Computation Expressions Workshop

The Computations Expressions Workshop collects and presents the content of several papers and presents the material as a set of tutorials. This content is intended to be used in a workshop setting.

Setting up

The workshop uses the `dotnet` CLI as a base case. However, attendees may elect to use an editor of their choice, though the steps to use those tools will be omitted from the tutorials.

Install `dotnet`

- Install the `dotnet` CLI through the .NET Core SDK

NOTE: Some IDEs will also install the `dotnet` CLI.

`git clone` Workshop Repository

Clone the workshop repository so you have the materials available locally. Each tutorial is also tagged so that you can reference the finished result if you get stuck. `git clone https://github.com/panesofglass/computation-expressions-workshop`

Create a New Expecto Project

Now that you have the basics, you can get started by creating a new project with Expecto.

- Install the Expecto template with `dotnet new -i Expecto.Template`
- Create a new project with `dotnet new expecto -lang fsharp`

Outline

Introduction

What?

1. Computation Expressions
2. Query Expressions

Why?

1. Reduce arrowhead pattern
2. Familiar syntax, e.g. `let` and `do` with extensions: `let!`, `do!`, `return`, etc.
3. Language integrated queries (LINQ), e.g. `query { for x in source do select x }`
4. Language extensions without macros

Exercises

1. Computation Expressions: `OptionBuilder`

    1. Without a computation expression
    2. `Return`
    3. `Bind`
    4. Side effects and `Delay`
    5. `Run`ning a delayed computation
    6. `do!` with a `unit` result using `Combine`
    7. `if ... then` without an `else`
2. Combining Results: `ChoiceBuilder`
3. CEs for Compuations: `StateBuilder`
    1. Without a computation expression
    2. Side effects passed into a computation
    3. Differences between "container" and "computation"
4. Sequences: `StackBuilder`
    1. `Yield` = `Return`
    2. `For` = `Bind`
    3. Differences in `Delay`
5. Error Handling, Disposal, and More
    1. `TryWith`
    2. `TryFinally`
    3. `Using`
    4. `While`
    5. `Quote`
6. Extending Computation Expressions
    1. Method Overloads
    2. Adding Custom Extensions
7. Query Expressions
8. Embedded DSLs
    1. NuGet
    2. IL
    3. Saturn
    4. Freya

Project

Wrap up