

IDEs and debugging JEDI

Wojciech Smigaj (Met Office)

Some popular IDEs with C++ support

- CLion
- Eclipse
- Qt Creator
- Visual Studio Community/Professional/Enterprise (Windows only)
- Visual Studio Code
- XCode (Mac only)

IDE benefits

- Quick code navigation
- Autocompletion, contextual help
- On-the fly detection of syntax errors
- Help with refactoring (e.g. renaming variables)
- Integrated debugger

```
115 void MetOfficeBuddyCheck::applyFilter(const std::vector<bool> & apply,  
116                                       const Variables & filtervars,  
117                                       std::vector<std::vector<bool>> & flagged) const {  
118     // Identify observations  
119     std::vector<size_t> getValidObservationIds(const std::vector<bool> & apply)  
120     const std::vector<size_t> validObsIds = getValidObservationIds(apply);  
121     Metadata obsData = collectMetadata() expected ';' at end of declaration  
122     const std::vector<float> bgErrorHorizCorrScales = calcBackgroundErrorHorizontalCorrelationScales(  
123         validObsIds, obsData.latitudes);
```

Debugger basics

- **Breakpoints:** code locations where the debugger will interrupt a program. Can be unconditional or conditional.

```
22  ObsErrorFactorLatRad::ObsErrorFactorLatRad(const eckit::Local
23  : invars_() {
24  // Check options
25  options_.deserialize(conf);
26
27  ASSERT((options_.latitudeParameters.value()).size() == 4);
```

- Once a breakpoint is hit, the developer can examine the state of the program (e.g. inspect the **call stack** and values of **local variables**).
- Subsequently, they can resume normal execution or **step through** the code.
 - **Step over:** execute a line of code as a whole.
 - **Step into:** like *Step over* unless the current line contains a function call. If so, suspend execution at the first statement in that function.
 - **Step out:** execute code until the current function returns.

Debugging: compilation flags

Binaries must contain (or be accompanied by) debugging symbols.

- To build JEDI with debugging symbols, select the *Debug* or *RelWithDebInfo* configuration with CMake/Ecbuild
 - `ecbuild --build=Debug path_to_source_folder`
 - `cmake -DCMAKE_BUILD_TYPE=Debug path_to_source_folder`
- Compiling with optimisations makes debugging harder. If you can afford it, use the *Debug* rather than the *RelWithDebInfo* configuration.