OFIQ C# Wrapper

A modern .NET 8 wrapper library for the Open Source Face Image Quality (OFIQ) C++ library, providing a type-safe, performant C# API for facial image quality assessment.

Features

- **Full ISO/IEC 29794-5 Compliance**: Wraps all 28 quality measures defined in the standard
- **Cross-Platform Support**: Windows, Linux, macOS (x64, ARM64)
- **Modern C# 12**: Leverages latest .NET 8 features including primary constructors, records, and enhanced pattern matching
- **Safe Interop**: Memory-safe P/Invoke with proper resource cleanup
- **Multiple Image Formats**: Support for PNG, JPEG, BMP via System.Drawing.Common
- **Comprehensive Error Handling**: Structured exception hierarchy with detailed error information
- **Performance Optimized**: Minimal overhead over native calls with efficient memory usage

Project Structure

```
OFIQ-CSharp-Wrapper/
├── src/
│   ├── OFIQ.Native/          # P/Invoke interop layer
│   ├── OFIQ.Core/            # Core data structures and types
│   ├── OFIQ/                 # Main API (OFIQEngine)
│   └── OFIQ.Extensions/      # DI integration and helpers
├── samples/
│   ├── OFIQ.Samples.Console/ # Console application example
│   └── OFIQ.Samples.WebApi/  # Web API example
└── tests/
    ├── OFIQ.Tests.Unit/      # Unit tests
    ├── OFIQ.Tests.Integration/ # Integration tests
    └── OFIQ.Benchmarks/      # Performance benchmarks
```

Quick Start

Prerequisites

- .NET 8.0 SDK or later
- OFIQ native library (`libofiq_lib.so`/`libofiq_lib.dylib`/`ofiq_lib.dll`)
- OFIQ configuration files and models

Installation

1. Add the OFIQ C# wrapper to your project:

```
<PackageReference Include="OFIQ" Version="1.0.0" />
```

2. Ensure the native OFIQ library is available in your runtime path.

Basic Usage

```csharp
using OFIQ;
using OFIQ.Core.Types;

// Initialize the engine
using var engine = new OFIQEngine();
engine.Initialize("/path/to/ofiq/config");

// Assess image quality
var assessment = engine.AssessQuality("/path/to/face.jpg");

// Display results
Console.WriteLine($"Overall Quality: {assessment.OverallQuality:F1}");
foreach (var measure in assessment.QualityMeasures)
{
    if (measure.IsSuccess)
    {
        Console.WriteLine($"{measure.Measure}: {measure.QualityValue:F1}");
    }
}
```

Advanced Usage

```csharp
// Using Bitmap directly
using var bitmap = new Bitmap("/path/to/face.jpg");
var assessment = engine.AssessQuality(bitmap);

// Get version information
var version = engine.GetVersion();
Console.WriteLine($"OFIQ Version: {version}");

// Working with individual measures
var specificMeasure = assessment.GetMeasureResult(QualityMeasure.EyesOpen);
if (specificMeasure.HasValue && specificMeasure.Value.IsSuccess)
{
    Console.WriteLine($"Eyes Open Quality:
{specificMeasure.Value.QualityValue:F1}");
}
```

Quality Measures

The wrapper supports all 28 quality measures from ISO/IEC 29794-5:

Measure ID	Name	Description

Measure ID	Name	Description
0x41	UnifiedQualityScore	Overall quality score
0x42	BackgroundUniformity	Background consistency
0x43	IlluminationUniformity	Lighting consistency
0x44	LuminanceMean	Average brightness
0x45	LuminanceVariance	Brightness variation
0x46	UnderExposurePrevention	Dark image prevention
0x47	OverExposurePrevention	Bright image prevention
0x48	DynamicRange	Contrast range
0x49	Sharpness	Image clarity
0x4A	NoCompressionArtifacts	Compression quality
0x4B	NaturalColour	Color accuracy
0x4C	SingleFacePresent	Single face detection
0x4D	EyesOpen	Eyes open state
0x4E	MouthClosed	Mouth closed state
0x4F	EyesVisible	Eyes visibility
0x50	MouthOcclusionPrevention	Mouth obstruction prevention
0x51	FaceOcclusionPrevention	Face obstruction prevention
0x52	InterEyeDistance	Distance between eyes
0x53	HeadSize	Head proportion
0x54-0x57	CropOfTheFaceImage	Face positioning
0x58-0x5A	HeadPose	Head orientation
0x5B	ExpressionNeutrality	Facial expression
0x5C	NoHeadCoverings	Headwear absence

Configuration

The wrapper requires OFIQ configuration files in JAXN format. Place your configuration files in a directory and provide the path during initialization:

```
engine.Initialize("/path/to/config", "ofiq_config.jaxn");
```

Error Handling

The wrapper provides detailed exception types:

```
try
{
    var assessment = engine.AssessQuality(imagePath);
}
catch (ConfigurationException ex)
{
    Console.WriteLine($"Configuration error: {ex.Message}");
}
catch (ImageLoadException ex)
{
    Console.WriteLine($"Image loading error: {ex.Message}");
}
catch (OFIQException ex)
{
    Console.WriteLine($"OFIQ error: {ex.Message}");
}
```

Platform Support

Platform	**Architecture**	**Native Library**
Windows	x64	`ofiq_lib.dll`
Linux	x64	`libofiq_lib.so`
Linux	ARM64	`libofiq_lib.so`
macOS	x64	`libofiq_lib.dylib`
macOS	ARM64	`libofiq_lib.dylib`

Building from Source

1. Clone the repository:

```
git clone https://github.com/your-org/OFIQ-CSharp-Wrapper.git
cd OFIQ-CSharp-Wrapper
```

2. Build the solution:

```
dotnet build OFIQ-CSharp-Wrapper.sln
```

3. Run tests:

```
dotnet test
```

Sample Applications

Console Application

```
dotnet run --project samples/OFIQ.Samples.Console -- /path/to/config
/path/to/image.jpg
```

Web API

```
dotnet run --project samples/OFIQ.Samples.WebApi
```

Performance

The wrapper is designed for minimal performance overhead:

- **Memory Efficient**: Proper disposal of native resources
- **Thread-Safe**: Safe for concurrent usage
- **Native AOT Ready**: Supports ahead-of-time compilation

Contributing

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Add tests
5. Submit a pull request

License

Acknowledgments

- OFIQ Project Team for the excellent C++ library
- .NET Team for the modern runtime and language features
- Contributors and testers

Support

- Documentation: GitHub Wiki
- Issues: GitHub Issues
- Discussions: GitHub Discussions