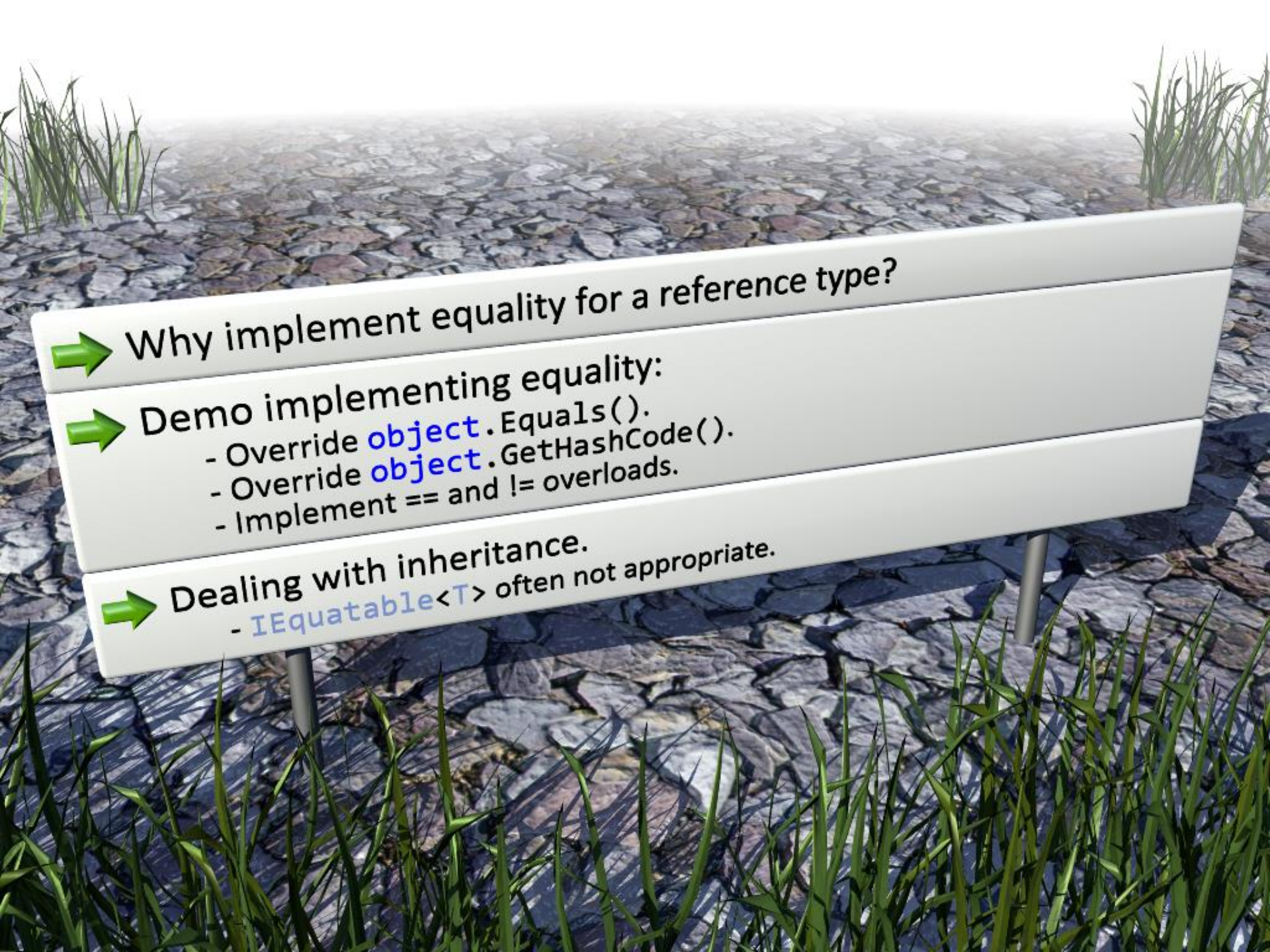
Implementing Equality for Reference Types

Simon Robinson
http://TechieSimon.com
@TechieSimon

pluralsight
hardcore dev and IT training


7. Strings
4. Overriding Equality: Value Types
5. Overriding Equality: Reference Types
8. Comparers and Equality Comparers
10. Structural Equality
6. Comparisons in .NET
3. == in C#
9. Hash Codes
2. Equality in .NET
1.Why Is Equality So Hard?


Why implement equality for a reference type?
Demo implementing equality:
- Override object.Equals().
- Override object.GetHashCode().
- Implement == and != overloads.
Dealing with inheritance.
- IEquatable<T> often not appropriate.

Why Override Equality (Ref Types)?

Why Override Equality (Ref Types)?

```
// a, b of type class MyType
if (a == b) {
```

Want this to do value equality

Reference equality in C# is understood and expected by many devs!!!

This is a good reason for NOT overriding equality for reference types

You Might Override Equality For…

String wrappers

```
class FirstName
{
    private string _value;
```

```
if (name1 == name2) {
```

Mathematical types

```
class Vector
{
```

```
if (vector1 == vector2) {
```

Value equality *might* be clearer here

Alternatively...

Equality Comparer

`IEqualityComparer<T>`

Can't use == for equality with equality comparers

```
if (myEqualityComparer.Equals(obj1, obj2) {
```

Write an equality comparer for your type instead

Code Demo

Equality/Type Safety/OOP

Breaks with inheritance

Type safe

```
public static bool operator == (Food x, Food y)
{
    return x._name == y._name && x._group == y._group;
}
```

```
public static bool operator == (Food x, Food y)
{
    return object.Equals(x, y);
}
```

Not type safe

Works with inheritance

Equality/Type Safety/OOP

Inheritance

Equality

~~Type safety~~

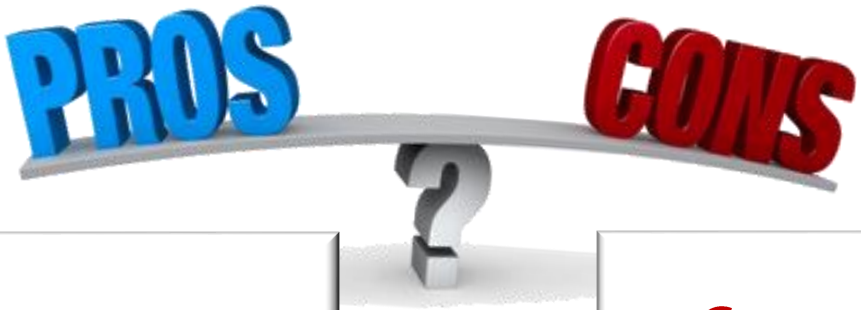
```
public static bool operator == (Food x, Food y)
{
    return object.Equals(x, y);
}
```

Not type safe

Works with inheritance

Is Implementing IEquatable<T> Worth It?

(for sealed classes)



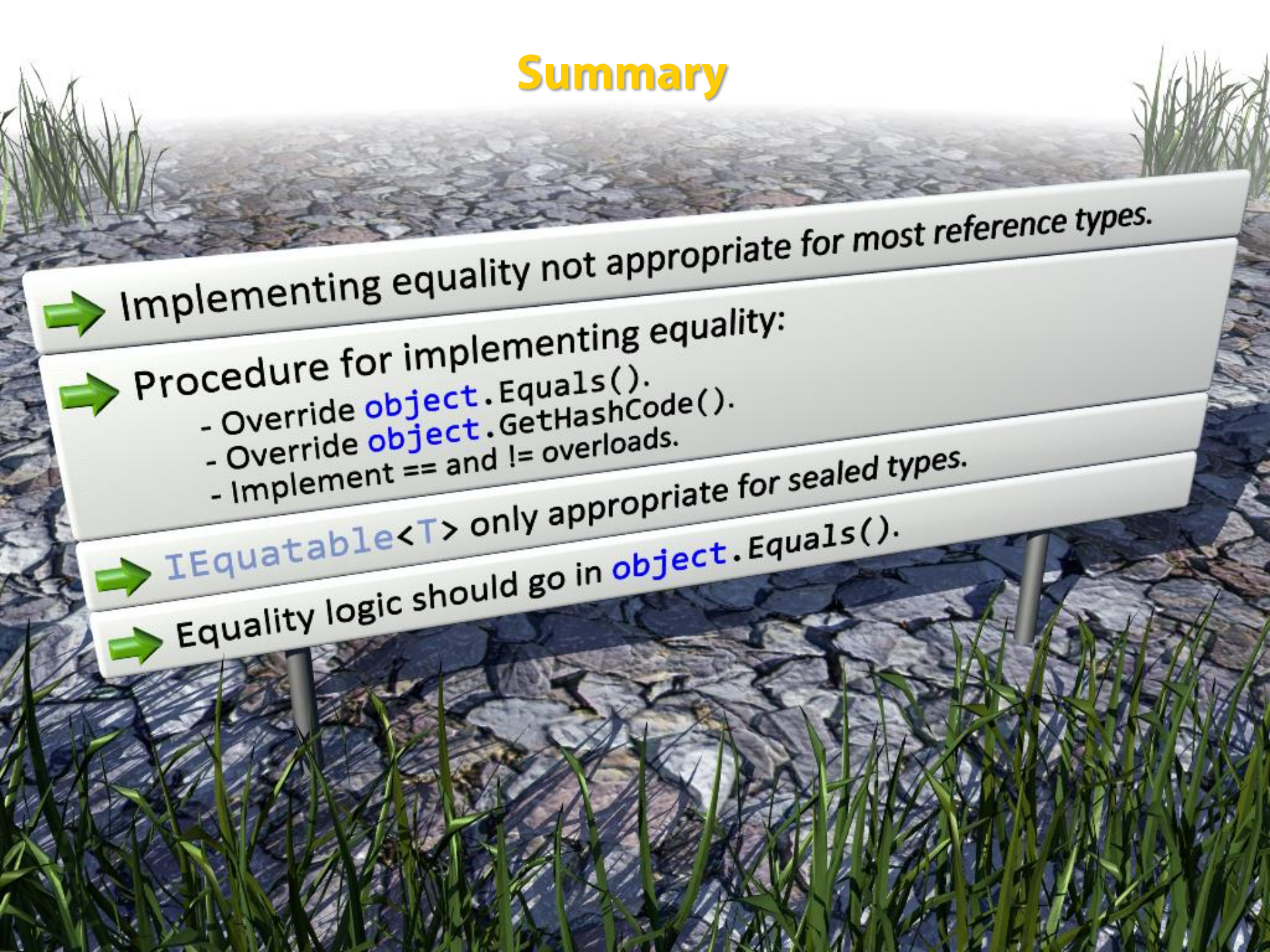
Small performance benefit

(`string` implements `IEquatable<string>`)

Complicates the type

Need to remember
3 ways
to implement equality

Summary


Implementing equality not appropriate for most reference types.
Procedure for implementing equality:
- Override object.Equals().
- Override object.GetHashCode().
- Implement == and != overloads.
IEquatable<T> only appropriate for sealed types.
Equality logic should go in object.Equals().