@Contract Annotations

In this section:

Syntax of the @Contract annotations

The `@Contract` annotation has two attributes: `value` and `pure`.

The `@Contract` annotation value has the following syntax:

```
contract ::= (clause ';')* clause
clause ::= args '->' effect
args ::= ((arg ',')* arg )?
arg ::= value-constraint
value-constraint ::= 'any' | 'null' | '!null' | 'false' | 'true'
effect ::= value-constraint | 'fail'
```

The constraints here are:

```
_ - any value
null - null value
!null - a value statically proved to be not-null
true - true boolean value
false - false boolean value
fail - the method throws exception, if the arguments satisfy argument constraints
```

The `pure` attribute is intended for the methods that do not change the state of their objects, but just return a new value. This attribute is either `false` (by default), or `true`.

Example

Consider the following code:

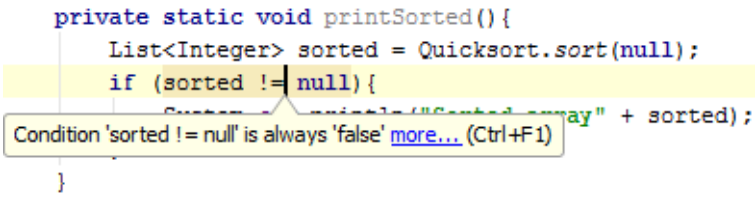
```
private static void printSorted(){
            List <Integer> sorted = Quicksort.sort(null);
            if (sorted != null){
                System.out.println("Sorted array" + sorted);
        }
    }

public static <T extends Comparable<T>> List<T> sort(List<T> list)
    {
        if(list != null){
            List<T> copy = new ArrayList<T>(list);
            sort(copy);
            return copy;
    }
        else {
            return null;
        }
    }
```

IntelliJ IDEA doesn't complain, because it doesn't know that a null input yields a null output.

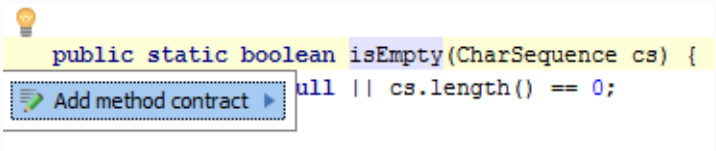
Let's decorate the `sort()` method with `@Contract` annotation, specifying that null inputs yield null outputs.

```
@Contract("null -> null")
public static <T extends Comparable<T>> List<T> sort(List<T> list)
{
    if (list != null){
        List<T> copy = new ArrayList<T>(list);
        sort(copy);
        return copy;
    }
    else {
        return null;
    }
}
```

IntelliJ IDEA immediately recognizes that `if` statement is extraneous, and reports about the condition that is always false:



IntelliJ IDEA suggests a quick fix for the methods of the library classes:



See Also

Procedures:

- Code Inspection
- Intention Actions

Web Resources:

- Developer Community