1 An Introduction to F#

F# is a functional-first programming language. That is, F# is a programming language which emphasizes the functional aspect of programming to a greater extent than other languages you may be familiar with such as C#, Visual Basic .NET, Python and JavaScript. You will already be quite familiar with functions: you met them in high school when you drew a plot of the function,

$$y = x^2$$

using your trusty Sharp EL 9300 graphing calculator or your favorite plotting program on the WWW (e.g. Desmos at https://desmos.com/calculator). Functions as opposed to objects are the first-class citizens of functional programming languages like F#. They allow one to think about and solve problems using features prominent in functional programming such as immutability, pipelining, partial application, type-inference, and strong typing. (Note that these features are not unique to F#, they exist in other functional programming languages, and indeed in ordinary imperative programming languages such as C#, Visual Basic .NET, Python and JavaScript).

F# was created by Don Syme who is an Australian Computer Scientist and Principal Researcher at Microsoft Research, Cambridge, UK. He is still very active in the F# community and makes contributions to the F# language to this day (March 24th 2021). You could call him the Grandfather of F# or the Supreme Leader of the F# community.

F# allows one to think about a particular programming problem or task in a clean and simple way. It has the clean simplicity of Python while retaining the performance and type safety found in languages such as C++ and C#. F# can be used to write small programs with specific goals set at the start to large systems with open-ended goals.

F# is fully open source and fully cross-platform. F# is integrated within Visual Studio and so can make full use of all the integrated development environment's (IDE's) features such as Intellisense, packages management, documentation browser, etc. Alternatively, The Visual F# Tools can be downloaded from https://github.com/Microsoft/visualfsharp to developer F# programs without an IDE. On top of that F# code can be run in Jupyter Notebooks hosted on Microsoft Azure Services.

I also use Visual Studio Code sometimes because it is cross-platform and has about a million different extensions from Markdown edting support to copy code/text with line numbers and everything else you can think of in-between.

1.1 Hello, World!

As is traditional when learning most programming languages we will begin by writing a “Hello, World!” program. To do so first download and install the .NET SDK. You search the Internet for instructions how to download and install the .NET SDK. You are also free to choose whichever OS you desire.

Open a command terminal and type in

```
dotnet new console -lang "F#" -o hello
```

`cd` into the directory hello which was created when you typed in the above command. Then, open the file `Program.fs` which was created when the directory was created. It should look something like:

```
// Learn more about F# at http://docs.microsoft.com/dotnet/fsharp

open System

// Define a function to construct a message to print
let from whom =
    sprintf "from %s" whom

[<EntryPoint>]
let main argv =
    let message = from "F#" // Call the function
    printfn "Hello world %s" message
    0 // return an integer exit code
```

Replace the contents of the file with:

```
printfn "Hello, World!"
```

Now go back to the terminal and type in:

```
dotnet run
```

Pretty cool! Right? One line of code is all that's needed to print out a message to the console and a one line command is all you need to build and run the executable that prints out the message. That's a lot better than C or C++ and just as good as Python or any other scripting language.

1.1.1 But Wait!

I know what you're thinking!: There is something wrong here! Where is the function. There is a `printfn` function being called here which is taking a single string parameter as input and then outputting that string parameter to the console. But where is our function? Isn't F# supposed to be a functional-first programming language where nearly everything is a function? That is true. But in this case to make things easier like scripting, the F# compiler is creating a "main" function for us which is then called by the .NET runtime. To make this more explicit we can wrap the "Hello, World" code inside of a function ourselves. So let's do that.

Change the code

1.2 The dotnet Command

Your should familiarize yourself with the `dotnet` command as you will be using it a lot. You can get help by typing in `dotnet -h` and also by searching and reading the Microsoft documentation online. Here are a few exercises that you can try after you've familiarized yourself with the `dotnet` command.

1.2.1 Exercises

1. What is the default build configuration?
2. In which directory is the output stored?
3. What (SDK) command do you need to use to clean the output inside the build directory? Which files/directories still remain after you've cleaned the

build directory?
4. What command do you need to type in to build a Release version of the "Hello, World!" application?
5. How do you run the Release version of the application.
6. How do you clean the Release build?

1.3 Summary

In this chapter your wrote your first F# program!: A simple "Hello, World!" program.

To do this you setup your system to write F# programs. You downloaded and installed the .NET SDK (if you didn't already have it installed) and then created a new console project. You then modified the code to print "Hello, World!" to the console.