InterBit
Training & Consulting Ltd.

Structural Design Patterns

Structural Patterns

InterBit Training & Consulting Ltd.

- Structural patterns describe how classes and objects can be **combined to form larger structures.**

- You're often required to choose between **Inheritance** and **Composition.**
  - Inheritance is usually simpler and requires less coding & less allocations.
  - Composition may allow much more flexibility.

Structural Patterns

- **Adapter**
  - force a class to conform to a new interface.
- **Bridge**
  - Separates abstraction from implementation so that both may vary (and evolve) separately.
- **Composite**
  - An object may consist of other objects – recursively.

Structural Patterns

InterBit Training & Consulting Ltd.

- **Decorator**
  - dynamically add functionality to objects.
- **Flyweight**
  - share objects to save allocations. This may require to remove state variables, exchanging them with parameters passed to methods.

- **Façade**
  - Simplify the access to a complex system.

Structural Patterns

InterBit
Training & Consulting Ltd.

- **Proxy**
  - The access to an object may be controlled by another object, with a similar interface.

Note

InterBit
Training & Consulting Ltd.

- During the following discussion, you may note that some patterns have **similar outlines**.
- However, those patterns will reflect **different problems.**
  - Different program evolution (maintenance).
  - Different emphasis points.