CS474 HW2 - Sherryl Mathew George

Building and running the code:

./gradlew clean && ./gradlew build	Build depending on whether you are on Linux or Windows
./gradlew clean && ./gradlew test	Run unit test cases
./gradlew clean && ./gradlew run	Run the application

Note: while running make sure current user has file read/write permissions on the cloned folder to successfully run the application

Modules:

- 1. *annotations* - Contains the main annotation interface file
- 2. *annotations-proc* - Contains the annotations processor with all the required dependencies to run this annotation as a stand alone module
- 3. Implementation1 - Simple generic list implementation which uses the annotations
- 4. Implementation2 - Binary tree implementation implementing two parallel iterators
- 5. Implementation3 - Social media spammer

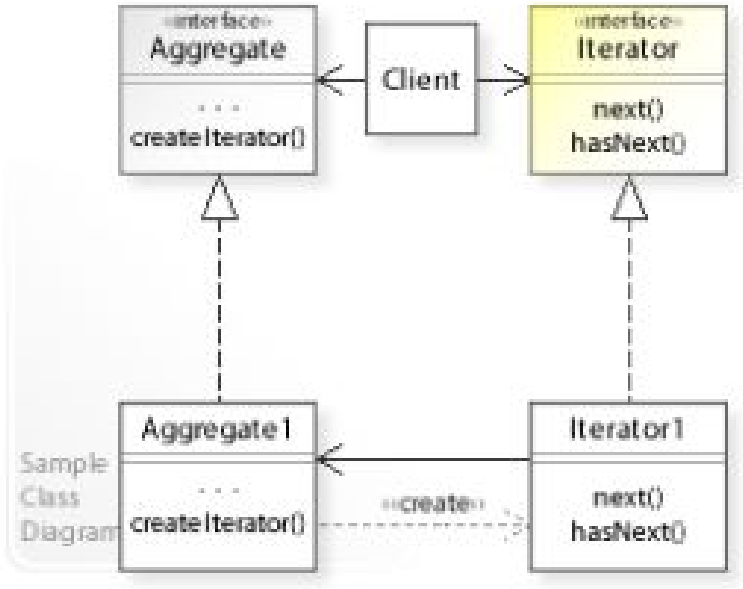
Note: All implementations are copied from the Internet from various sources as noted in each implementation. No modifications have been made other than adding annotations.

Deploying the Annotation processor in other projects:

1. Copy the modules *annotations* and *annotations-proc* over to the new project
2. Add the below two lines to the build to gradle and you will have the annotation processor ready to be used in the new project:

    compile project(':annotations')
    annotationProcessor(project(':annotations-proc'))

Iterator Pattern



Above is the basic implementation of the Iterator pattern. The key idea in this pattern is to take responsibility for access and traversal out of the list [aggregate] object and put it into an iterator object." [GoF, p257]

Aggregate

- Defines an interface for creating an Iterator object.

Aggregate1 or ConcreteAggregate

- Implement createIterator() by returning an instance of the corresponding iterator class (Iterator1 or ConcreteIterator).

Iterator

- Defines an interface for accessing and traversing the elements of an Aggregate object.

Iterator1 or ConcreteIterator

- Implement the Iterator interface.
- An iterator is usually implemented as an inner class of an aggregate class so that it can access the internal (private) data structures of the aggregate.

Above is the basic gist of the implementation that need to be carried out to have an Iterator Pattern.

Annotation Implementation

Annotation Type Hierarchy

- **annotations.IteratorPattern**
  - **annotations.IteratorPattern.Aggregate**
    - **annotations.IteratorPattern.Aggregate.IteratorFactory**
  - **annotations.IteratorPattern.ConcreteAggregate**
    - **annotations.IteratorPattern.ConcreteAggregate.IteratorFactory**
    - **annotations.IteratorPattern.ConcreteIterator**
  - **annotations.IteratorPattern.Iterator**
    - **annotations.IteratorPattern.Iterator.hasNext**
    - **annotations.IteratorPattern.Iterator.next**

Annotation Types & Usage

IteratorPattern

Annotations for the Iterator Pattern All annotations are enforced during compile time only

IteratorPattern.Aggregate

Annotation for Aggregate interface or abstract class. Rules are enforced in *AnnotationProcessor.checkAggregateAnnotation*

IteratorPattern.Aggregate.IteratorFactory

Annotates an iterator factory method inside and Aggregate. Should be applied only to methods

IteratorPattern.ConcreteAggregate

Annotates the concrete aggregate inside an Iterator pattern. The ConcreteAggregate has two more annotations as defined in the hierarchy above. Rules are enforced in *AnnotationProcessor.checkConcreteAggregateAnnotation*

IteratorPattern.ConcreteAggregate.IteratorFactory

For the factory method inside ConcreteAggregate.

Parameters:

- returnType(Class) - Mandatory, Provides the return type that is to be returned by the IteratorFactory method

IteratorPattern.ConcreteIterator

Annotation for Iterator concrete iterator class. Rules are enforced in AnnotationProcessor.checkConcreteIteratorAnnotation.

IteratorPattern.Iterator

Annotation for Iterator interface. Rules are enforced in **AnnotationProcessor.checkAIteratorAnnotation**

Parameters:

- returnType(Class, optional) - Should be used to specify the return type of the next() method in case a non-generic return type is being used
- typeVariableReturnIndex(int, optional) - Should be used to specify the index of the type variable that is used in the next() method in the order it is specified in the Iterator interface signature.

IteratorPattern.Iterator.hasNext

Annotation for hasNext() method inside the Iterator Enforced in the same method as for Iterator.

IteratorPattern.Iterator.next

Annotation for next() method inside the Iterator Enforced in the same method as for Iterator.

Rules of Implementation

IteratorPattern.Aggregate

- Can be applied to only abstract class or interface
- Should have at least one IteratorFactory annotated method within the interface/class declaration
- Has limited support to support generic types as of now will work only with one generic parameter passed to the Aggregate interface
- Can be applied to only classes or interfaces

IteratorPattern.Aggregate.IteratorFactory

- At least one annotation has to be present inside an Aggregate annotation.
- Should be enclosed inside the Aggregate annotation
- Cannot be static or private
- The return type of the method should be assignable to Iterator interface both in case of generic and non-generic case.
- Can be applied to only methods

IteratorPattern.ConcreteAggregate

- Can be used only for classes
- Cannot be used for abstract classes or interfaces
- Should have at least one method annotated with ConcreteAggregate.IteratorFactory
- Should have at least one class annotated with IteratorPattern.ConcreteIterator
- Inherits from the type annotated with IteratorPattern.Aggregate

IteratorPattern.ConcreteAggregate.IteratorFactory

- Should be enclosed within IteratorPattern.ConcreteAggregate
- Has a mandatory parameter *returnType* of type Class. Should specify the return type of the method and the same should be assignable to the type annotated with IteratorPattern.Iterator
- Can be applied to only methods
- Method cannot be private or static
- Method should be annotated with @Override also

IteratorPattern.ConcreteIterator

- Should be enclosed within IteratorPattern.ConcreteAggregate
- Inherits from the type annotated with IteratorPattern.Iterator
- Can be applied to only classes
- Cannot have modifiers abstract
  Should be a private class

IteratorPattern.Iterator

- Can be applied to only interfaces
- Has two parameters, 1. returnType(optional) - Should be used to specify the return type of the next() method in case a non-generic return type is being used
  2. typeVariableReturnIndex - Should be used to specify the index of the type variable that is used in the next() method in the order it is specified in the Iterator interface signature.
- Should have IteratorPattern.Iterator.hasNext on a methods
- Should have IteratorPattern.Iterator.next on a method

IteratorPattern.Iterator.hasNext

- Should be annotated inside IteratorPattern.Iterator
- Can be applied only to methods
- Shouldn't be private or static
- Return type needs to be boolean

IteratorPattern.Iterator.next

- Should be annotated inside IteratorPattern.Iterator
- Can be applied only to methods
- Shouldn't be private or static
- Return type needs to match with what is declared as per the IteratorPattern.Iterator parameters.

Sample Output

Annotation processor will be run only when a build is performed on a changed code through gradle (i.e. gradle does not build unchanged code).
Running the annotation processor during the build results in the below logs being output for each of the implementations:

```
> Task :implementation1:compileJava
18:05:46.374 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing IteratorAnnotation - Starting
18:05:46.376 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing Annotation - Iterator
18:05:46.376 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing Annotation - hasNext
18:05:46.377 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing Annotation - next
18:05:46.377 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing AggregateAnnotation - Ending
18:05:46.377 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing AggregateAnnotation - Starting
18:05:46.377 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing Annotation - Aggregate
18:05:46.378 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing Annotation - IteratorFactory
18:05:46.378 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing AggregateAnnotation - Ending
18:05:46.378 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing ConcreteAggregateAnnotation - Starting
18:05:46.378 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing Annotation - ConcreteAggregate
18:05:46.379 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing Annotation - ConcreteIterator
18:05:46.379 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing Annotation - IteratorFactory
18:05:46.380 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing ConcreteAggregateAnnotation - Ending
18:05:46.380 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing ConcreteIterator - Starting
18:05:46.380 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Processing ConcreteIterator - Ending
18:05:46.381 [Task worker for ':' Thread 5] INFO  processor.AnnotationProcessor - Annotation Processing ended no more rounds to process
```

Errors if any are reported so:

```
> Task :implementation1:compileJava FAILED
19:55:52.559 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing IteratorAnnotation - Starting
19:55:52.563 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing Annotation - Iterator
19:55:52.565 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing Annotation - next
19:55:52.565 [Task worker for ':'] ERROR processor.AnnotationProcessor - Annotation violation - interface annotations.IteratorPattern$Iterator should have atleast one method annotated with interface annotations.IteratorPattern$
19:55:52.566 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing AggregateAnnotation - Ending
19:55:52.566 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing AggregateAnnotation - Starting
19:55:52.566 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing Annotation - Aggregate
19:55:52.567 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing Annotation - IteratorFactory
19:55:52.567 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing AggregateAnnotation - Ending
19:55:52.568 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing ConcreteAggregateAnnotation - Starting
19:55:52.568 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing Annotation - ConcreteAggregate
19:55:52.569 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing Annotation - ConcreteIterator
19:55:52.569 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing Annotation - IteratorFactory
19:55:52.570 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing ConcreteAggregateAnnotation - Ending
19:55:52.570 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing ConcreteIterator - Starting
19:55:52.570 [Task worker for ':'] INFO  processor.AnnotationProcessor - Processing ConcreteIterator - Ending
/home/sherryl/IdeaProjects/cs474.hw2/implementation1/src/main/java/implementation/Iterator.java:10: error: interface annotations.IteratorPattern$Iterator should have atleast one method annotated with interface annotations.Itera
public interface Iterator<T>{
                 ^
19:55:52.574 [Task worker for ':'] INFO  processor.AnnotationProcessor - Annotation Processing ended no more rounds to process
1 error
```

Unit Testing

Unit tests are written using testNG and Google compile-testing

Logging

Use log4j2 for logging into files and typesafe is used for logging.