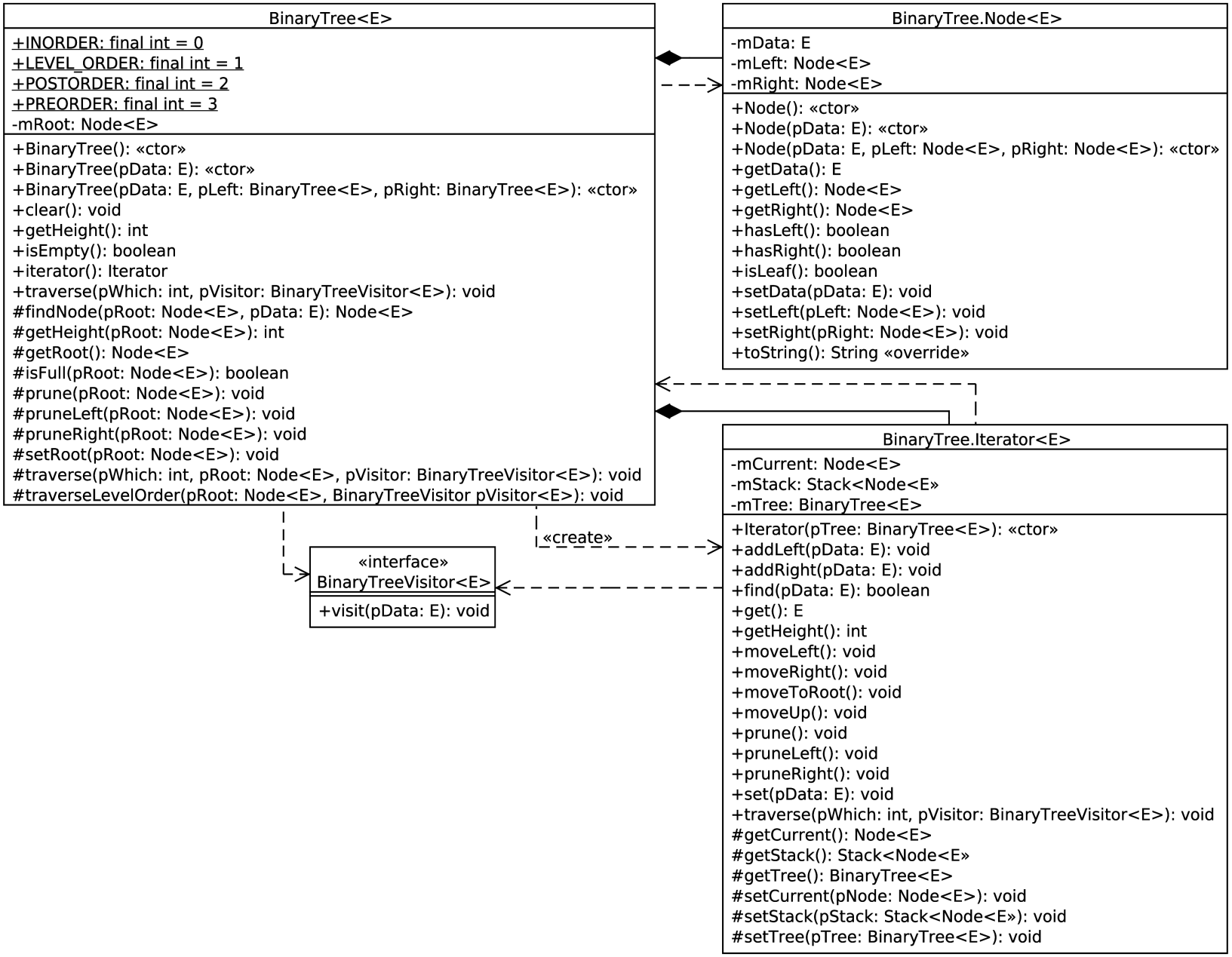
10. Trees :: Binary Trees :: Java Implementation :: UML Class Diagram



10. Trees :: Binary Trees :: *BinaryTree<E>* Class

Our *BinaryTree* class is a generic class in that we must specify a type parameter *E* (which can be any class or interface type) when instantiating *BinaryTree* objects. *E* specifies the data type of the data that will is stored in each node of our *BinaryTree*, i.e., *E* could be *Integer*, *String*, *Shape*, and so on.

To specify that a class is a generic class, we follow the class name by <> and inside the <> we specify the type parameter. It is a Java Class Library convention that *E* is used as the generic type identifier for elements of a collection:

```
public class BinaryTree<E> {
}
```

This will permit us to create *BinaryTree* objects which stores objects of any class or interface type:

```
BinaryTree<Integer> treeOfIntegers = new BinaryTree<>();
BinaryTree<String> treeOfStrings = new BinaryTree<>();
BinaryTree<Shape> treeOfShapes = new BinaryTree<>();
BinaryTree<Comparable> treeOfComparables = new BinaryTree<>();
```