Putting a final on field makes only the reference as final i.e it cannot point to another object. But the object itself can be modifiable.

1) Custom objects. Ex: Employee is immutable class. But if it has Address which is mutable then caller can change the object value. So to avoid this, on the getter return new object.

2) On date

3) On list

If you do not want your list to be changed return a copy of it:

public List<String> get() {

return new ArrayList<String>(this.list);

}

Or you can return a [unmodifiable list](http://stackoverflow.com/users/449325/alpian):

public List<String> get() {

return Collections.unmodifiableList(this.list);

}

/\*\*

    \* Date class is mutable so we need a little care here.

    \* We should not return the reference of original instance variable.

    \* Instead a new Date object, with content copied to it, should be returned.

    \* \*/

    public Date getMutableField() {

        return new Date(mutableDate.getTime());

    }