Azure Serverless with F#

# Overview

The project builds a single page application (SPA) using the tools provided in the F# ecosystem. Goals are to have the majority of the app as client side javascript (compiled from F# using Fable <https://fable.io/>) and a “serverless” server-side hosted on Azure Functions <https://azure.microsoft.com/en-us/services/functions/>

# Dev Environment

Visual Studio Code with the following extensions:

* Azure Account (required for Azure Functions)
* Azure Functions
* C#
* Ionide-FAKE
* Ionide-fsharp
* Ionide-paket
* Debugger for Chrome / Firefox

Font: <https://github.com/tonsky/FiraCode/>

Dotnet core SDK <https://dotnet.microsoft.com/download/dotnet-core/>

Node js <https://nodejs.org/en/download/>

Azure functions core tools <https://github.com/Azure/azure-functions-core-tools>

Git <https://www.git-scm.com/>

# Project Structure

## Scaffold files

.gitattributes

<https://www.git-scm.com/docs/gitattributes> (copied from Feliz template) Defining attributes per path

.gitignore

<https://www.git-scm.com/docs/gitignore> (copied from Feliz template) Specifies intentionally untracked files to ignore

package.json

Stores all javascript packages. Create with npm init

Use npm install to install javascript packages into the node\_modules folder. Note node modules with fable bindings should be installed with femto

Use npm install -D to install javascript packages into the node\_modules folder

## Dotnet Tools

dotnet new tool-manifest

Creates ".\.config\dotnet-tools.json"

dotnet tool install fable

<https://fable.io/>

Installs Fable (F# -> js compiler)

dotnet tool install femto

<https://github.com/Zaid-Ajaj/Femto>

Installs Femto package manager (combines paket, dotnet package manager, with npm, js package manager)

dotnet tool restore

Restores the tools making them available for use

## Public folder

Stores files to be bundled and made available from azure functions

e.g. Favicons for the app

## Src\shared folder

All source code for the app’s operation shared between client and server

dotnet new classlib -lang F#

## Src\client folder

All source code for the app’s operation on the client. All code here will be compiled to js only and run on the browser.

dotnet new classlib -lang F#

Install client packages (femto also installs npm packages to package.json)

dotnet femto install Feliz.MaterialUI

dotnet femto install Fable.DateFunctions

dotnet add package Feliz.Router

## Src\server folder

All source code for the app’s operation on the server (Azure functions)

func init

Change the create csproj to fsproj. Delete .gitignore

Note func new --language F# Doesn’t work

dotnet new func -lang F#

dotnet new http -lang F# --name TestFunc

<https://github.com/Azure/azure-functions-core-tools/issues/1663#issuecomment-786104570>

## Tests folder

All source code for testing

Js side testing <https://github.com/Zaid-Ajaj/Fable.Mocha>

.Net side testing <https://github.com/haf/expecto>

## Solution file

dotnet new sln

<https://docs.microsoft.com/en-us/dotnet/core/tools/dotnet-sln>

Stores all of the project files

## Auto generated

* .ionide folder
* node\_modules folder

Material Icons <https://material.io/resources/icons/?search=user&style=baseline>