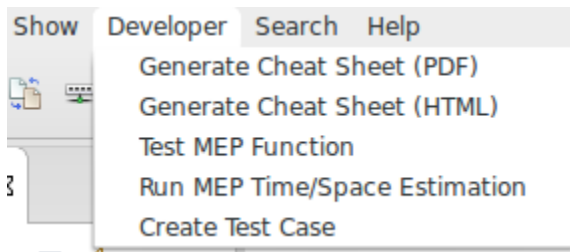


Developer Feature

The Developer Feature provides utilities to simplify the development and testing of components in Odyssey.



All utilities are accessible from within the main menu. Currently, the feature consists of three main parts:

CheatSheet Generator

The CheatSheet Generator adds the possibility to create a [cheatsheet](#) out of a running Odyssey setup. The user can choose between PDF and HTML. In case of PDF, the feature generates a LaTeX file with the documentation of each available operator in the current setup and compiles it into a PDF using PDFLaTeX by default. The command to compile the TeX source can be changed by setting the *latex* system parameter. In case of HTML, the feature generates an HTML document with the documentation of each available operator in the current setup.

The CheatSheet Generator can also be used by lecturers to generate course specific operator documentations.

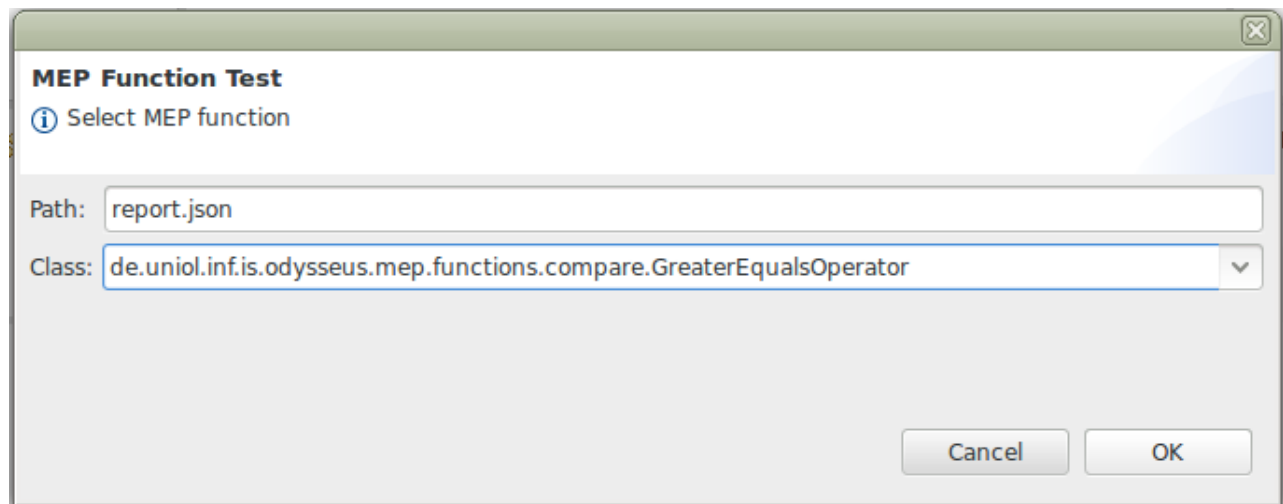
MEP Calculator

The MEP Calculator can be used to test MEP functions and complex expressions used in Select/Join predicates. Therefore, one can enter an arbitrary expression in the first line. The tool will then call the internal preprocessing to estimate the return type and output the optimized form of the expression, the conjunctive normal form (CNF), and the disjunctive normal form (DNF).

A screenshot of the 'MEP Calculator' tool interface. The window has a title bar with a close button. The interface contains five input fields with labels on the left: 'Expression:', 'Return Type:', 'Optimized:', 'CNF:', and 'DNF:'. The 'Expression' field contains the text 'max([1.0,2.0]) < x || (1.0 > x && a < b && 1 < 2)'. The 'Return Type' field contains 'Boolean'. The 'Optimized' field contains '(((1) > (x)) && ((a) < (b))) || ((2) < (x))'. The 'CNF' field contains '(((2) < (x)) || ((1) > (x))) && (((2) < (x)) || ((a) < (b)))'. The 'DNF' field contains '(((1) > (x)) && ((a) < (b))) || ((2) < (x))'.

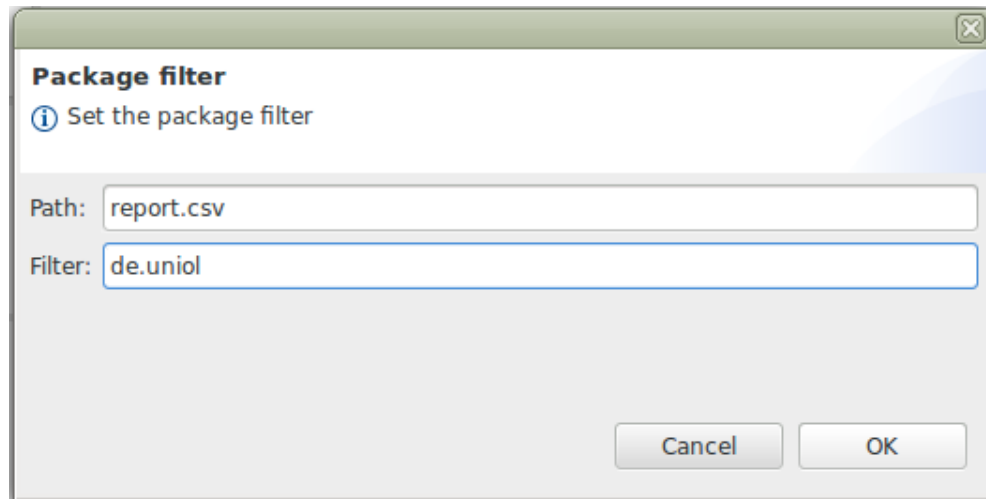
MEP Tests and Time/Space requirements

The first utility called "Test MEP Function" allows to test a specific MEP function with different accepted types to make sure the function shows the expected behavior for all accepted data types. To do so the tool generates specific test cases for all permutations of possible accepted data type combinations.



The result of the MEP function test is stored in JSON format and includes a detailed report about all tests including parameter type, value, and stacktrace in case of an error.

The second utility called "Run MEP Time/Space Estimation" estimates the average time and space requirements for all MEP functions and calculates the relative time and space complexity score which can be used for the predicate optimization in the rewrite phase during the query compilation.



The result is a CSV file with the following schema:Symbol, Class, Time Score, Space Score, Time (ns), Space (byte)

The estimation can be restricted to a specific set of packages by defining a filter. Note: The estimated time and space score is a relative value of the estimated time and space requirements. Also the estimated scores can differ depending on the platform.

Testcase Generator

The last entry in the developer menu opens a wizard to create a "Test Case" using the Testcase Generator. The [Testcase Generator](#) provides an editor to generate operator specific integration tests by generating a operator specific query including an input stream, and a resulting output stream.

A screenshot of a software configuration window. The window has a light gray background and a blue gradient header. It contains the following fields and controls:

- Name:** A text input field containing the text "MyTestcase".
- Operator:** A dropdown menu with "MAP" selected and a downward arrow.
- Path:** A text input field containing the path "/home/ckuka/Documents/Projects".
- Choose from project:** A button with the text "Choose from project".
- Navigation buttons:** Four buttons at the bottom: "< Back", "Next >", "Cancel", and "Finish".

Please visit the documentation of the [Testcase Generator](#) and the [Integration Test](#) documentation for more information on how to test in Odysseus.