

Flyweight Pattern

Niraj Bhatt

Design Patterns Course

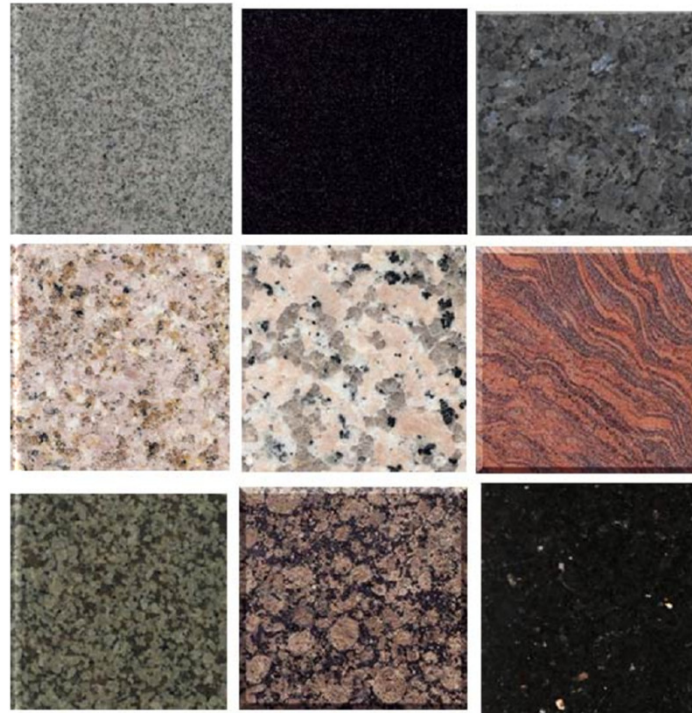


Intent

- Reduce storage costs for large number of objects
- Share objects to be used in multiple contexts simultaneously
- Retain object oriented granularity and flexibility

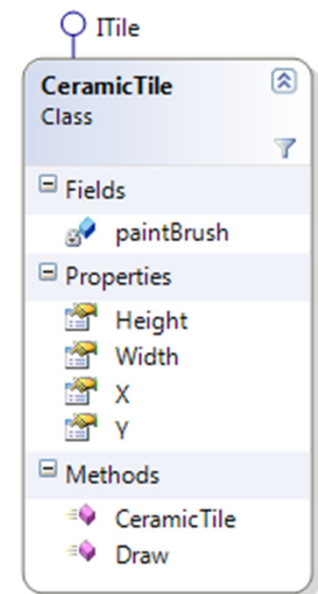
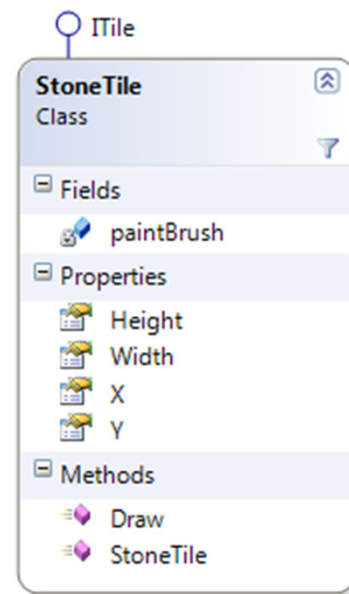
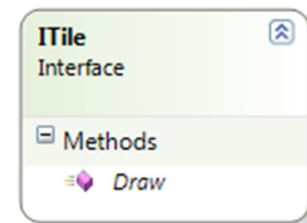
Motivating Example

- Interiors Design Application
- Tiles
 - Ceramic
 - Stone
 - Quarry

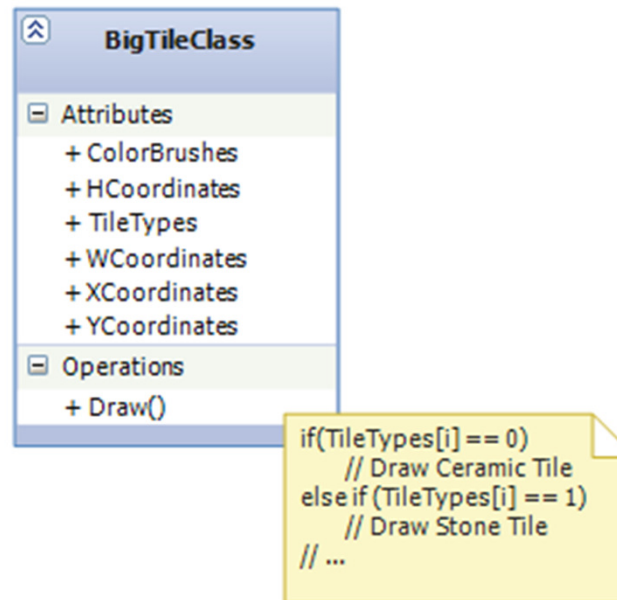


Motivating Example

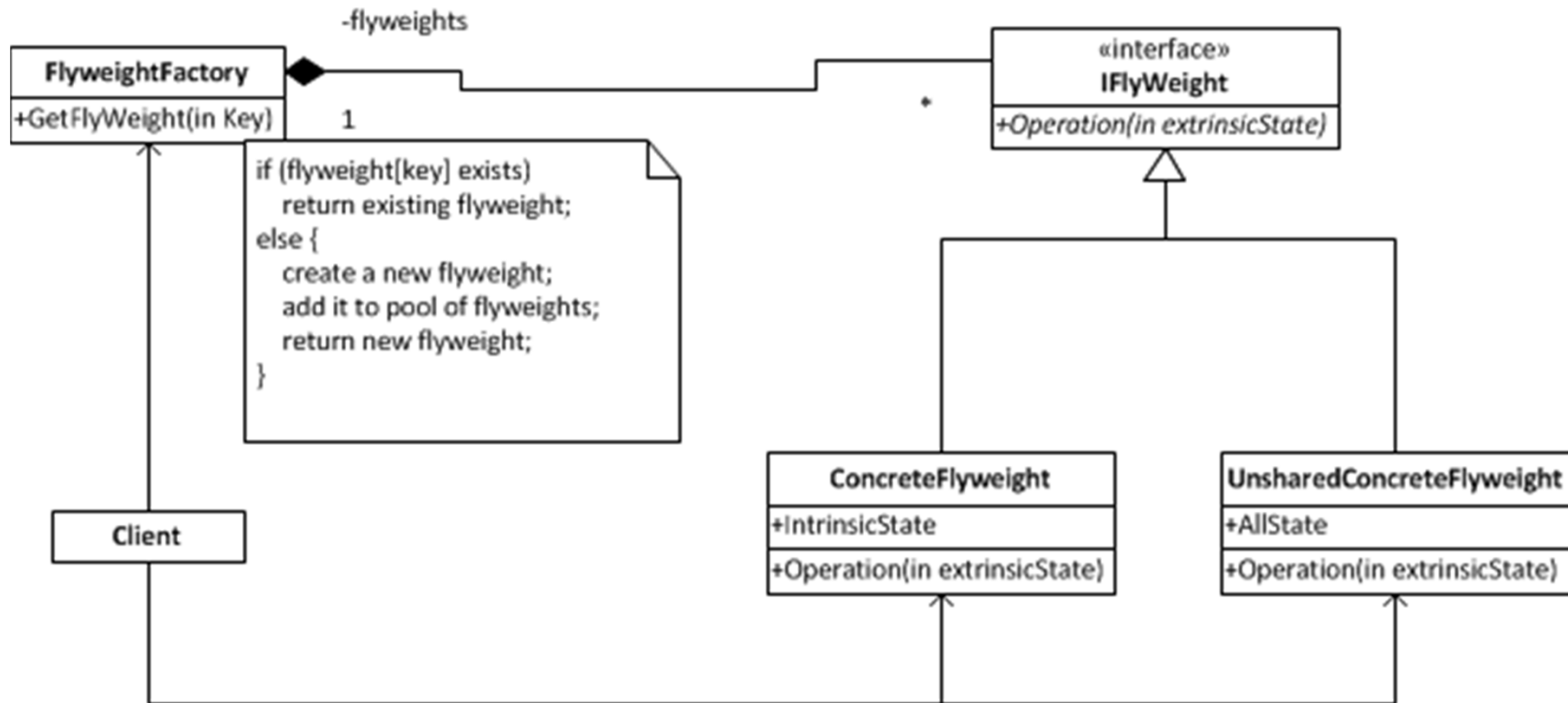
- One Object Per Tile
- E.g. 1,000,000 Tiles
= 1,000,000 Objects



Anti-Approach / Bad Design



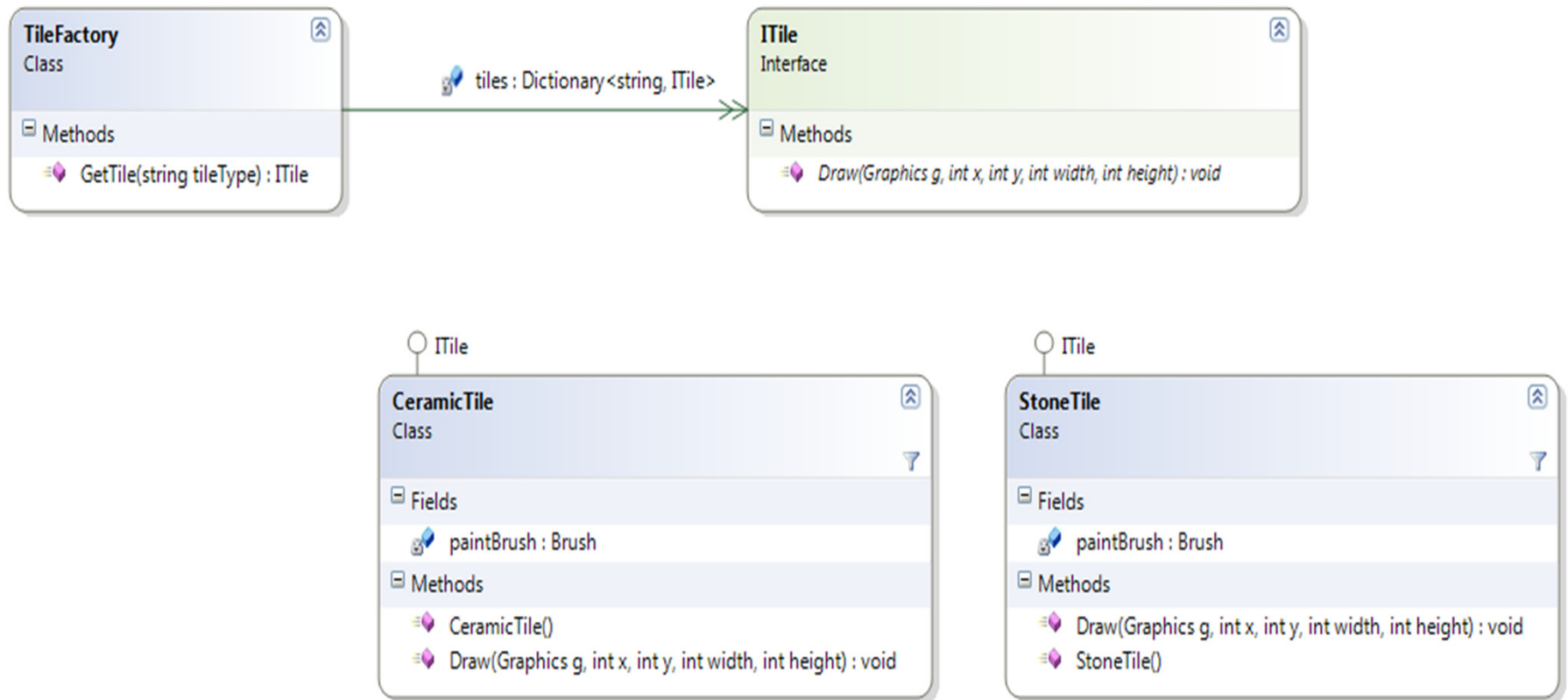
Structure



Consequences

- **Group of objects are replaced by few shared objects once extrinsic state is removed**
- **Storage savings are derived from**
 - Reduced instances
 - Amount of intrinsic state per object
 - Whether extrinsic state is computed or stored
- **Application can no longer depend on object's identity**

Implementation Example



Real World Examples

- **.NET String Class**

- String.Intern Method – returns a flyweight
- Used to reduce memory footprint, increase performance
- intern pool, contains a single reference to each unique string in program

- **Considerations**

- Interned strings are not released till CLR terminates
- To intern a string, you must first create (allocate memory) it

Related Patterns

- Composite
- State
- Strategy

Summary

- Use Flyweight pattern to
 - Share Objects for better program efficiency
 - Retain fine grained objects

For more in-depth **online** developer **training** visit



on-demand content from authors you **trust**

