Simplifying Code with Dynamic C#

EXAMPLE USES OF DYNAMIC CODE

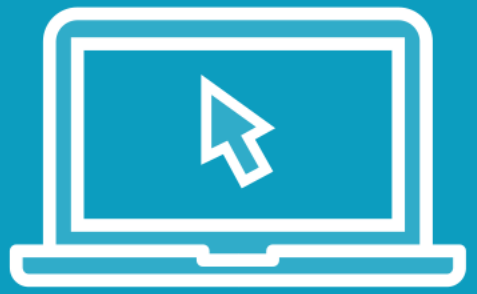
Jason Roberts
.NET MVP

@robertsjason dontcodetired.com

Overview

- Simplifying reflection code
- Reduced code with numeric methods
- COM interop without interop assemblies
- Less plumbing code with dynamic JSON
- Dynamically populating Excel with arbitrary JSON data
- Dynamic JSON in Web API
- ExpandoObject and XAML databinding
- Dynamic SQL query results with Dapper
- Improving test code with dynamics
- Dynamic code considerations

Demo



Simplifying Reflection Code

- Invoke StringBuilder AppendLine method
- InvokeMethodUsingReflection()
- Reflection API (System.Reflection)
- More code
- More “clutter”
- InvokeMethodUsingDynamic()
- Dynamic (runtime) binding
- Reduced code
- Reduced clutter
- Clearer intent

Demo

Creating Unified Numeric Methods

Numeric types have no common interface

Numeric methods require multiple overloads for each numeric type

CommonMathDynamic.Add method

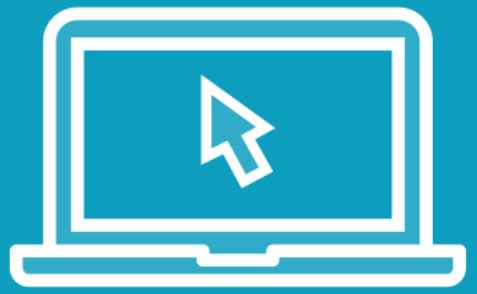
Not type safe

CommonMathDynamicWithGenerics.Add

Type safe but relies on implicit cast

CommonMathDynamicWithGenerics with explicit cast

Demo



COM Interop
Without Interop
Assemblies

Interop with Microsoft Excel

Add text to cell

Auto-fit first column

No interop assembly

```
object excel =
Activator.CreateInstance(excelType);
```

Compiler cannot invoke

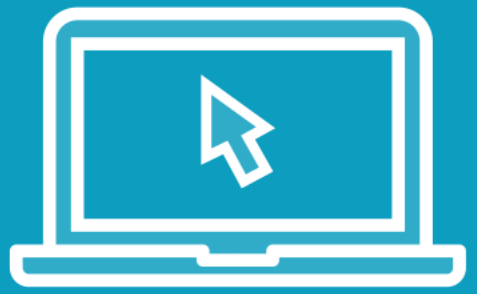
Dynamic binding

Demo

Reducing Plumbing Code with Dynamic JSON

- Parsing JSON data
- JSON.NET NuGet package
- Strong typed
- Additional Customer class code
- Dynamic version
- Reduce amount of code
- Remove Customer class
- Access members dynamically

Demo



Dynamically Populating Excel with Arbitrary JSON Data

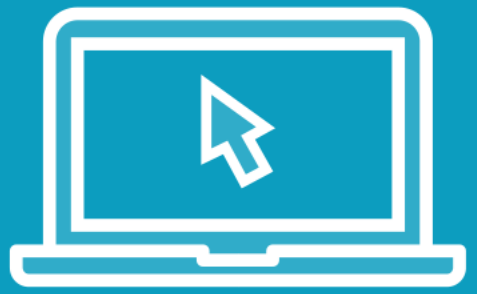
- Combine COM interop & dynamic JSON
- Small number of lines of code
- Read JSON data file
- Dynamically parse
- Populate rows in Excel

Demo

Dynamic JSON in Web API

- Web API method
- Takes JSON data payload
- Dynamic JSON.NET
- No need for strong typed class code

Demo



ExpandoObject and XAML Databinding

XAML databinding to dynamic objects

Simulate getting data from service

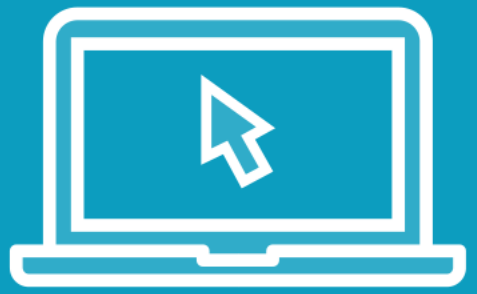
Need to write additional Customer class

Refactor to dynamic

XAML binding to ExpandoObject

Delete additional Customer class

Demo



Dynamic SQL Query Results with Dapper

Dapper micro ORM

Get customer data from SQL server

Typed version

Requires additional Customer class code

Dynamic version

Don’t need additional Customer class

Demo

Improving SpecFlow Test Code with Dynamics

SpecFlow[1] tabular data in steps

Non-dynamic code

Cluttered and hard to read

Dynamic version

Install SpecFlow.Assist.Dynamic NuGet

More readable & understandable

[1] "Business Readable Automated Tests with SpecFlow 2.0" Pluralsight course

Dynamic Code Considerations

Static type safety

Dynamic convenience/benefits

Unit/integration

Automated UI

Performance overhead

Runtime dynamic “lookup”

~100ms

Call site caching

~100ns

Summary

- Simplifying reflection code
- Reduced code with numeric methods
- COM interop without interop assemblies
- Less plumbing code with dynamic JSON
- Dynamically populating Excel with arbitrary JSON data
- Dynamic JSON in Web API
- ExpandoObject and XAML databinding
- Dynamic SQL query results with Dapper
- Improving test code with dynamics
- Dynamic code considerations

Next:

Creating Custom Dynamic Classes