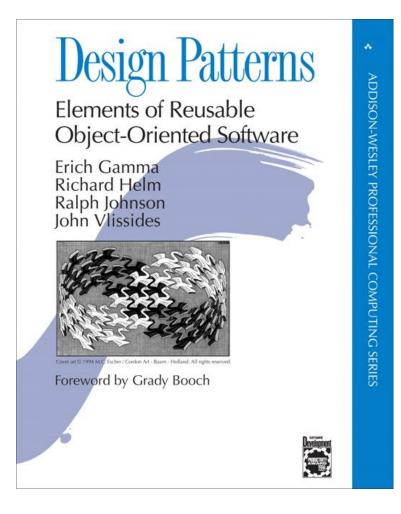
## Course Summary



**Gerald Britton**IT Solutions designer

@GeraldBritton www.linkedin.com/in/geraldbritton





**Creational Structural** 

**Behavioral** 

### SOLID Principles of Object-oriented Design

Single responsibility

Open-closed

Liskov substitution

Interface segregation

**Dependency** inversion



Don't Repeat Yourself Don't code it, copy it

Don't copy it, link to it

- Use Python modules

Don't link to it, load it

**DRY** 

Not limited to OOP



### Abstract Base Classes

#### awesome\_abc.py

```
import abc
class AwesomeABC(abc.ABC):
    @abc.abstractmethod
    def must_implement(self, value):
        pass
    def concrete_method(self, value):
        return f'Value is: {value}'
class ThisIsAwesome(AwesomeABC):
    def must_implement(self, value):
        return value * 42
```

### Other Design Patterns

**Parallel Functional** Asynchronous



# Thank you!

