C# Generics

Understanding the Need for Generics

Thomas Claudius Huber
Software Developer

@thomasclaudiush www.thomasclaudiushuber.com

Module Outline

- How this course is structured
- Implement a stack class for doubles
- Make the stack work with any type
    - Use object instead of double
    - Copy and paste for every type
    - Create a generic stack class
- Use existing generic classes

WIRED BRAIN
COFFEE

How This Course Is Structured

A company that runs several coffee shops

They want a .NET console app to load and save employees and organizations

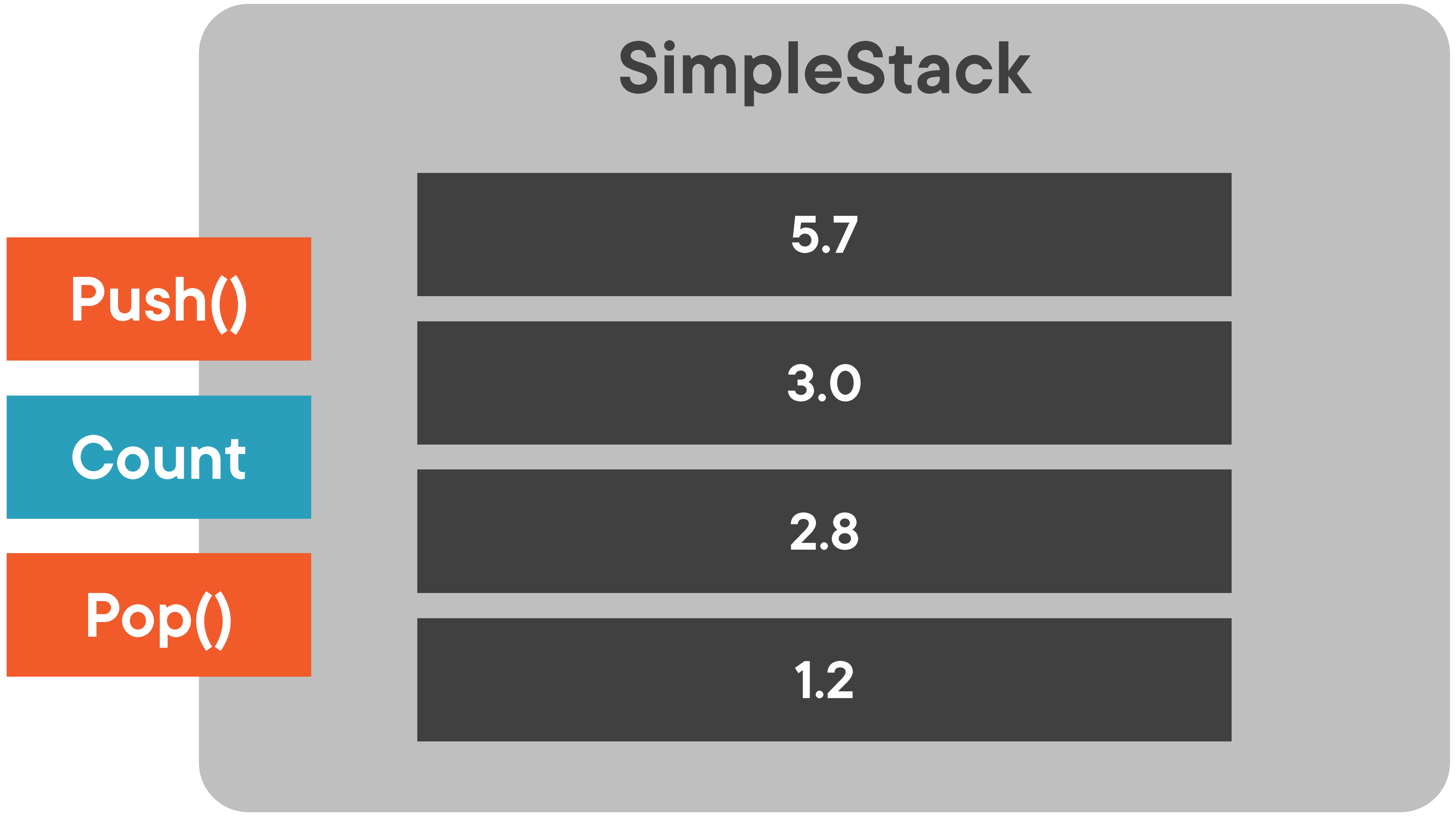
They want you and me as a team to build the .NET console app

How This Course Is Structured

- Understanding the Need for Generics
- Implementing Generic Classes
- Working with Generic Interfaces
- Creating Generic Methods and Delegates
- Knowing the Special Cases with Generics

WIRED BRAIN
COFFEE

Implement a Stack Class for Doubles



Why implementing another Stack class?

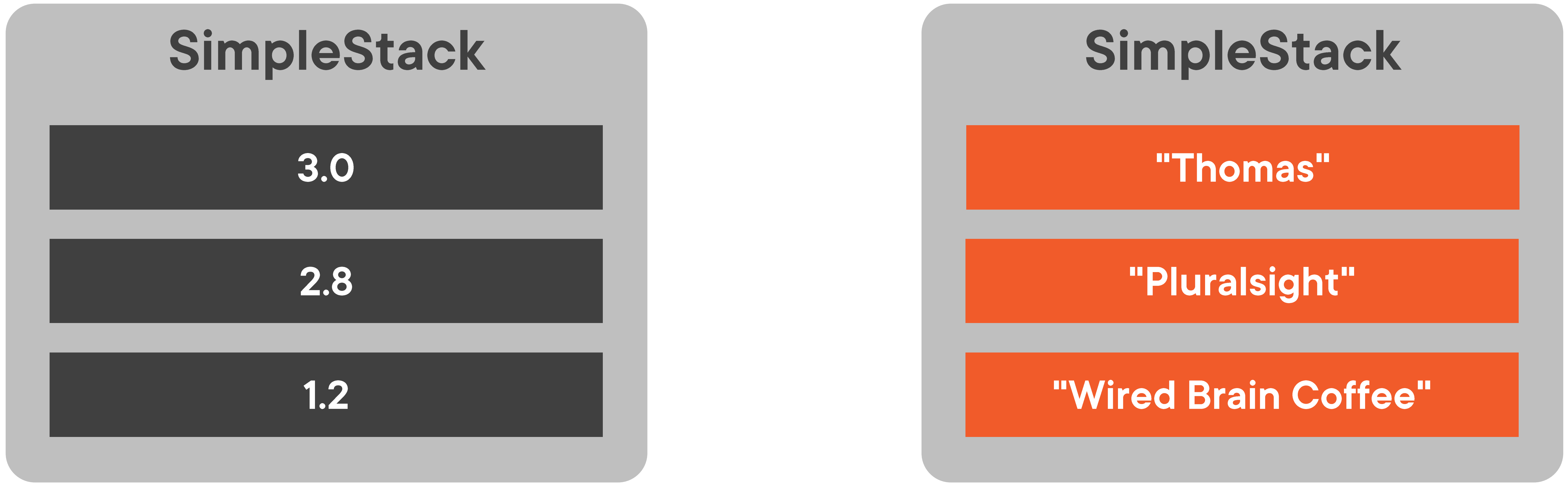
1. Implementing a Stack class is a great exercise

2. It will help you to understand the need for Generics in C#

Demo

Implement a SimpleStack class for doubles

The New Requirement



Demo

Prepare the code
for the new requirement

Demo

Make the SimpleStack usable with any type

- Use object instead of double
- Copy and paste for every new type
- Create a generic SimpleStack class

Know the Advantages of Generics

Code reuse

Object approach

Type-safety

Performance
(no boxing / unboxing)

Copy and paste

Know the Advantages of Generics

Code reuse

```
public class SimpleStack<T>
{
  public void Push(T item) { }
}
```

Type-safety

```
var stack = new SimpleStack<double>();

stack.Push("Thomas"); // Does not compile
```

Performance
(no boxing / unboxing)

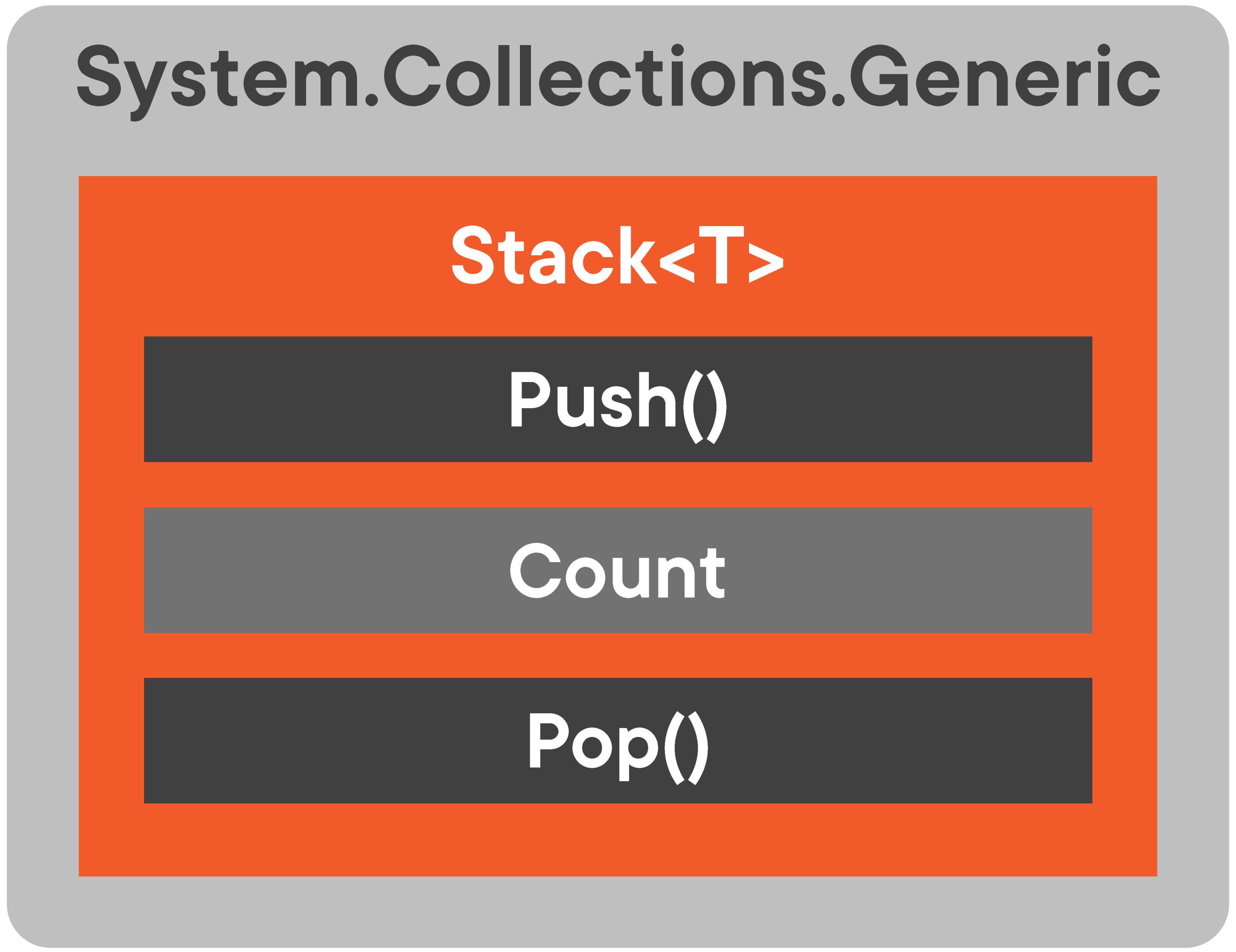
```
var stack = new SimpleStack<double>();

stack.Push(2.8); // No boxing
```

Use the Stack<T> Class of .NET

System.Collections.Generic

- List<T>
- Queue<T>
- Stack<T>
- Dictionary<TKey, TValue>

Use the Stack<T> Class of .NET



Demo

Use the Stack<T> class of .NET

Summary

- Implement a SimpleStack class that works as a storage for any type
  - Use the object type
  - Copy and paste for every type
  - Create a generic SimpleStack<T> class
- Advantages of generics
  - Code reuse
  - Type-safety
  - Performance
- Use the Stack<T> class of .NET

Up Next:
Implementing Generic Classes