DOWN THE RABBIT HOLE:

NON-NULLABLE REFERENCE TYPES

About the speaker

Pieter Nijs
Senior .NET Consultant & Mobile Expert @ Xpirit Belgium
xpirit.com/pieter-nijs

Microsoft MVP Windows Development

MADN Board Member
madn.be

Blog.PieEatingNinjas.be
@nijspieter

About is talk

- Relative new feature (C#8)
- Started using it recently
- Convinced about the positive impact
- Little known
  - Cold feet?

1 more thing…

- A LOT of content
- 2 parts
  - Intro/basics (75 minutes)
    - Enabling & using non-nullability
    - Null-forgiving
    - Deserialization
    - Attributes
    - Pittfalls
  - More ‘advanced’ stuff (30+ minutes)
    - More attributes
    - Integration with EF
    - Generics

The problem

The problem

Value Types

- Always have a value
  - Don’t need to check for nulls
  - Cannot assign null value
  - We can just ‘use’ its value
- Intent = have a variable/parameter that always has a value!

The problem

Nullable Value Types (? or Nullable<T>)

- Can be null, is optional
  - Can access value with .Value
  - Check with HasValue or do null-check
- Intent = have a variable/parameter that CAN have a value!

The problem

(Nullable) Value Types are awesome!

- As a developer you can
  - Define them as non-nullable
    - Mandatory value
    - They will always hold a value
  - Define them as nullable
    - Optional value
    - Can or cannot hold a value
    - Not forcing a dummy/'default' value
- The intent is plain as day

The problem

Reference Types

- Can always be null
- Implicit nullable (<> value types)
- Null value can only be checked at runtime
  - Too little too late…
  - Static Code Analyzers?
- Do developers care about nullability?

The problem

Reference Types are jerks!

- As a developer you can NOT
  - Define them as non-nullable
  - Define them as nullable
- No way to tell other devs your intent

The solution

The solution

Nullable and non-nullable Reference Types

- Just like with Value Types
  - Developer defines ‘nullability’ of variables/parameters

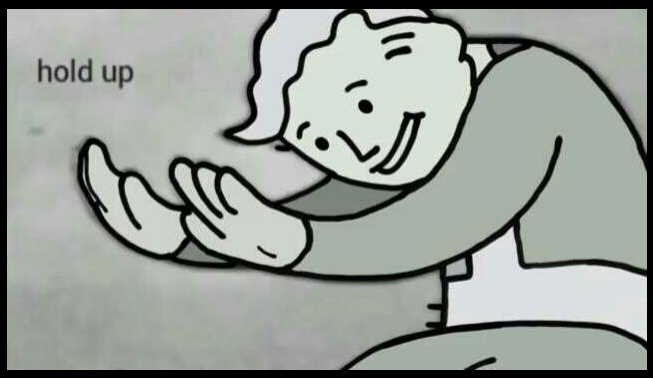
The solution

Nullable and non-nullable Reference Types

- Nullable

  `string? text = null;`
- Non-nullable

  `string text = “Hello World“;`



The solution

Nullable and non-nullable Reference Types

- Non-nullable

  string text = “Hello World“;

=> Breaking language change

The solution

Nullable and non-nullable Reference Types

- Opt-in / opt-out
  - On Project Level (csproj)
    <Nullable>enable</Nullable>
    <Nullable>warnings</Nullable>
  - Directives
    #nullable enable
    #nullable enable annotations
    #nullable enable warnings

The solution

Nullable and non-nullable Reference Types

- Nullable contexts
  - Nullable annotations context
    - Non-nullable by default, can use ? and !
    - Static code analysis (warnings context + maybe null, not null)
  - Nullable warnings context
    - No need for annotations context
    - Static code analysis (warning when we are accessing possible null value)

The solution

Nullable and non-nullable Reference Types

- Best way to learn
  - <nullable>enable</nullable>
  - 'Treat warnings as errors'
    - OR at least custom Code Analysis Rule Set

The solution

Nullable and non-nullable Reference Types

DEMO

Pitfalls

- Structs
- Arrays
- Can’t force caller to use Nullable Annotations
- Force (null! Default!)
- Static analysis can’t “see” everything

Pitfalls

DEMO

Wrap-up

- A lot to think about, a lot of new things to learn
  - NullReferenceExceptions not 100% avoided
  - Is it all worth it?

Wrap-up

- (Non-) Nullable Reference types are awesome!
  - Null-Checks ‘public entry points’
  - Null-Checks after deserialization
  - Selectively disable warnings (but try to avoid)

  - Does help writing better, more robust, more maintainable code
  - Moving runtime issues to build time issues
  - Intent is much more clear

Call to action

- Green field projects
  - <nullable>enable</nullable>
  - Treat Warnings as errors
    - Rule set (null-related warnings as errors)
- Existing projects
  - <nullable>warnings</nullable>
  - Gradually fix warnings

Resources

- Rule set (null-related warnings as error)
  - GitHub Gist: https://bit.ly/3mQRwv1
- Demos (available soon)
  - GitHub: https://bit.ly/3guJZ4b

The solution

Nullable and non-nullable Reference Types

MORE DEMOs

- More attributes
- Generics
- Integration with EF