Sportradar.OddsFeed.SDKCore`.

This is your roadmap to a smooth transition from your current SDK version to the latest version. The upgrade is designed to elevate your experience and align the SDK more closely with your business needs.

## Page Contents



## Here are the general steps you can follow to complete the transition:

### 1. Upgrade Dependencies

Upgrade Sdk nuget package to `Sportradar.OddsFeed.SDKCore` 2.0.0 or newer. Before upgrading you might need to upgrade some of the dependent libraries, like libraries for logging or dependency injection.

SDK dependent libraries were upgraded or replaced.

- Removed libraries
  - Newtonsoft.Json
  - Castle.Core
- Added libraries
  - Microsoft.Extensions.Diagnostics.HealthChecks 7.0.11
  - Microsoft.Extensions.Http 7.0.0
- Upgraded libraries
  - Humanizer 2.8.26 -> 2.14.1
  - RabbitMQ.Client 5.1.2 -> 6.5.0
  - Microsoft.Extensions.Logging.Abstractions 3.1.0 -> 7.0.0
  - System.Configuration.ConfigurationManager 4.7.0 -> 7.0.0
- Replaced libraries
  - App.Metrics with OpenTelemetry 1.6.0
  - System.Runtime.Caching with Microsoft.Extensions.Caching.Memory 7.0.0
  - Unity with Microsoft.Extensions.DependencyInjection 7.0.0

## 2. Build new UofSdk instance

The feed instance is now named `UofSdk`. Building new feed instance is changed. You need to build configuration and register all the sdk services before creating new UofSdk instance.

```
var uofConfiguration = UofSdk.GetConfigurationBuilder().BuildFromConfigFile();
var services = new ServiceCollection();
services.AddUofSdk(uofConfiguration);
var uofSdk = new UofSdk(services.BuildServiceProvider());
```

> **Note:** When configuring WebHost services (as in ASP.NET Core apps) via `WebApplicationBuilder` you should create a scope before creating `UofSdk`. Registering services for logging, telemetry and user classes is omitted for simplicity (but don't forget to add them).

```
var webAppBuilder = WebApplication.CreateBuilder(args).WebHost.ConfigureServices(
(hostBuilderContext, serviceCollection) => {
serviceCollection.AddUofSdk(configuration);
});
var app = webAppBuilder.Build();
var uofSdk = new UofSdk(app.Services.CreateScope().ServiceProvider);
```

# 3. Update the methods and classes in your code

Review your codebase to identify any parts that might be affected by the upgrade. Look for deprecated methods or classes that have been removed in the new version. Update your code to use the new APIs provided by the UOF SDK 2.0.0. This may involve making changes to method calls, imports, and class references. Handle any breaking changes or deprecations by updating your code accordingly. You can contact support if you encounter specific issues.

The following classes and methods are changed. Hence, you will be needed to update your code to use new names.

## Root Classes Renamed

- IOddsFeed to IUofSdk
- IOddsFeedSession to IUofSession
- Feed to UofSdk
- ReplayFeed to UofSdkForReplay
- IOddsFeedExt to IUofSdkExtended
- FeedExt to UofSdkExtended
- IOddsFeedConfigurationSection to IUofConfigurationSection
- OddsFeedConfigurationSection to UofConfigurationSection
- IOddsFeedConfiguration to IUofConfiguration
- Removed OperationManager (properties moved to IUofConfiguration)
- IEnvironmentSelector - removed SelectIntegration() and SelectProduction() - use SelectEnvironment (SdkEnvironment ufEnvironment)
- Renamed Feed.CreateBuilder() to UofSdk.GetSessionBuilder() for creating new IUofSession
- config section moved to Sportradar.OddsFeed.SDK.Api.Internal.Config.UofConfigurationSection

## Removed methods and classes

- IOutcomeSettlement.Result
- IOddsFeedConfigurationSection.UseIntegrationEnvironment
- IRound.GroupName

- IRound.GetGroupName()

## Added or changed methods

- Added support for IVenue.Courses (returns list of ICourse instead of list of IHole)

## Enum values renamed to CamelCase

- MessageType
- ExceptionHandlingStrategy
- CashoutStatus
- FixtureChangeType
- MarketStatus
- OddsChangeReason
- PropertyUsage
- ResourceTypeGroup

## Changes and/or new features

- Added support for IVenue.Courses (returns list of ICourse instead of list of IHole)
- Added support for ICompetitor.Division - now contains division id and name (moved and replaced from ITeamCompetitor)
- Extended IJersey with SquareColor and HorizontalStripesColor

## Name changes

The following are changed to improve the consistency. Some classes were also moved to different namespace.

- URN -> Urn
- ReplayPlayerStatus.Setting_up -> SettingUp
- IFixture.StartTimeTBD -> StartTimeTbd
- EventStatus.Not_Started -> NotStarted
- FeedMessage.EventURN -> EventUrn
- IRound.Name -> Names
- IRound.PhaseOrGroupLongName -> PhaseOrGroupLongNames
- IProducerManager.Get() -> GetProducer()

## Changed namespaces

- `API` namespace renamed to `Api`
- `REST` namespace renamed to `Rest`
- replay interfaces moved to `Api.Replay`
- UofSdk managers moved to `Api.Managers`
- UofSdk providers moved to `Api.Managers`

- enum types moved to `Common.Enums`
- IUofConfigurationSection moved to `Api.Internal.Config`
- configuration interfaces moved to `Api.Config`
- MessageInterest class moved to `Api.Config`

# 4. Update the Configuration

The configuration settings were split between configuration class and OperationManager. OperationManager is removed and all settings are consolidated within the `IUofConfiguration` interface.

You have two ways for constructing final configuration

1. combining App.config and IConfigurationBuilder or
2. just programmatically via IConfigurationBuilder

Some of the options were removed from App.config section options and can only be configured via configuration builder.

## Upgrade configuration in App.config

Replace

```
<configSections>
   <section name="oddsFeedSection"
type="Sportradar.OddsFeed.SDK.API.Internal.OddsFeedConfigurationSection,Sportradar.O
ddsFeed.SDK" />
</configSections>
```

with

```
<configSections>
<section name="uofSdkSection"
type="Sportradar.OddsFeed.SDK.Api.Internal.Config.UofConfigurationSection,Sportradar
.OddsFeed.SDK" />
</configSections>
```

## Other changes

You'll need to re-configure the following App.config configuration attributes.

| Key | IsRequired | Description |
|---|---|---|
| accessToken | yes | The token you are currently using can also be used with our new version |
| defaultLanguage or desiredLanguages | yes | desiredLanguages is renamed from supportedLanguages |
| nodeId | no | Recommended to be set – must be unique per UofSdk instance |
| environment | no | If not set, 'Integration' will be used. Note: renamed from ufEnvironment |
| supportedLanguages | no | This is removed. Use desiredLanguages |
| inactivitySeconds | no | This is removed from App.config. Can be set through configuration builder |
| host | no | This setting is used only when using Custom environment |
| useSsl | no | This setting is used only when using Custom environment |
| virtualHost | no | This setting is used only when using Custom environment |
| apiHost | no | This setting is used only when using custom environment |
| apiUseSsl | no | This setting is used only when using Custom environment |
| exceptionHandlingStrategy | no | Sets a ExceptionHandlingStrategy enum member specifying how to handle exceptions thrown to outside callers ('Catch' or 'Throw') |
| disabledProducers | no | Sets the comma delimited list of ids of disabled producers (e.g. '1,2,7,9') |
| maxRecoveryTime | no | This is removed from App.config. Can be set through configuration builder |

| Key | IsRequired | Description |
| --- | --- | --- |
| adjustAfterAge | no | This is removed from App.config. Can be set through configuration builder |
| httpClientTimeout | no | This is removed from App.config. Can be set through configuration builder |
| recoveryHttpClientTimeout | no | This is removed from App.config. Can be set through configuration builder |

### Through ConfigurationBuilder

The full configuration can be also setup via ConfigurationBuilder obtained via `UofSdk.GetUofConfigurationBuilder()`. The resulting `IUofConfiguration` contains all the previously set configurations for the SDK.

## 5. Test your project

Thoroughly test your project after making the changes. Test all critical functionality to ensure that everything still works as expected. Pay special attention to any areas of your setup that interact with the sdk, as these are likely to be the most affected by the upgrade.

## 6. Update the Documentation

Update your project's documentation and any training materials to reflect the changes introduced by the upgrade. This will help your team members understand and work with the new version.

## 7. Deploy to production

Once you are confident that your project works correctly with the upgraded sdk, you can deploy the updated version to your production environment.

## 8. Monitoring and Maintenance

After deployment, monitor your project closely for any unexpected issues or performance problems. Be prepared to address any post-upgrade issues promptly.

## 9. Feedback and Reporting

If you encounter any bugs or issues in the `Sportradar.OddsFeed.SDKCore` v2.0.0 or newer, consider reporting them to support@sportradar.com. Providing feedback can help improve the SDK for future releases.