

# Restoration of Molecular Artifacts

## The new structure checker framework of Standardizer

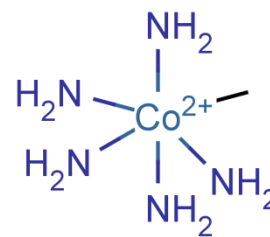
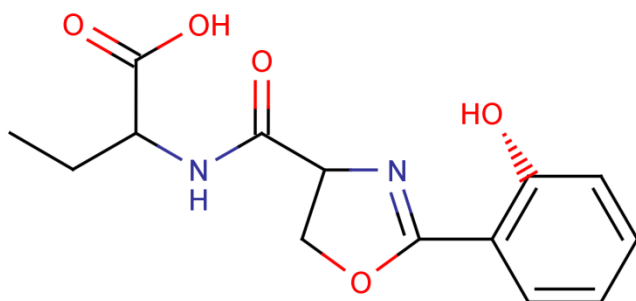
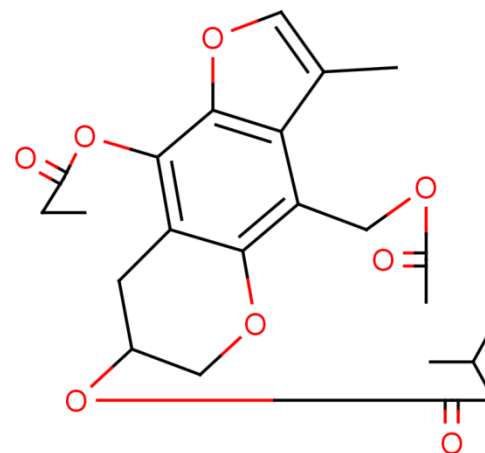
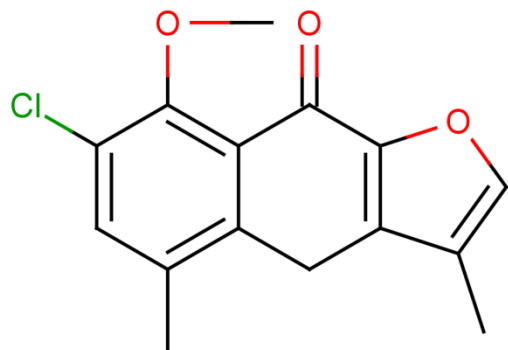
Attila Szabó, György Pirok



**ChemAxon**  
Solutions for Cheminformatics

# The Problem

Molecules registered in databases often contains drawing errors or unpreferred representations. Some other issues are originated from custom workarounds due to the limitations of older chemistry applications.



$\text{No}^{3-}$

$\text{No}^{3-}$

# The Solution – Structure Checker

Many common errors can be identified by a new software tool called Structure Checker. Some of them can be fixed automatically, others can be displayed for manual correction. Structure Checker first appeared in the form of API and in MarvinSketch to highlight specific structural problems.

The image displays the MarvinSketch 5.3.2 interface with the Structure Checker tool. The main window shows a chemical structure with two red highlights indicating errors. The Structure Checker panel on the left lists the following errors:

- Wedge Error Checker**: 1 Invalid wedge problem found
- Bond Angle Checker**: 2 bonds found with wrong angle

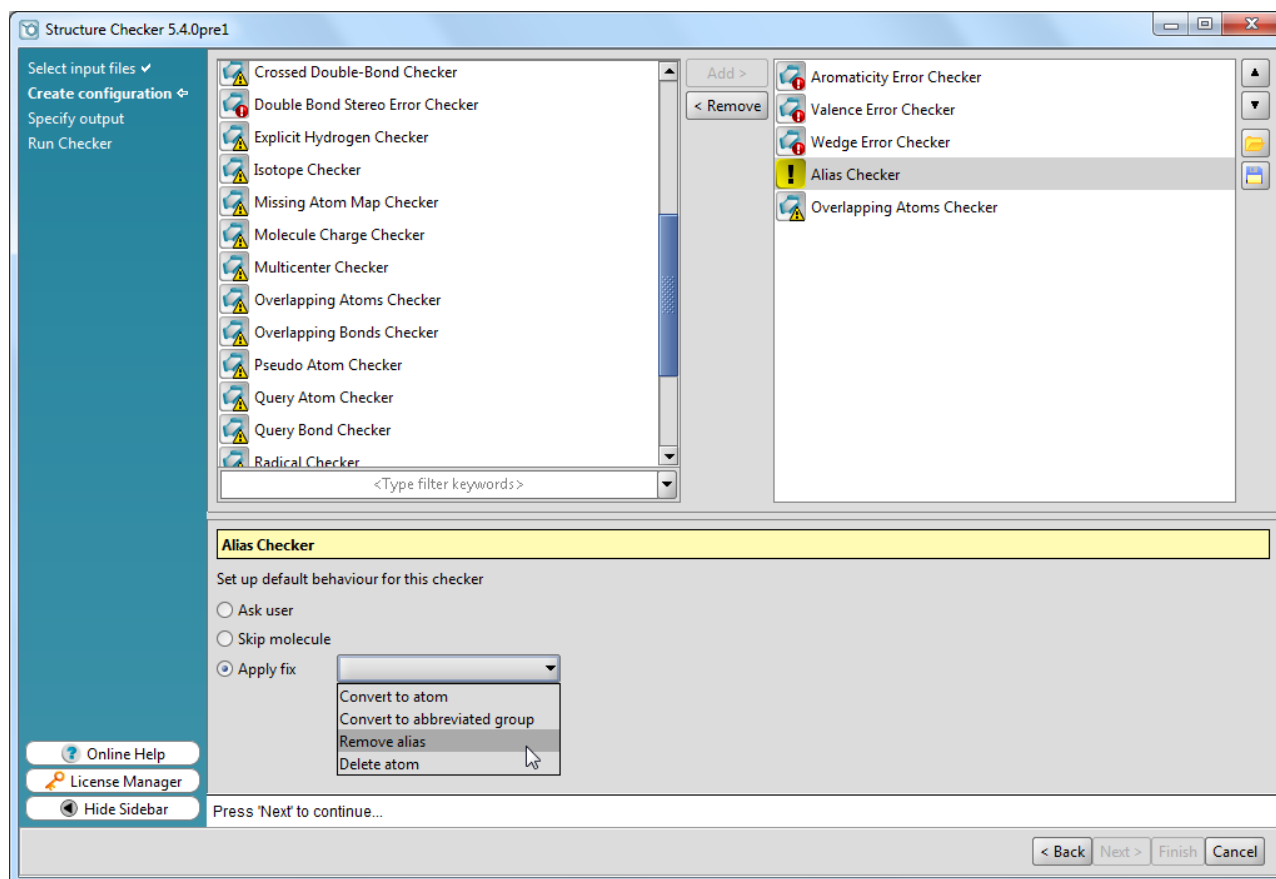
The Preferences dialog box is open, showing the 'Checkers' tab. It lists several checkers:

- Alias Checker**: Detects various atom aliases.
- Bond Length Checker**: Detects unpreferred bond lengths in structural formulas.
- Chiral Flag Error Checker**: Detects invalid chiral flags.
- Coordination System Error Checker**: Detects various coordination system drawing problems.

An 'Add new Checker...' button is visible next to the list. The main window also shows a status bar at the bottom indicating '2D \* Problems: 2'.

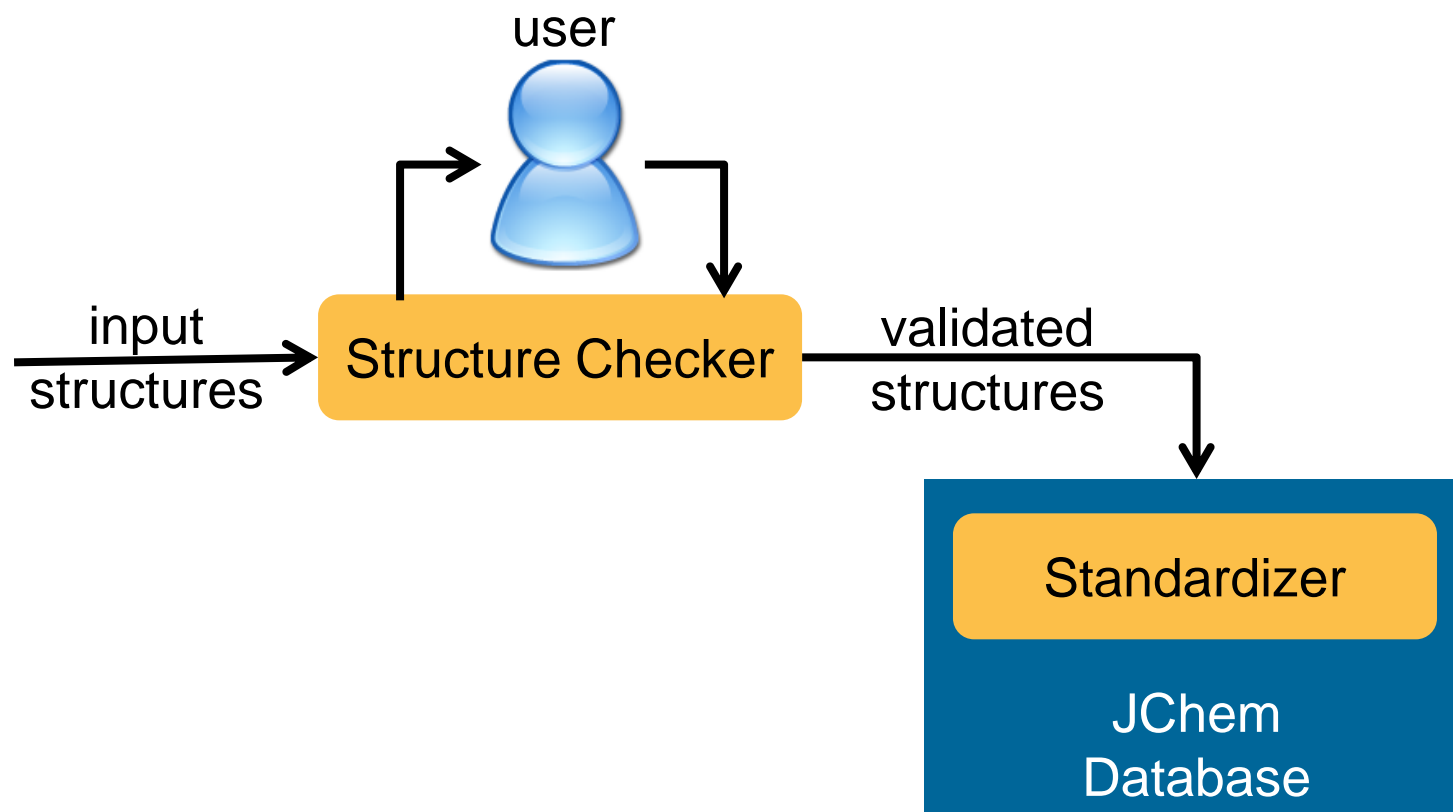
# Structure Checker Wizard

A command line tool and a wizard-like batch application of Structure Checker will be released in the next version of JChem. See live demo...



# Integration in Registration Systems

Structure Checker helps in the validation of structures before their registration in a database. Standardizer provides automatic canonicalization for correct chemical structure search.



---

Classes and Programming API

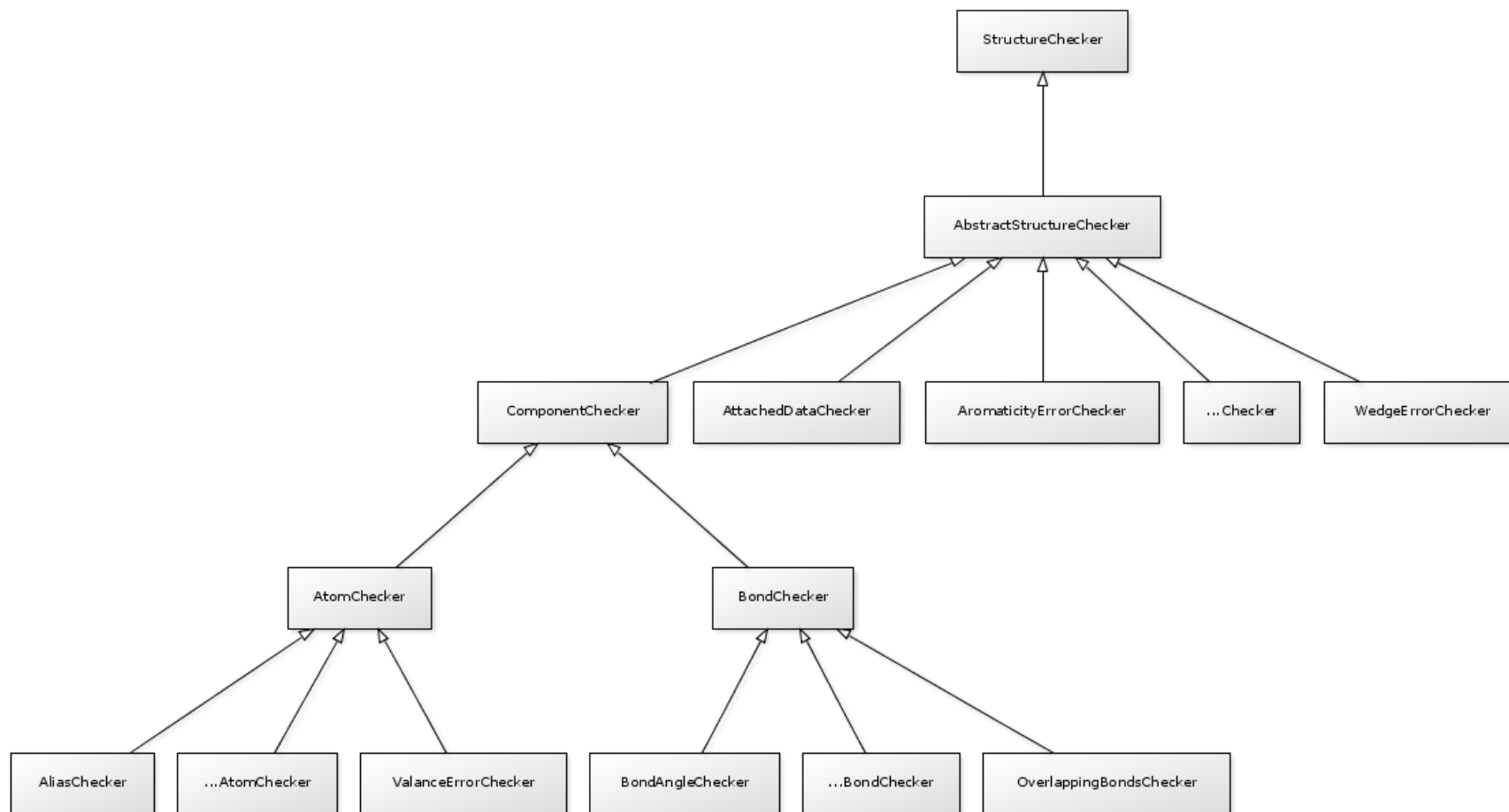
# ARCHITECTURE

# About Checkers and Fixers

---

- Checkers are classes for identifying specific issues of the given structures (aromaticity, query features, etc). The `check()` method creates a result which contains all information of the problem.
- Fixers are classes responsible for correcting specific problems by using the result provided by checkers. The `StructureCheckerResult` object.
- For the sake of flexibility, check and fix logics are well separated, checkers do not know about fixers, fixers do not know about checkers.

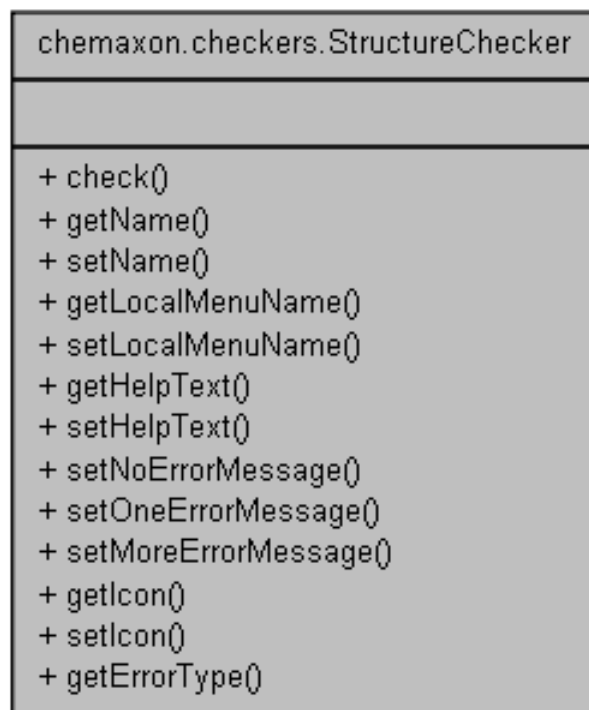
# Class hierarchy



# StructureChecker (interface)

---

- Checker classes have to implement the `StructureChecker` interface. The `check()` method contains the check logic which can return `StructureCheckerResult` instance describing the problem or `null` if no problem is detected.



# CheckerRunner

---

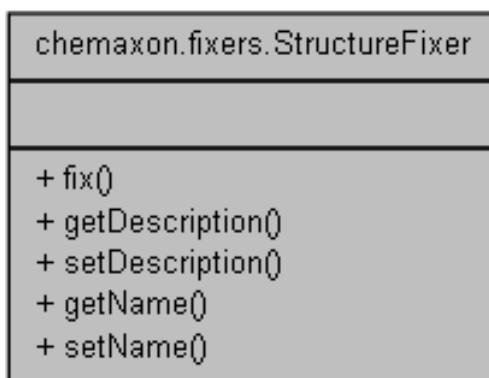
- The `CheckerRunner` interface defines an easy to use contract which makes possible to run a set of checkers from the API without using any `StructureChecker` instance directly
- The descendants of the `CheckerRunner` interface has to provide the functionality to run `StructureChecker` instances automatically on the given molecule and fix the problems with the associated `StructureFixer` instances.
- Example:

```
CheckerRunner runner;  
  
// initialize/initiate the current CheckerRunner instance  
List<StructureCheckerResult> results = runner.checkAndWait();  
  
// analyse checking results  
for (StructureCheckerResult result : results) {  
    List<StructureFixer> fixers = runner.getFixers(result);  
    // then execute one of the fixers  
}
```

# StructureFixer (interface)

---

- Fixer classes have to implement `StructureFixer` interface. The fixer logic is in the `fix()` method which returns `true` if the fix was successful `false` otherwise.
- Fixers can work from a `StructureCheckerResult` instances but know nothing about the checking methods. The results have to contain all necessary information of the problem (such as the `Molecule` instance, problematic atoms, etc.)



- `@Fixes` is an annotation to provide runtime information about a `StructureFixer` instance which kind of errors can fix.
- `StructureChecker` API using JAVA reflection and annotation technology to determine which fixers can fix the current problem (identified by a `StructureCheckerResult` instance)
- An example:

```
@Fixes ( {StructureCheckerErrorType.ABBREVIATED_GROUP,  
          StructureCheckerErrorType.ABBREVIATED_GROUP_WITH_ON  
          LY_EXPANDED} )
```

- This annotation means the `StructureFixer` instance (which has the annotation) can fix problems has `errorType` enumerated after `@Fixes`.

# Future directions

---

- Application specific configurations
- Command line application for batch checking
- Chemical Terms integration (IJC, JChem4XL, Cartridge, Web Services)
- Lots of new checkers and fixers

# Acknowledgements

---



Zsolt Mohácsi



István Rábel

---

# Questions