

# Stage 0: Empty Project Structure - Cheatsheet

---

## Overview

**Goal:** Establish working F# + C# solution structure

**Branch:** `demo/stage-0-blank`

**Time:** Pre-presentation setup (not shown to audience)

**Key Insight:** F# and C# interop seamlessly - set foundation for demo

---

## Project Structure

```
src/Live/
├── CustomerLive.sln           # Solution file
├── CustomerLiveDomain/       # F# Class Library
│   ├── CustomerLiveDomain.fsproj
│   └── Library.fs           # F# domain models
└── CustomerLiveDemo/         # C# Console App
    ├── CustomerLiveDemo.csproj
    └── Program.cs           # C# application logic
```

---

## Quick Setup Commands

```
# Navigate to worktree (if using build process)
cd src/Live

# Create solution
dotnet new sln -n CustomerLive

# Create F# class library
dotnet new classlib -lang F# -n CustomerLiveDomain

# Create C# console app
dotnet new console -n CustomerLiveDemo

# Add projects to solution
dotnet sln add CustomerLiveDomain/CustomerLiveDomain.fsproj
dotnet sln add CustomerLiveDemo/CustomerLiveDemo.csproj

# Add F# project reference to C# project
cd CustomerLiveDemo
dotnet add reference ../CustomerLiveDomain/CustomerLiveDomain.fsproj
cd ..

# Verify it builds
dotnet build
```

---

## Initial File Contents

### Library.fs (Minimal)

```
namespace CustomerLiveDomain

module Say =
    let hello name =
        printfn "Hello %s" name
```

### Program.cs (Minimal)

```
using System;

class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Customer Demo");
    }
}
```

---

## Verification Steps

### Build Success

```
dotnet build
# Should see: Build succeeded. 0 Warning(s). 0 Error(s).
```

### Run C# Console

```
dotnet run --project CustomerLiveDemo
# Should output: Customer Demo
```

### Verify F# Reference

```
// In Program.cs, add at top:
using CustomerLiveDomain;
```

```
// In Main:  
Say.hello("F#");  
// Should compile and print: Hello F#
```

---

## Key Points for Presentation Setup

- ☑ **F# Library + C# Console** - Common enterprise pattern
  - ☑ **Clean slate** - No domain logic yet
  - ☑ **Verifies tooling works** - Catch issues before presenting
  - ☑ **Reference works** - C# can call F# code
- 

## Git Branch Management

```
# Create branch  
git checkout main  
git checkout -b demo/stage-0-blank  
  
# Commit initial structure  
git add .  
git commit -m "Stage 0: Empty project structure"  
  
# Push to remote  
git push -u origin demo/stage-0-blank
```

---

## Troubleshooting

### Build Fails: SDK Not Found

```
# Check .NET SDK  
dotnet --version  
# Should be 6.0 or higher  
  
# Install if needed  
# Download from https://dotnet.microsoft.com/download
```

### F# Template Not Found

```
# Install F# templates  
dotnet new install Microsoft.DotNet.Common.ItemTemplates
```

### Reference Not Working

```
# Verify reference in .csproj
cat CustomerLiveDemo/CustomerLiveDemo.csproj
# Should contain:
# <ProjectReference Include="..\CustomerLiveDomain\CustomerLiveDomain.fsproj" />
```

---

## Pre-Presentation Checklist

- ☐ Solution builds without errors
  - ☐ C# console runs
  - ☐ F# library referenced correctly
  - ☐ Can call F# code from C# (test with Say.hello)
  - ☐ Branch pushed to origin (backup)
  - ☐ Ready to proceed to Stage 1
- 

## Note for Presenter

**This stage is NOT shown during the presentation.** It's the foundation you'll start from when checking out [demo/stage-1-naive-csharp](#) for the live demo.

Purpose: Ensures the project structure is ready and working before the presentation begins.