C# and Monads

Bangkok .NET User Group
Will Fuqua
2017-11-21

This Presentation

✓ Topics:

- What is a monad?
- Why use monads?
- C# language features and monads

✗ Not a topic:

- Formal definitions
- How to sneak monads into your code at work

A monad is a container type
that represents an
operation for chaining
its instances.

a monad is a container type...

```
IEnumerable<int> enumerableA =
    Enumerable.Range(1, 3); // 1, 2, 3

IEnumerable<int> enumerableB =
    Enumerable.Range(11, 3); // 11, 12, 13
```

...that represents an operation for chaining its instances

```
// enumerableA is 1, 2, 3
// enumerableB is 11, 12, 13

IEnumerable<(int, int)> pairs =
    from a in enumerableA
    from b in enumerableB
    select (a, b);

// (1, 11) (1, 12) (1, 13) (2, 11)...
```

IEnumerable<T>

is a container type

that represents

cartesian join chaining

of IEnumerable<T> instances

code

Nullable<T>

is a container type

that represents

“stop if null” chaining

of Nullable<T> instances

We've seen what
a monad is.

But what problem
is it solving?

We love generics!

- IEnumerable<T> makes many T
- Nullable<T> makes T optional
- Task<T> makes T asynchronous

… we're giving T superpowers!

We love function composition!

```
int GetIndexOfChar(string s, char c)
char GetCharAtIndex(string s, int i)
```

Given a string and a character, how do we return the next character in the string?

```
char GetNextCharacter(string s, char c) =>
GetCharAtIndex(s, GetIndexOfChar(s, c) + 1)
```

And here's the problem...

Generics do not
work well with composition

```
int? GetIndexOfChar(string s, char c)
char? GetCharAtIndex(string s, int i)
```

```
GetCharAtIndex(s, GetIndexOfChar(s, c) + 1)
```

Compile error!

How can we use generics,
but still be able to
compose functions?

spoiler: monads

How to write a monad

Implement two functions: Unit and Bind

- Unit
- Bind

Optional:

- Implement SelectMany so it works with LINQ

How to write a monad

We can create two simple functions:

```
Nullable<T> Unit<T>(T value)
```

wraps a 'T' value

```
Nullable<TResult> Bind<T, TResult>(
    this Nullable<T> target,
    Func<T, Nullable<TResult>> action)
```

Applies 'action' inside 'target' and returns a flattened object

How to write a monad

```
/// wraps a T value
Nullable<T> Unit<T>(T val) ⇒ new Nullable<T>(val);

/// applies action and returns a flattened value
Nullable<TResult> Bind<T, TResult>(
        this Nullable<T> target,
        Func<T, Nullable<TResult>> action) ⇒
    target.HasValue
        ? action(target.Value)
        : null;
```

```
int? GetIndexOfChar(string s, char c)
char? GetCharAtIndex(string s, int i)
```

We use these functions like this:

```
Nullable<char> result = Unit('H');

Nullable<char> result = GetIndexOfChar(str, 'H')
    .Bind(index ⇒ GetCharAtIndex(str, index + 1));
```

```
int? GetIndexOfChar(string s, char c)
char? GetCharAtIndex(string s, int i)
```

After implementing the LINQ Query Pattern:

```
Nullable<char> result =
    from index in GetIndexOfChar(str, 'H')
    from nextChar in GetCharAtIndex(str, index + 1)
    select nextChar ;
```

Optional: LINQ Query Pattern

```
Nullable<TResult> SelectMany<TSource, TResult>(
        this Nullable<TSource> source,
        Func<TSource, Nullable<TResult>> projector) ⇒
  Bind(source, projector);

Nullable<TResult> SelectMany<TSource, TMiddle, TResult>(
        this Nullable<TSource> source,
        Func<TSource, Nullable<TMiddle>> projector,
        Func<TSource, TMiddle, TResult> selector) ⇒
  source.Bind(source ⇒ projector(source)
              .Bind(result ⇒ Unit(selector(source, result))));
```

more code: three monads

Nullable / Optional

C# special syntax:

```
string sentence = "Hello world!";
char charToFind = 'H';

Nullable<char> result = (charToFind
    .GetIndexOfCharacter(sentence) + 1)
    ?.GetCharacterAtIndexExt(sentence);
```

Using monads instead:

```
string sentence = "Hello world!";
Nullable<char> charToFind = NullableMonad.Unit('H');

Nullable<char> nextChar = charToFind
    .Bind(chr => GetIndexOfCharacter(sentence, chr))
    .Bind(index => GetCharacterAtIndex(sentence, index + 1));
```

Error Handling

C# special syntax:

```
string sentence = "Hello World";
char charToFind = 'H';
try
{
    int index = GetIndexOfCharacterException(sentence, charToFind);
    int nextChar = GetCharacterAtIndexException(sentence, index + 1);
    return nextChar;
}
catch (Exception e)
{
    // e.g. do some logging
    throw;
}
```

Using monads instead:

```
string sentence = "Hello World";
Result<char> charToFind = ResultMonad.Unit('H');

Result<char> nextChar = charToFind
    .Bind(chr => GetIndexOfCharacter(sentence, chr))
    .Bind(index => GetCharacterAtIndex(sentence, index + 1));
```

Async

C# special syntax:

```
Task<string> input = Task.FromResult("24");

string number = await input;
int parsed = await ParseIntAsync(number);
int divided = await DivideByAsync(parsed, 2);
```

Using monads instead:

```
string sentence = "Hello world!";
Task<char> charToFind = AsyncMonad.Unit('H');

Task<char> divided = charToFind
    .Bind(chr => GetIndexOfCharacter(sentence, chr))
    .Bind(index => GetCharacterAtIndex(sentence, index));
```

The main point of monads

- This same pattern occurs in many different places:
  - Combining lists
  - Nullable (optional) values
  - Asynchronous computation
  - Error handling
  - Managing state and side effects
  - Parsing – Sprache
  - and many more, see the language-ext project

Next steps

- Read Eric Lippert's blog post series: Monads
- Play with the louthy/language-ext library
- Explore category theory in C# with Dixin Yan's Category Theory via C#

Thanks!

Downside of C# Monads

- No higher-kinded types
  - We have “parameterized types”
  - But not “parameterized generics”

- No type inference for function return types or parameters

- Not idiomatic C#

Instructions for use

Open this document in Google Slides (if you are at slidescarnival.com use the button below this presentation). **You have to be signed in to your Google account**

EDIT IN GOOGLE SLIDES

Go to the ***File*** menu and select ***Make a copy***.

You will get a copy of this document on your Google Drive and will be able to edit, add or delete slides.

EDIT IN POWERPOINT®

Go to the ***File*** menu and select ***Download as Microsoft PowerPoint***. You will get a .pptx file that you can edit in PowerPoint.

Remember to download and install the fonts used in this presentation (you'll find the links to the font files needed in the Presentation design slide)

More info on how to use this template at www.slidescarnival.com/help-use-presentation-template

Instructions for use

Open this document in Google Slides (if you are at slidescarnival.com use the button below this presentation). **You have to be signed in to your Google account**

EDIT IN GOOGLE SLIDES

Go to the ***File*** menu and select ***Make a copy***.

You will get a copy of this document on your Google Drive and will be able to edit, add or delete slides.

EDIT IN POWERPOINT®

Go to the ***File*** menu and select ***Download as Microsoft PowerPoint***. You will get a .pptx file that you can edit in PowerPoint.

Remember to download and install the fonts used in this presentation (you'll find the links to the font files needed in the Presentation design slide)

More info on how to use this template at www.slidescarnival.com/help-use-presentation-template

This is a slide title

- Here you have a list of items
- And some text
- But remember not to overload your slides with content

You audience will listen to you or read the content, but won’t do both.

You can also split your content

White

Is the color of milk and fresh snow, the color produced by the combination of all the colors of the visible spectrum.

Black

Is the color of coal, ebony, and of outer space. It is the darkest color, the result of the absence of or complete absorption of light.

In two or three columns

Yellow

Is the color of gold, butter and ripe lemons. In the spectrum of visible light, yellow is found between green and orange.

Blue

Is the colour of the clear sky and the deep sea. It is located between violet and green on the optical spectrum.

Red

Is the color of blood, and because of this it has historically been associated with sacrifice, danger and courage.

A picture is worth a thousand words

A complex idea can be conveyed with just a single still image, namely making it possible to absorb large amounts of data quickly.


Want big impact? **Use big image.**

And tables to compare data

	A	B	C
Yellow	10	20	7
Blue	30	15	10
Orange	5	24	16

89,526,124

Whoa! That’s a big number, aren’t you proud?

89,526,124$

That’s a lot of money

x

185,244 users

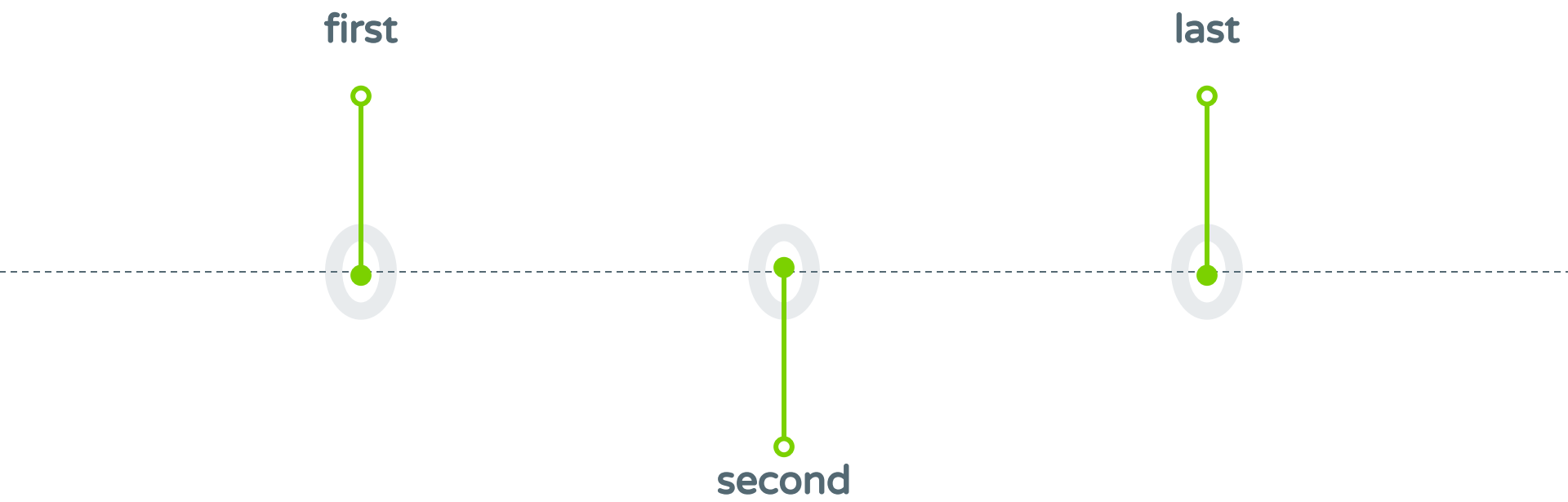
And a lot of users

x

100%

Total success!

Our process is easy



Let’s review some concepts

Yellow

Is the color of gold, butter and ripe lemons. In the spectrum of visible light, yellow is found between green and orange.

Blue

Is the colour of the clear sky and the deep sea. It is located between violet and green on the optical spectrum.

Red

Is the color of blood, and because of this it has historically been associated with sacrifice, danger and courage.

Yellow

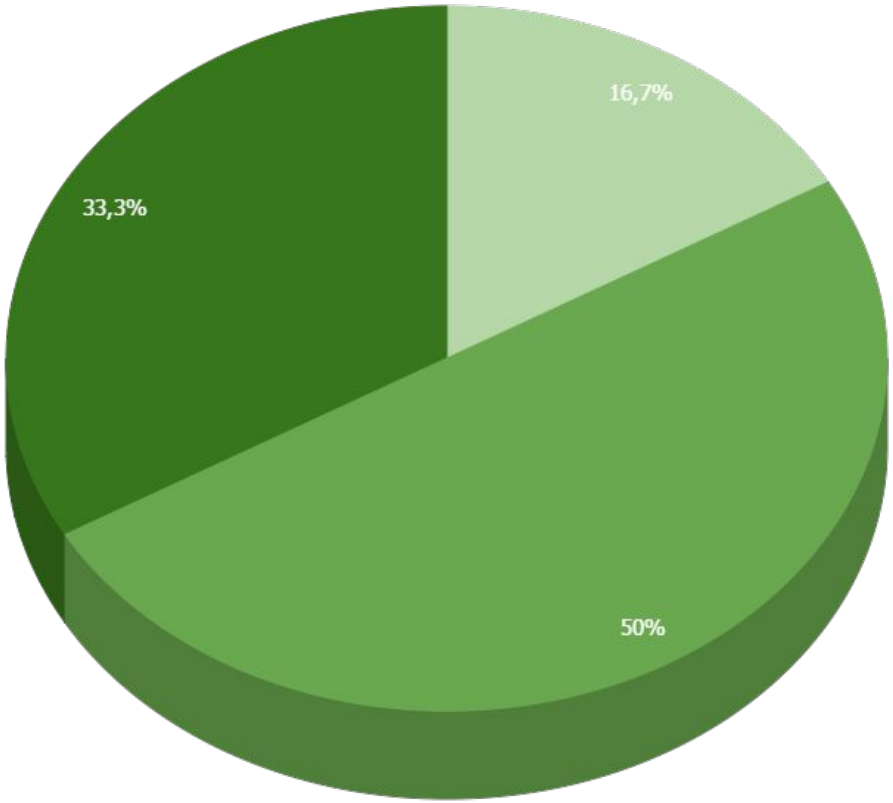
Is the color of gold, butter and ripe lemons. In the spectrum of visible light, yellow is found between green and orange.

Blue

Is the colour of the clear sky and the deep sea. It is located between violet and green on the optical spectrum.

Red

Is the color of blood, and because of this it has historically been associated with sacrifice, danger and courage.



You can copy&paste graphs from Google Sheets

Thanks!

Any questions?

You can find me at

@username & user@mail.me

Credits

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Unsplash

Presentation design

This presentation uses the following typographies and colors:

- Titles: **Varela Round**
- Body copy: **Varela Round**

You can download the fonts on this page:

https://www.google.com/fonts#UsePlace:use/Collection:Varela+Round

Click on the “arrow button” that appears on the top right

- Bright green **#7bd100**
- Gray **#546973**

You don’t need to keep this slide in your presentation. It’s only here to serve you as a design guide if you need to create new slides or download the fonts to edit the presentation in PowerPoint®

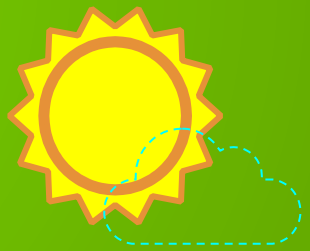
SlidesCarnival icons are editable shapes.

This means that you can:

- Resize them without losing quality.
- Change fill color and opacity.
- Change line color, width and style.

Isn't that nice? :)

Examples:



Now you can use any emoji as an icon!
And of course it resizes without losing quality and you can change the color.

How? Follow Google instructions
https://twitter.com/googledocs/status/730087240156643328