

Flyweight Pattern

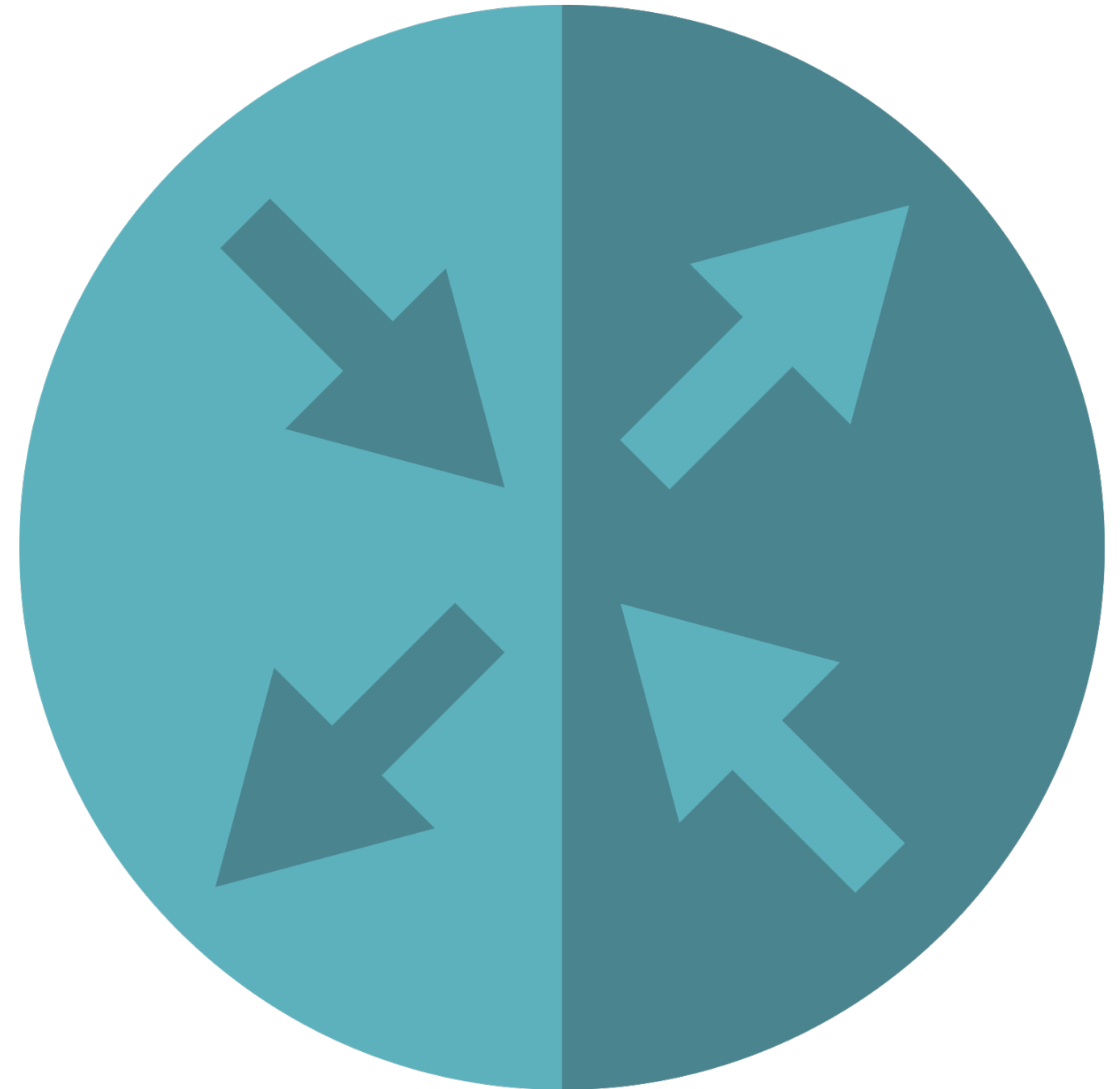


Bryan Hansen

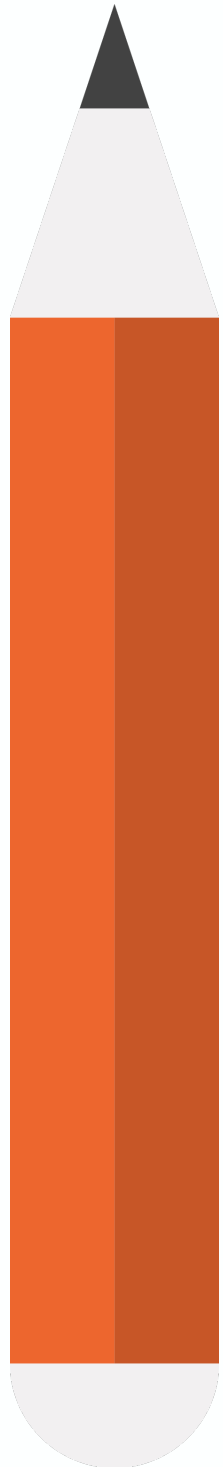
twitter: bh5k | <http://www.linkedin.com/in/hansenbryan>

Concepts

- More efficient use of memory
- Large number of similar objects
- Immutable
- Most of the object states can be extrinsic
- Examples:
 - `java.lang.String`
 - `java.lang.Integer#valueOf(int)`
 - `Boolean`, `Byte`, `Character`, `Short`, `Long`



Design



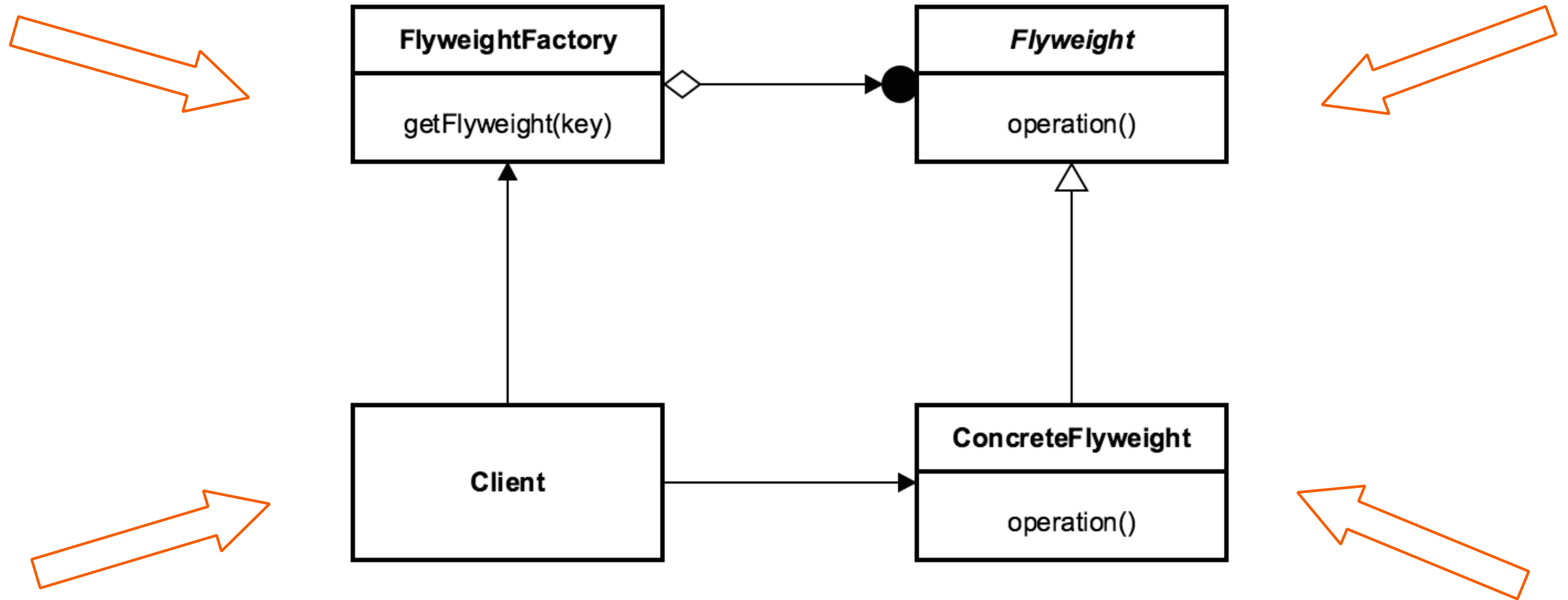
Pattern of patterns

Utilizes a Factory

Encompasses Creation and Structure

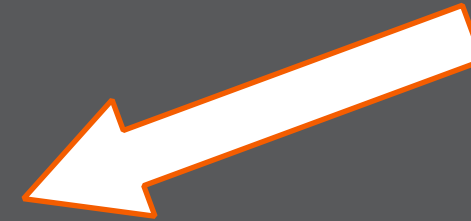
Client, Factory, Flyweight, ConcreteFlyweight

UML



Everyday Example - Integer

```
Integer firstInt = Integer.valueOf(5);
```



```
Integer secondInt = Integer.valueOf(5);
```

```
Integer thirdInt = Integer.valueOf(10);
```

```
System.out.println(System.identityHashCode(firstInt));
```

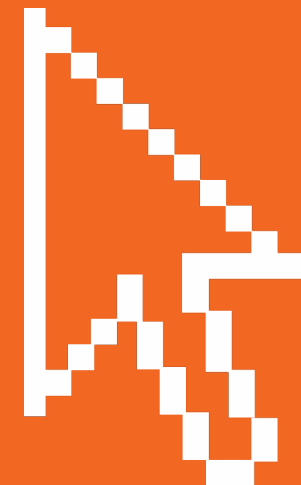
```
System.out.println(System.identityHashCode(secondInt));
```

```
System.out.println(System.identityHashCode(thirdInt));
```

Exercise Flyweight

Inventory Management System

Client, Catalog, Order, Item



Pitfalls

- Complex pattern
- Premature optimization
- Must understand Factory
- Not a graphical pattern



Contrast

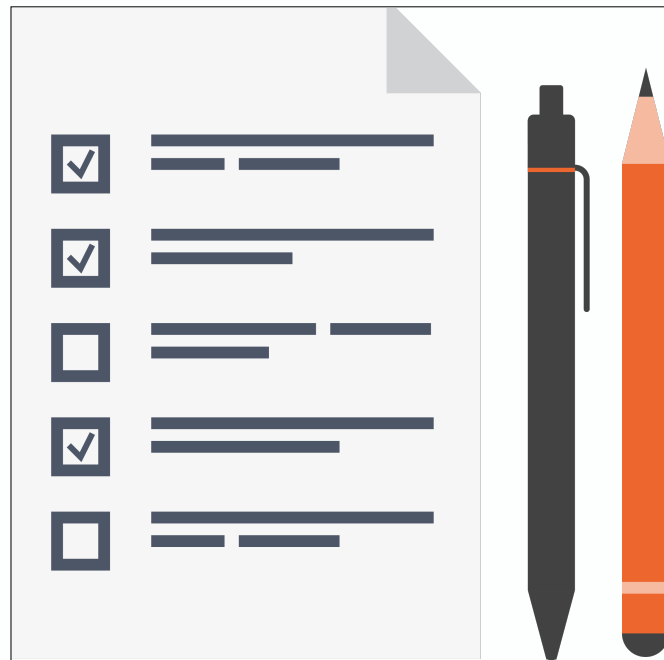
Flyweight

- Memory Optimization
- Optimization Pattern
- Immutable Objects

Facade

- Refactoring Pattern
- Simplified Client
- Provides a different interface

Flyweight Summary



- Great for Memory Management
- A little bit complex
- Used a lot by the core API