Data-driven Coding and Patterns

Simon Robinson
Software Developer

@TechieSimon www.SimonRobinson.com

Data-driven Code

Program logic depends on the data being used

Example: Business Rules (Charging a Customer)


```
switch (orderDetails)
{
    // lots of cases
```

C# Patterns

Special C# syntax to test if an instance satisfies a (possibly complex) condition

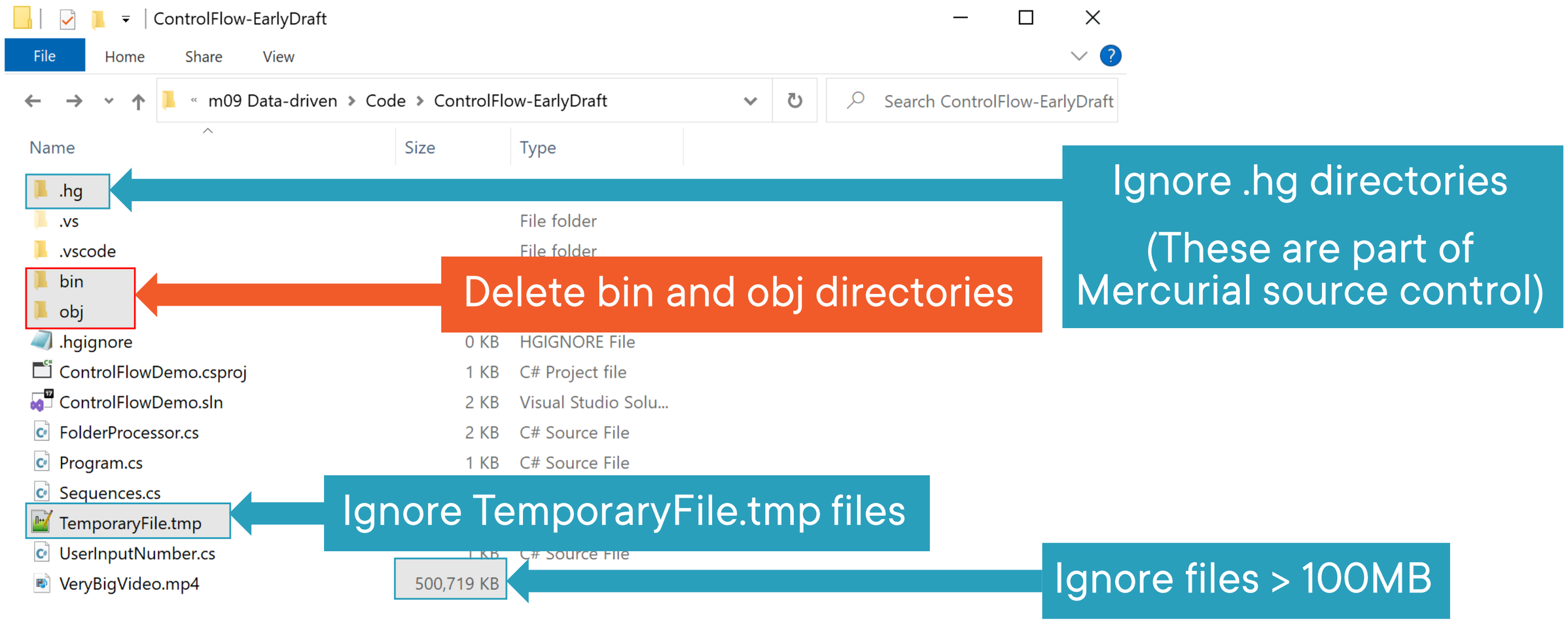
Allowed in:

- `is` clauses
- `switch` statements
- `switch` expressions

Demo: Backup App

App to backup files, but can also clean up to preserve disc space

The Business Rules for this App:



Demo

Create the backup/cleanup app

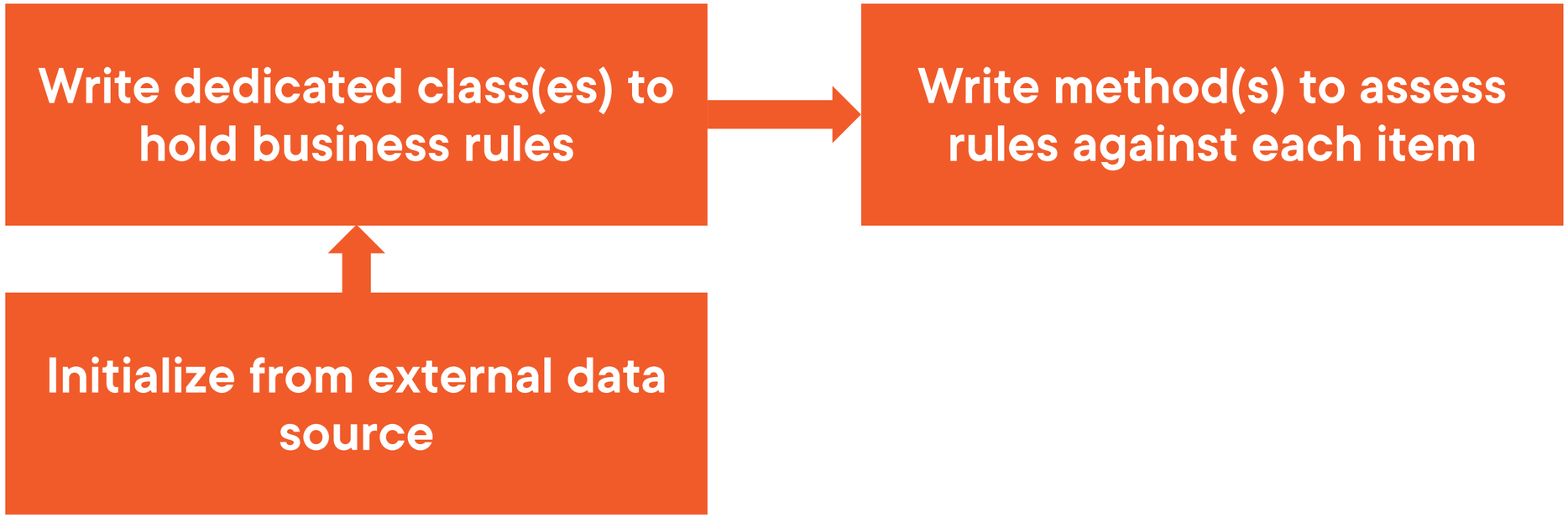
- Implement the business rules using C# patterns and `switch`

Demo

Import rules from a file (so they aren't hard-coded)

- To allow changing the rules without re-compiling
- This will prevent using C# patterns

Write Data-driven Code:



Items were file system objects in the demo,
but might equally be clients or orders etc.

Summary

Data-driven code

- Patterns are great, but only if the data is constant
- If data is loaded externally, you can't use patterns
- Create class(es) to hold business rules
- Write code to use whatever logic is required to test objects against business rules