Debugging using Totalview

Inspect program variables; watch execution; stop exactly where program dies; step through new routines; dual debugging; plotting of arrays; (replay engine on linux)

Pros:

- works with C, C++, Fortran, some python and cuda
- works with OpenMP and MPI (many flavors)
- runs on Linux and Mac (but Mac version not as good)

Cons:

- location of menu items not that intuitive

High-level Overview

- set breakpoints at specific lines (conditionals too)
- stop in <subroutine name>
- modify vars on-the-fly and continue executing
- set watch points *(only on linux right now)*
- display variables, arrays, pointers, objects *(2D plotting on linux)*

Compilation

generates debugging info

- gfortran line: -g -fbounds-check -ffpe-trap=invalid,overflow,zero
- ifort line: -g -CB -CU -fpe0 -ftrapuv

check bounds

check uninitialized

sets uninitialized stack vars to weird val

-check all?

Debugging Examples Demo

1. where did my program die? (Fortran)

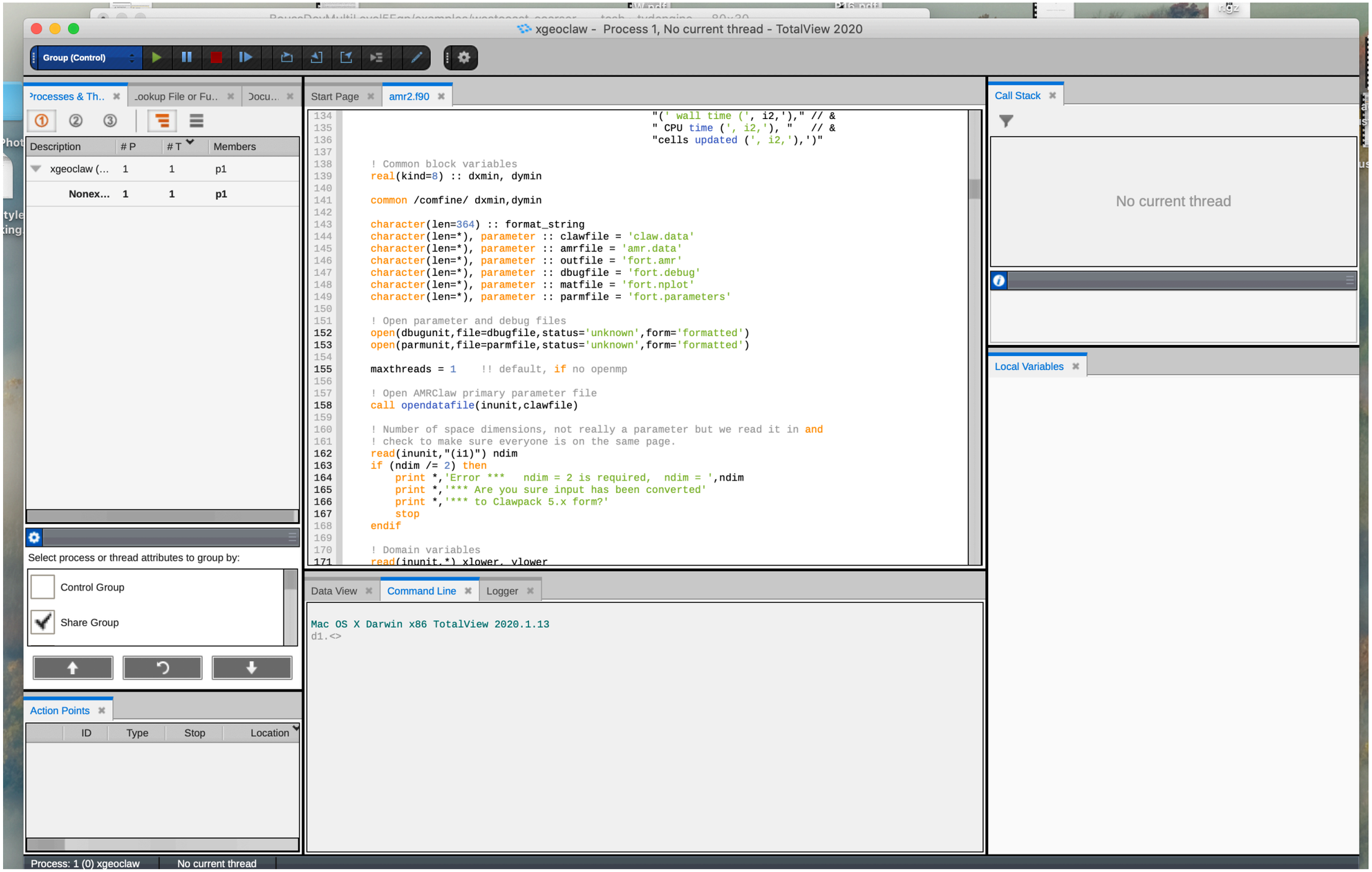
2. look at variables at source of error (Fortran)
   (w/o multiple print statements and recompiles)

3. modify vars on-the-fly and continue executing (Fortran)
   (use of expression list, and conditional breakpoints)

4. openmp example *(expression list across threads)* (C)

5. mpi example *(expression list across processes)*
   (totalview -args mpiexec -np 4 <mpi_executable>)

New GUI Totalview



also a new user interface (-newui) version if you prefer guis.