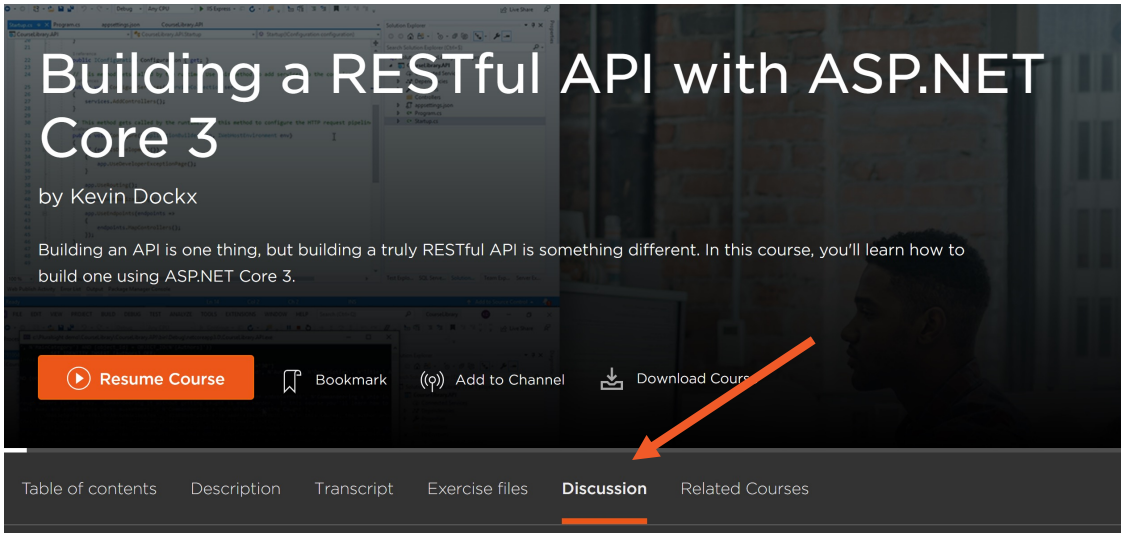
Coming Up

- Course prerequisites and tooling
- Defining reflection
  - Assemblies, modules, types, members
- Reflection use cases and considerations
- Introducing the demo application
- Reading metadata
  - MethodInfo and its specialized forms
  - Early binding, late binding, and BindingFlags

Discussion tab on the course page

Twitter: @KevinDockx



(course shown is one of my other courses, not this one)

Course Prerequisites

.NET Framework Class Libraries with C#

- **Knowledge of C#**

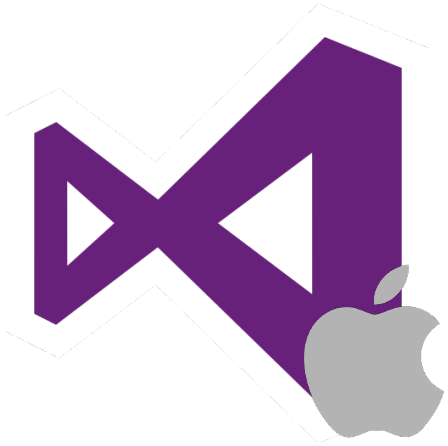
Frameworks and Tooling

Visual Studio 2019
v16.8 or better

.NET 5

Frameworks and Tooling

Visual Studio 2019
v16.8 or better

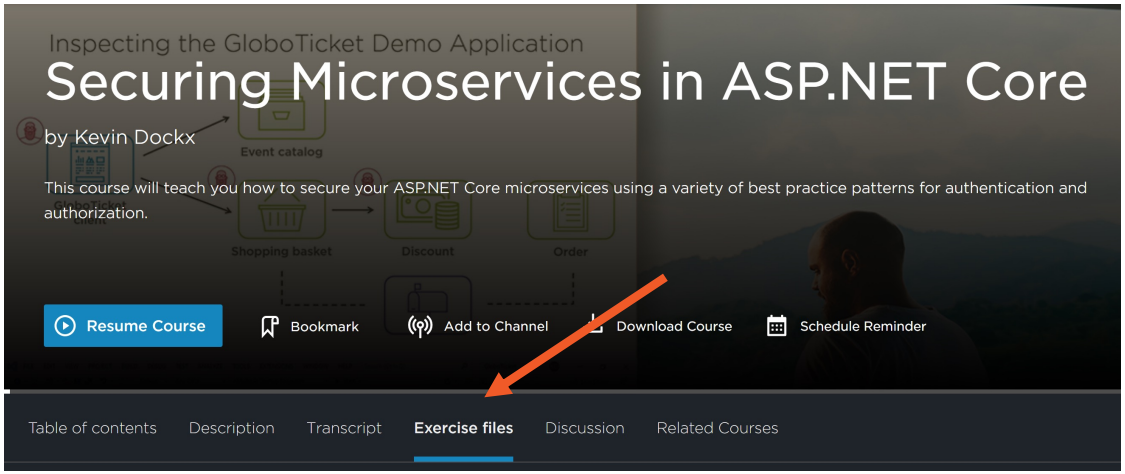


Visual Studio for Mac

Visual Studio Code

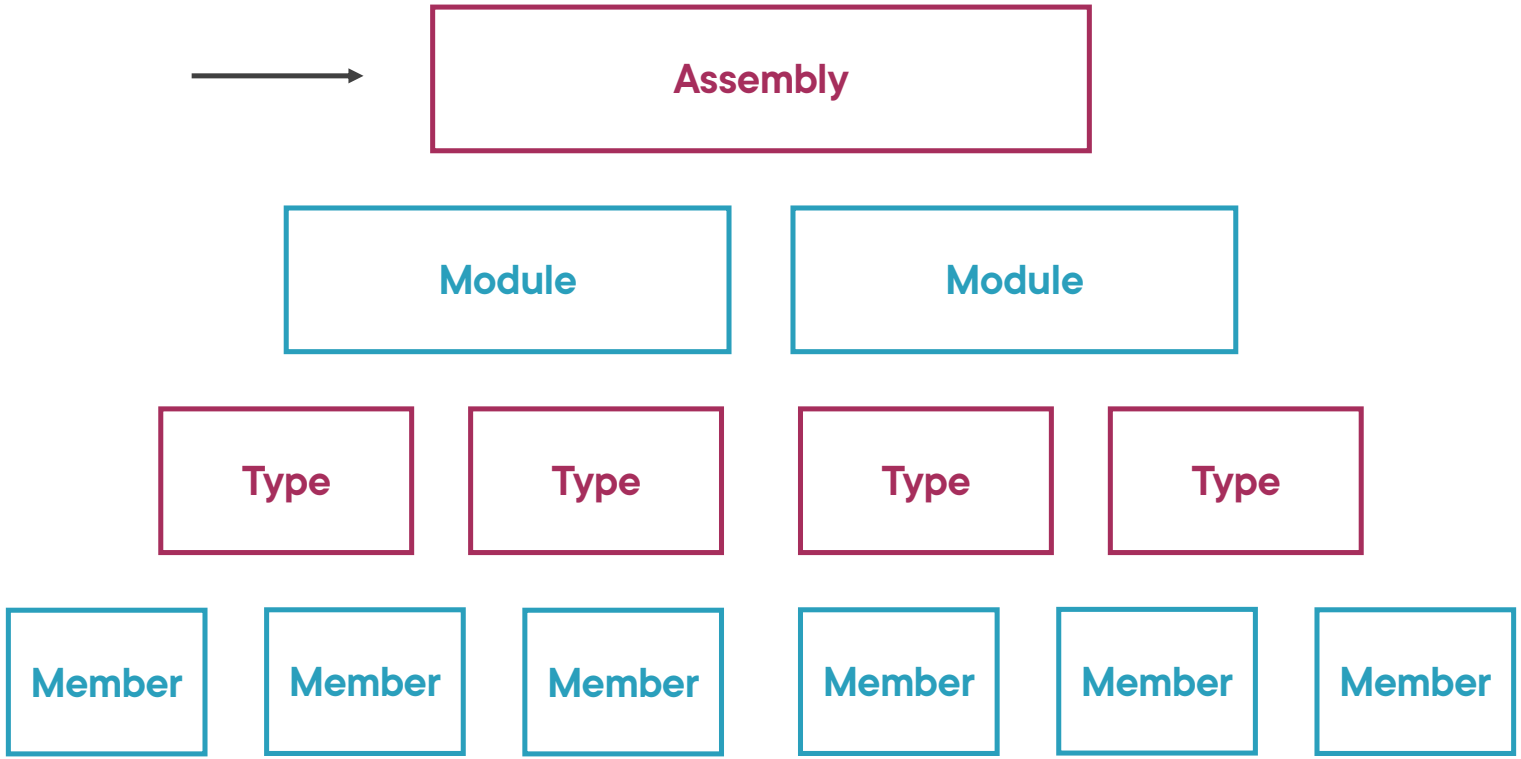
JetBrains Rider

Exercise files tab on the course page



(course shown is one of my other courses, not this one)

Defining Reflection



Assembly

A collection of types and resources that are built to work together and form a logical unit of functionality

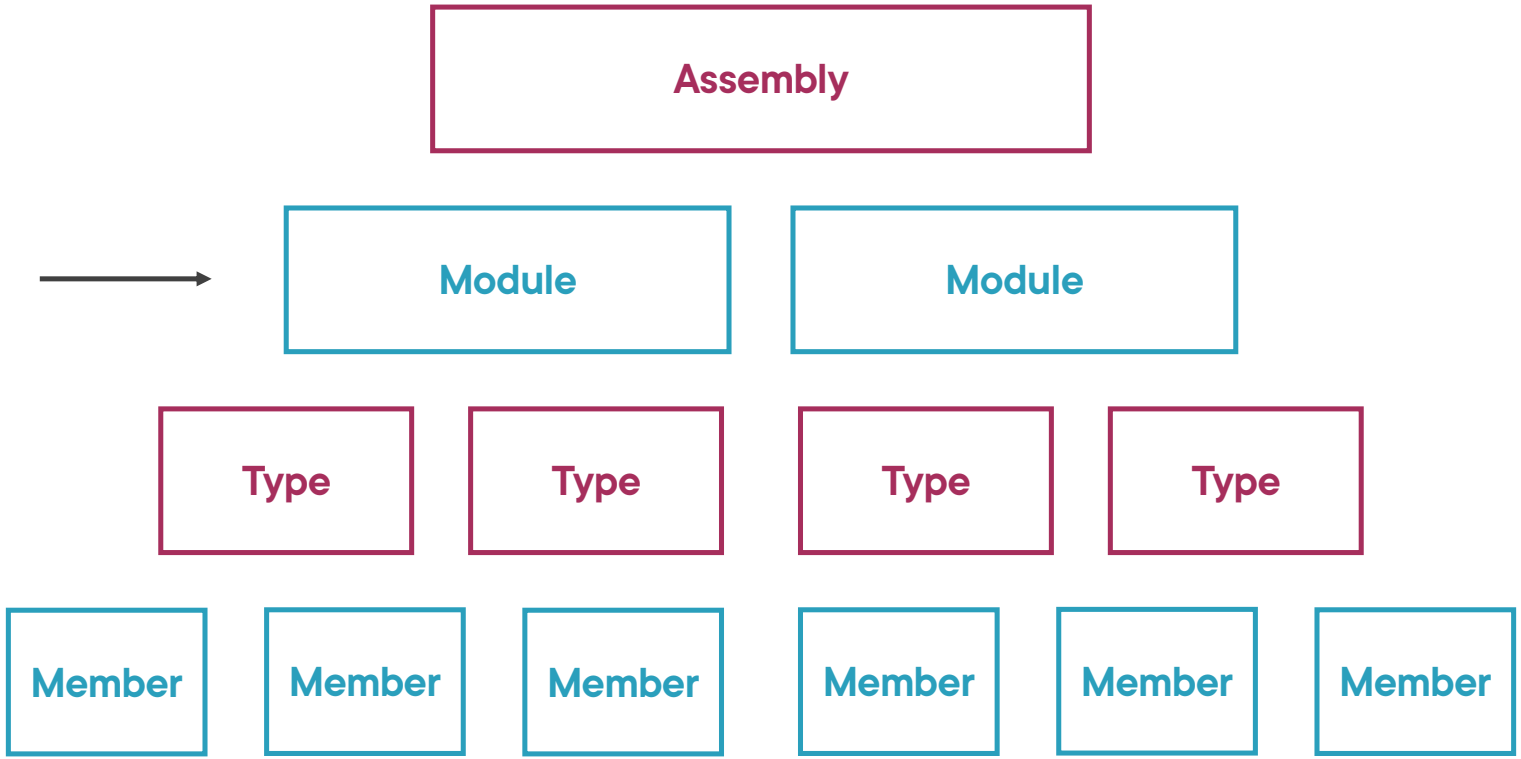
Assembly

.EXE, .DLL

Building blocks of .NET applications

Contains type metadata

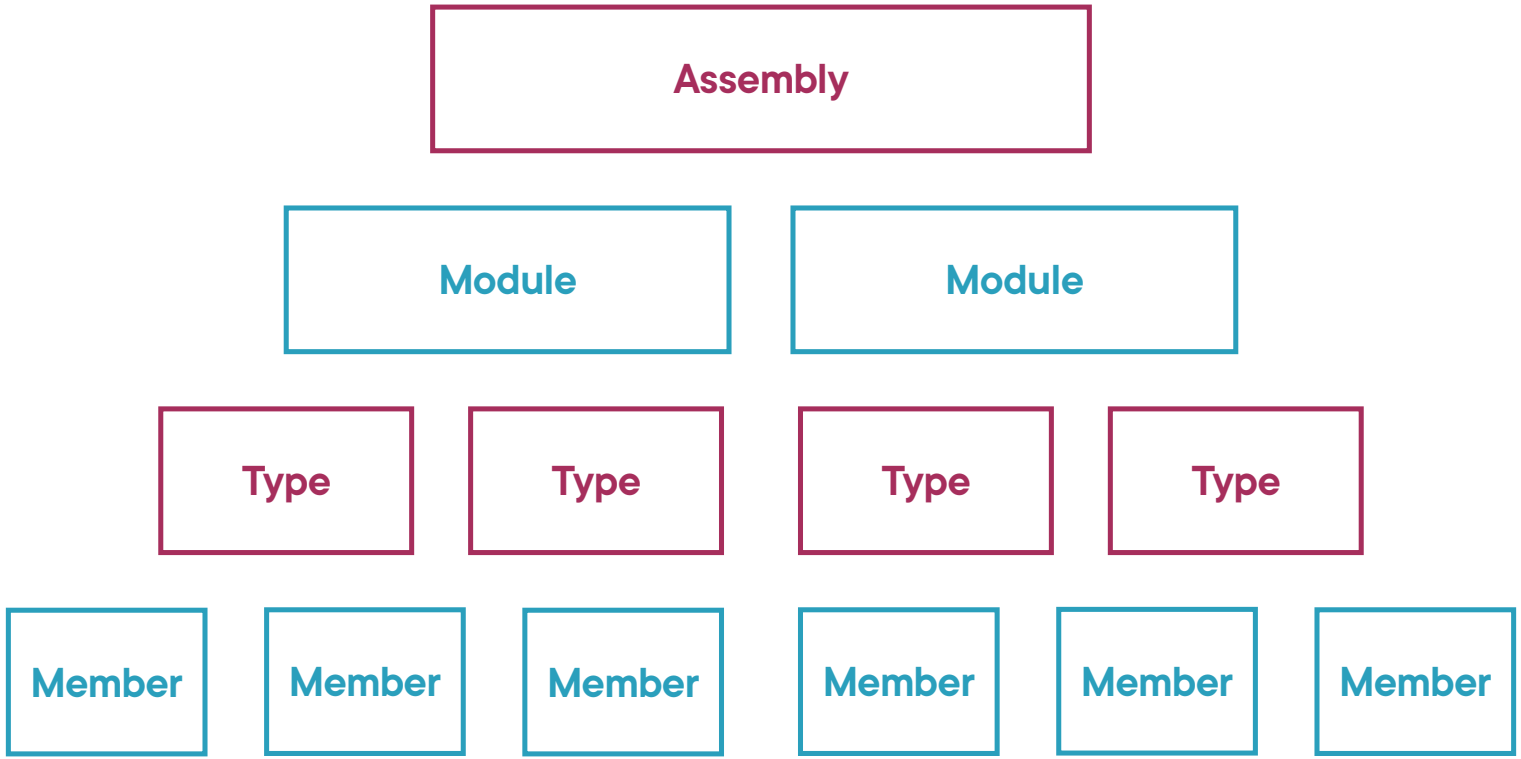
Defining Reflection



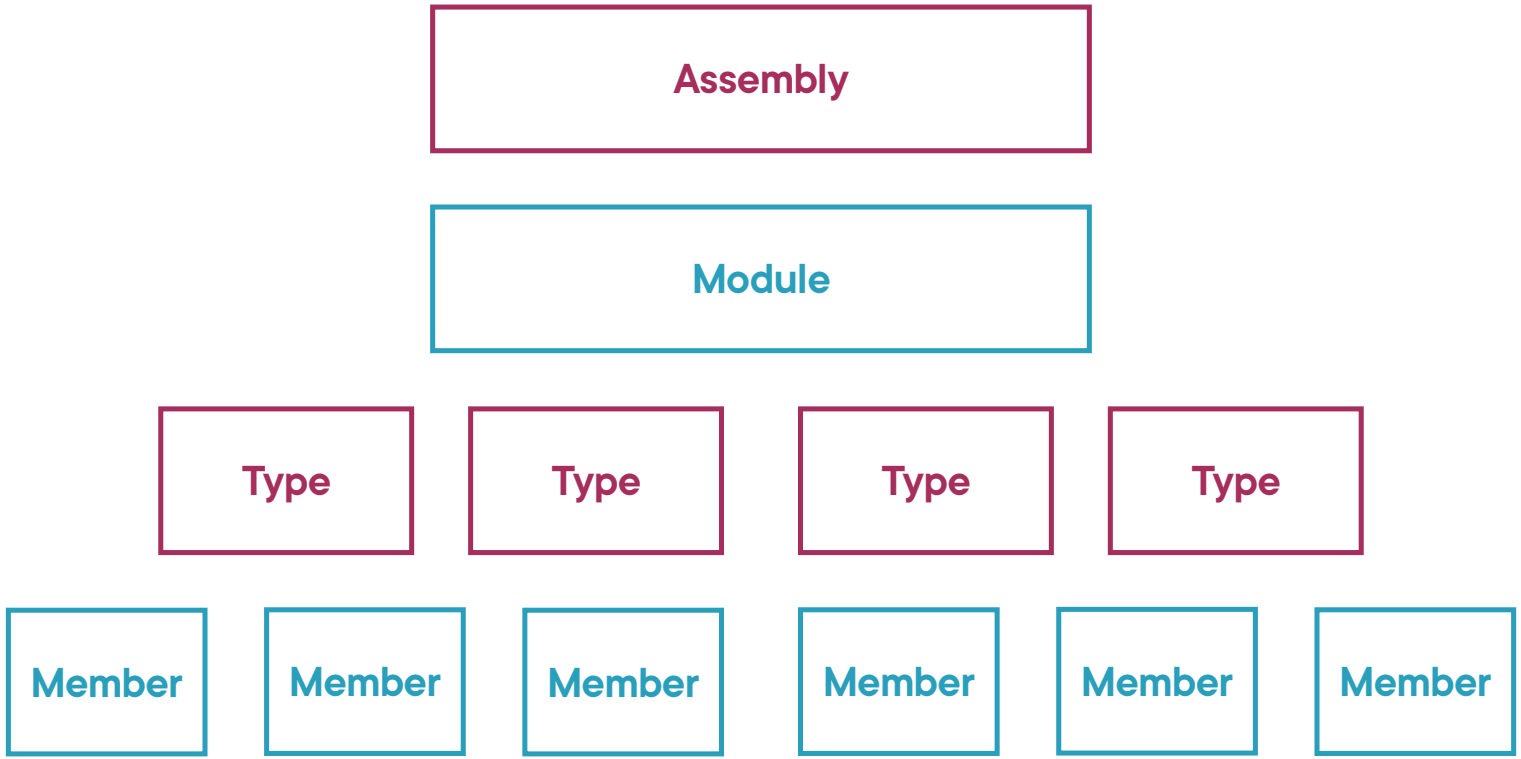
Module

A portable executable file, such as type.dll or application.exe, consisting of one or more classes and interfaces

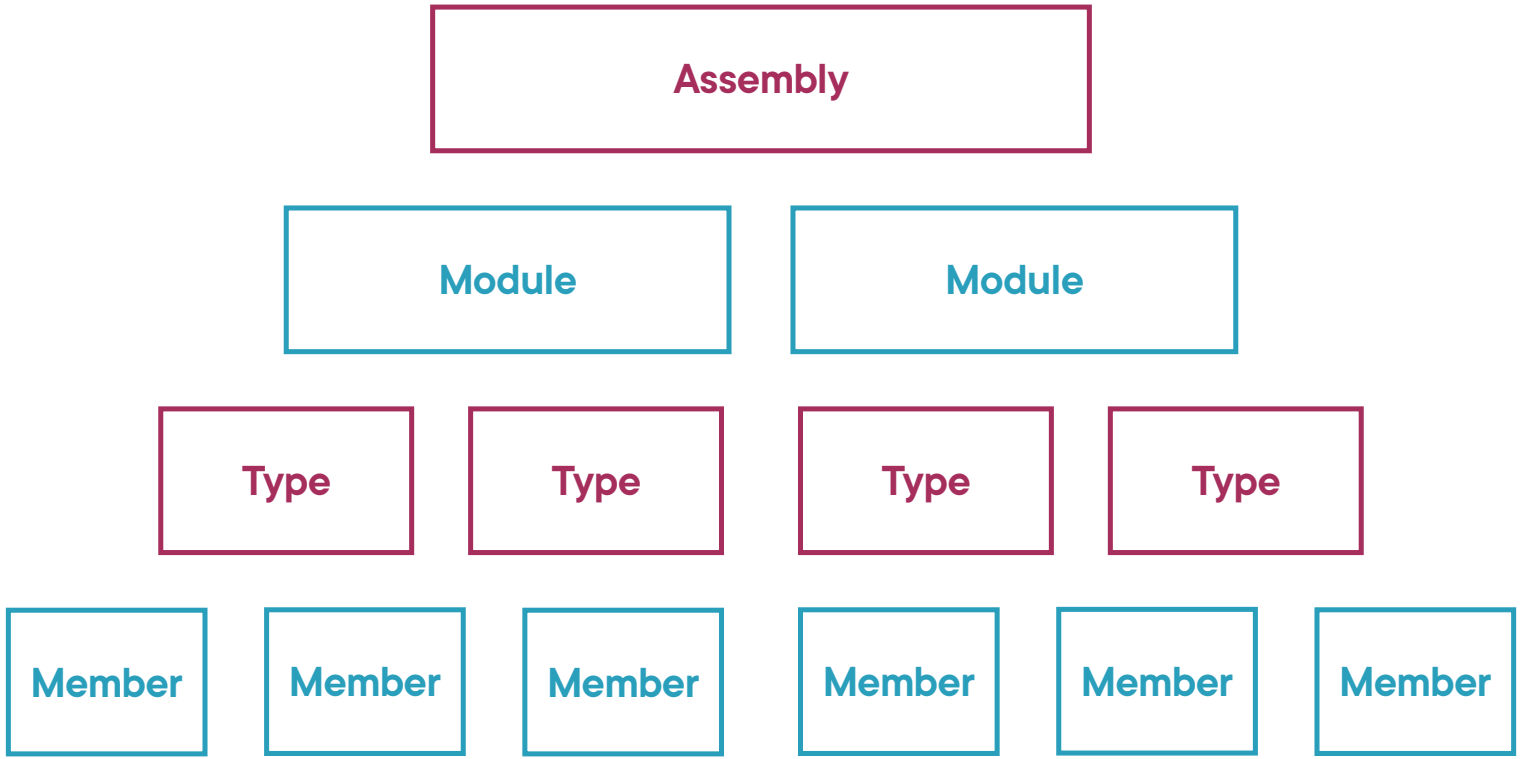
Defining Reflection



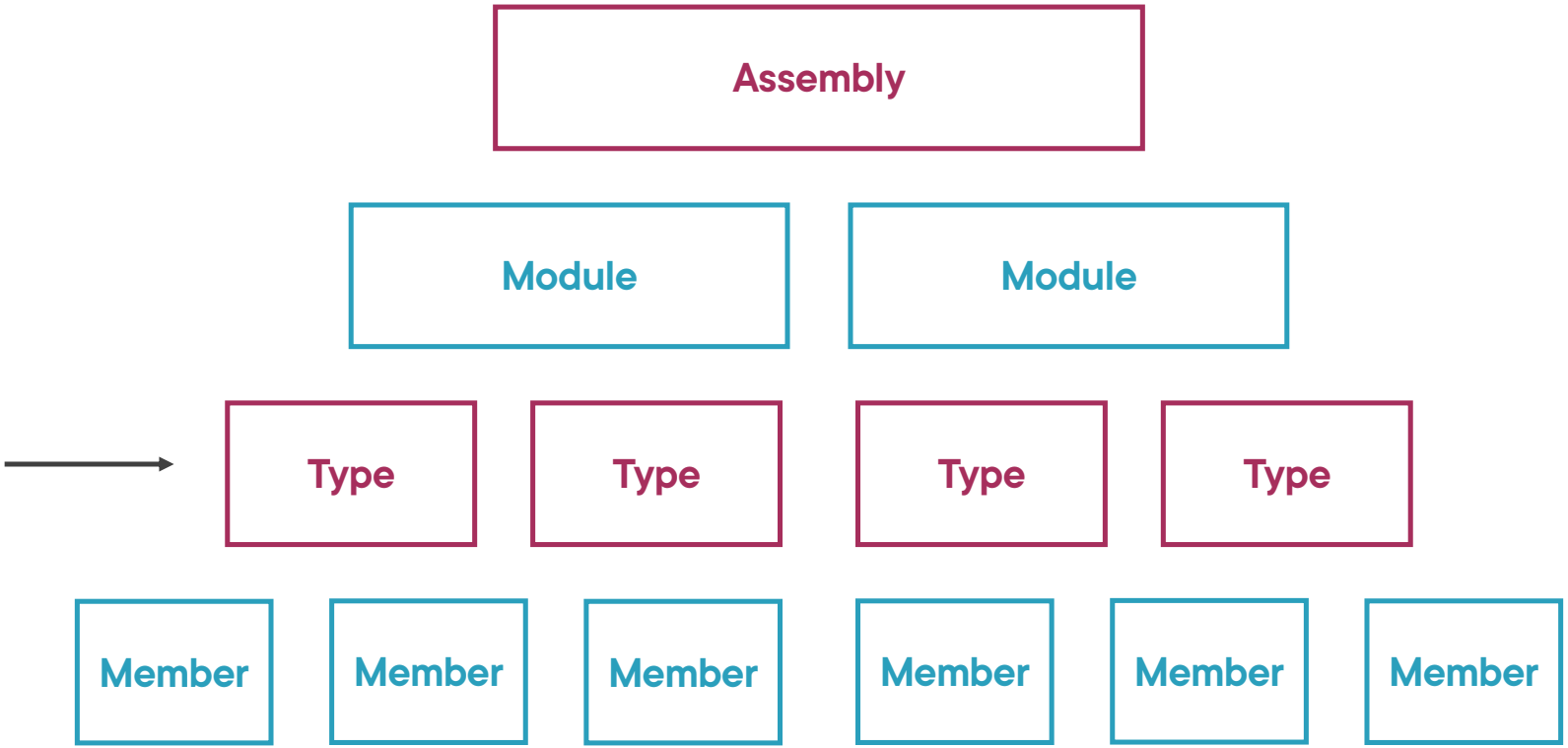
Defining Reflection



Defining Reflection



Defining Reflection



Type

A collection of members: fields that hold data, methods that can be executed, and so on

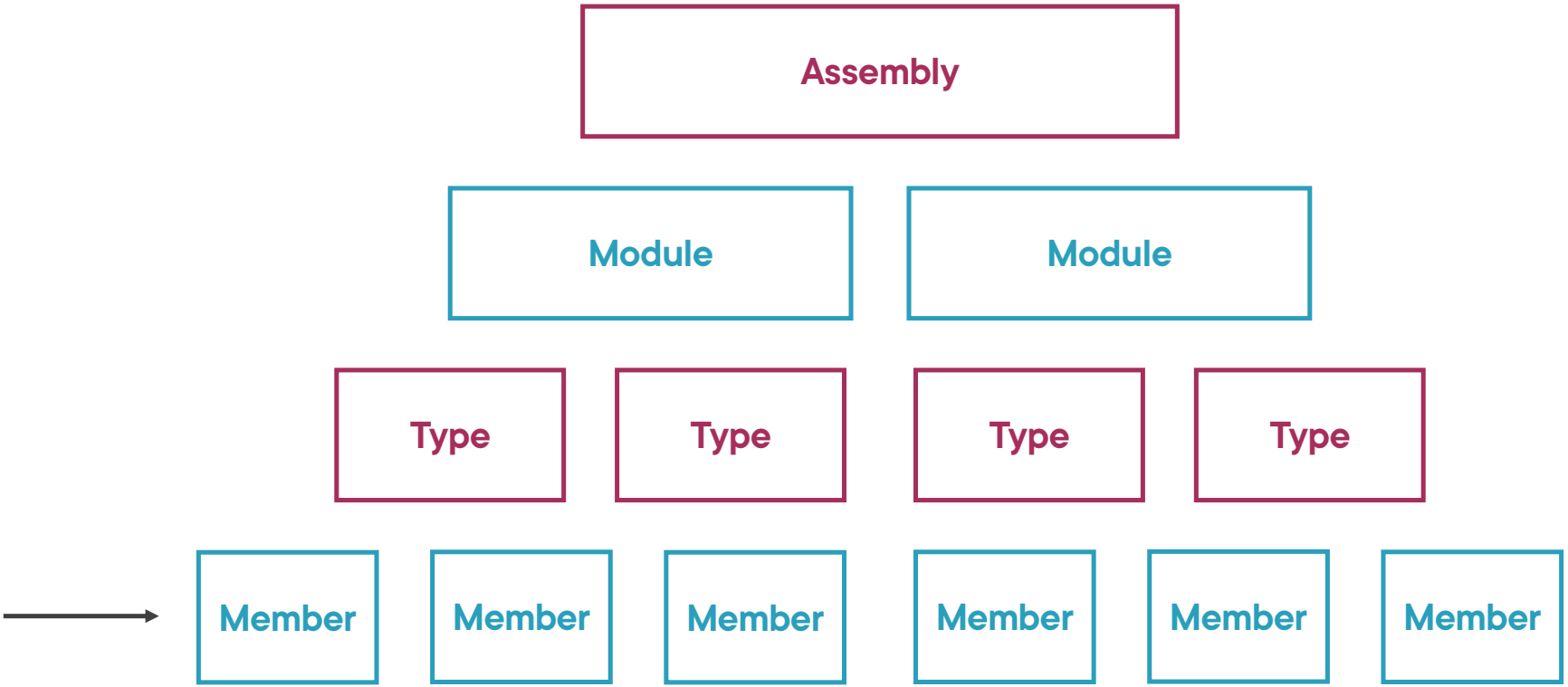
Type

class, struct, enum, …

A type contains metadata describing itself

- **Base type**
- **Interfaces**
- **Permitted operations**
- **…**

Defining Reflection



Member

Represents the data and behavior of a type

Reflection

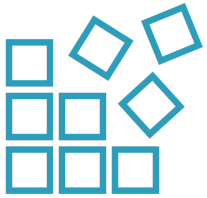
The process by which a computer program can observe and modify its own structure and behavior

Reflection Basics

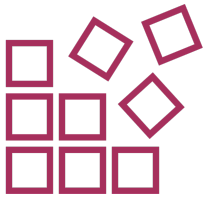
Reflection provides objects that encapsulate assemblies, modules and types

- **System.Reflection namespace**

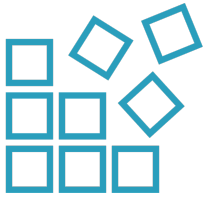
Reflection Basics



Dynamically creating an instance of a type



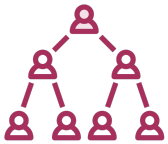
Binding the type to an existing object



Getting the type from an existing object

Reflection Use Cases and Considerations

Dependency injection containers



Calling private or protected methods, fields, properties

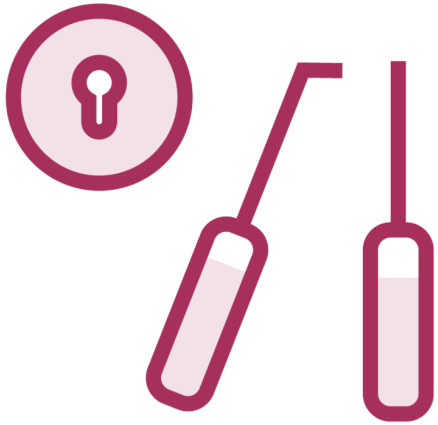
Serialization

Type inspector applications

Code analysis tools

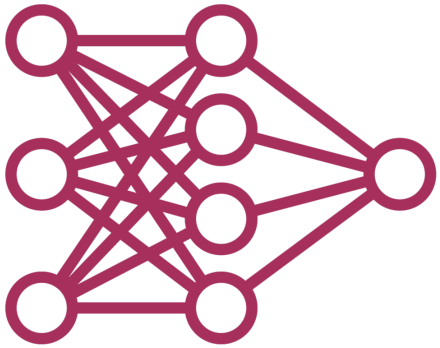
Reflection Use Cases and Considerations

Reflection is relatively slow



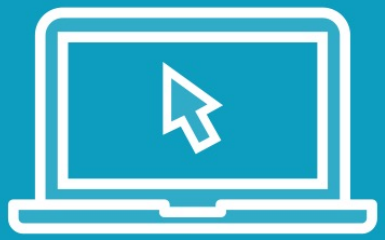
Security is merely a suggestion

Reflection is error-prone due to its dynamic nature



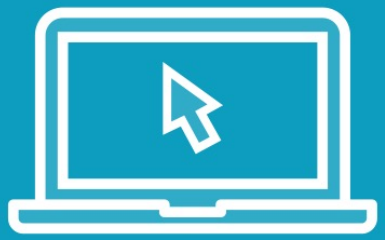
Working with reflection is complex

Demo



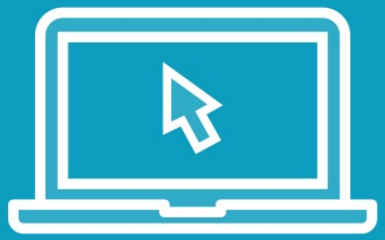
Introducing the demo application

Demo



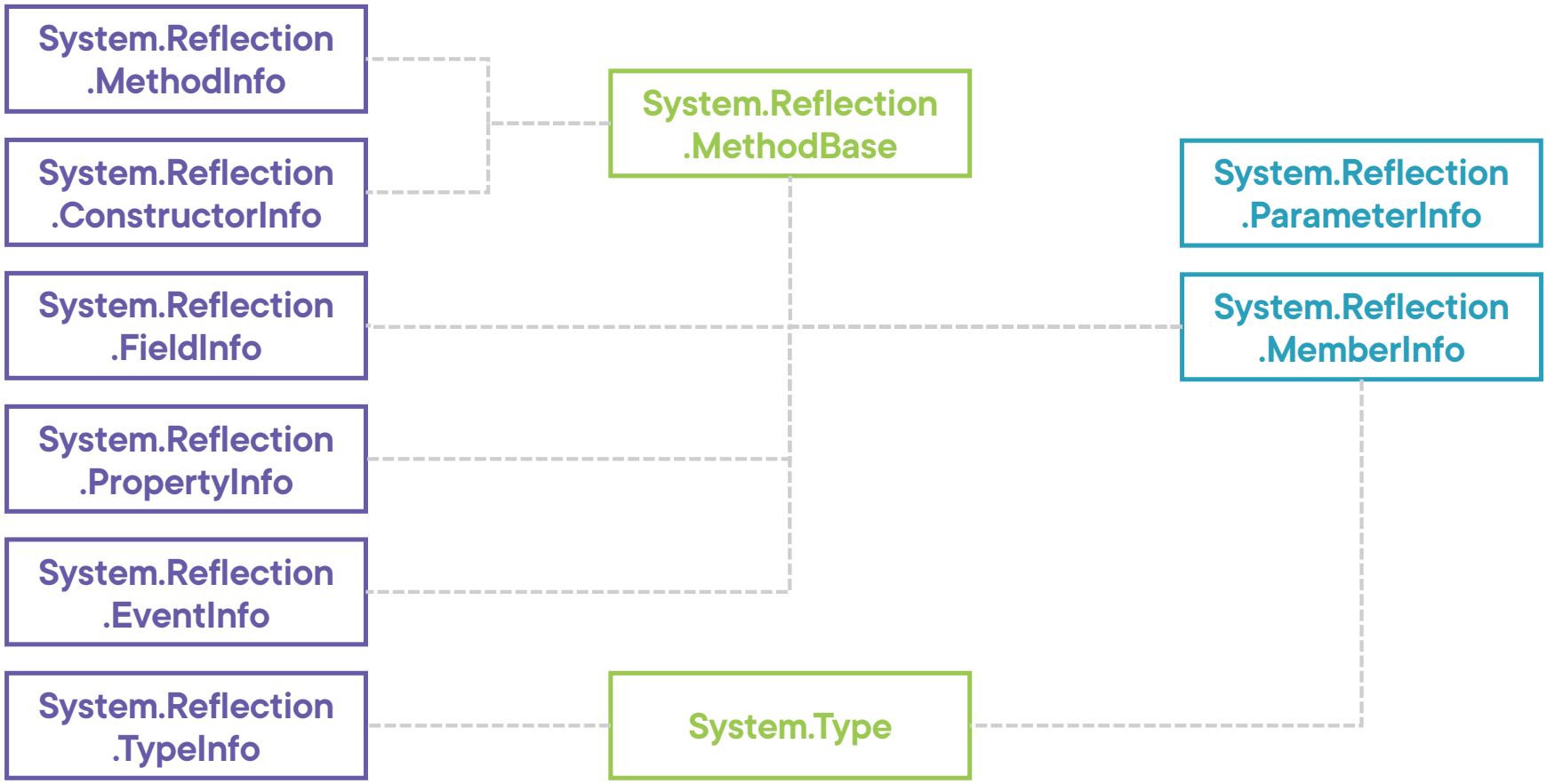
Inspecting a type

Demo



Getting info about a type

Inspecting Specialized Forms of MethodInfo



Binding

The process of locating the declaration (that is, the implementation) that corresponds to a uniquely specified type

Early and Late Binding

Early binding

Looks for methods and properties and checks whether they exist and match at compile time

You won't be able to compile a mismatch

Late binding

The objects are dynamic, so the compiler cannot give a warning

The actual type is only decided upon at runtime

BindingFlags enumeration

Used to control binding and to control how reflection searches

BindingFlags Enumeration

Control how reflection searches

BindingFlags.Public,

BindingFlags.Instance, ...

Control the binding itself

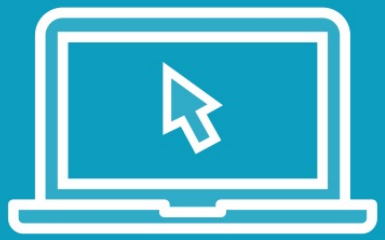
BindingFlags.GetProperty

BindingFlags.SetField, ...

Can be combined bitwise

BindingFlags.Instance | BindingFlags.NonPublic

Demo



Controlling the way reflection searches with BindingFlags

Summary

Reflection is the process by which a computer program can observe and modify its own structure and behavior

- **Dynamic instance creation**
- **Binding a type to an existing object**
- **Getting the type from an existing object**

Summary

Reflection is commonly used

… but it should be used with care

Summary

The first step is gathering metadata

- **MemberInfo class**
  - **MethodInfo, PropertyInfo, TypeInfo, ...**

Summary

Binding

- **Early binding**
- **Late binding**
- **BindingFlags enumeration**