Iterable.java (folder: java.base/java/lang)

```
/* Copyright (c) 2003, 2013, Oracle and/or its affiliates. All rights reserved.
 * ORACLE PROPRIETARY/CONFIDENTIAL. Use is subject to license terms. */
package java.lang;

import java.util.Iterator;
import java.util.Objects;
import java.util.Spliterator;
import java.util.Spliterators;
import java.util.function.Consumer;

/*Implementing this interface allows an object to be the target of the enhanced
 * {@code for} statement (sometimes called the "for-each loop" statement).
 *
 * @param <T> the type of elements returned by the iterator
 *
 * @since 1.5
 * @jls 14.14.2 The enhanced {@code for} statement
 */
public interface Iterable<T> {
    /**
     * Returns an iterator over elements of type {@code T}.
     *
     * @return an Iterator.
     */
    Iterator<T> iterator();

    /* Performs the given action for each element of the {@code Iterable}
     * until all elements have been processed or the action throws an
     * exception.  Actions are performed in the order of iteration, if that
     * order is specified.  Exceptions thrown by the action are relayed to the
     * caller.
     * <p>
     * The behavior of this method is unspecified if the action performs
     * side-effects that modify the underlying source of elements, unless an
     * overriding class has specified a concurrent modification policy.
     *
     * @implSpec
     * <p>The default implementation behaves as if:
```

```
     * <pre>{@code
     *     for (T t : this)
     *         action.accept(t);
     * }</pre>
     * @param action The action to be performed for each element
     * @throws NullPointerException if the specified action is null
     * @since 1.8
     */
    default void forEach(Consumer<? super T> action) {
        Objects.requireNonNull(action);
        for (T t : this) {
            action.accept(t);
        }
    }

    /* Creates a {@link Spliterator} over the elements described by this
     * {@code Iterable}.
     * @implSpec
     * The default implementation creates an
     * <em><a href="../util/Spliterator.html#binding">early-binding</a></em>
     * spliterator from the iterable's {@code Iterator}.  The spliterator
     * inherits the <em>fail-fast</em> properties of the iterable's iterator.
     *
     * @implNote
     * The default implementation should usually be overridden.  The
     * spliterator returned by the default implementation has poor splitting
     * capabilities, is unsized, and does not report any spliterator
     * characteristics. Implementing classes can nearly always provide a
     * better implementation.
     *
     * @return a {@code Spliterator} over the elements described by this
     * {@code Iterable}.
     * @since 1.8
     */
    default Spliterator<T> spliterator() {
        return Spliterators.spliteratorUnknownSize(iterator(), 0);
    }
}
```