

VS Code and You

Why You Should Make Your Computer Work for You
(and How!)

Adrianna Foster^a and Sam Rabin^b

a: Project Scientist I, NCAR CGD Terrestrial Sciences Section

b: Software Engineer II, NCAR CGD Terrestrial Sciences Section

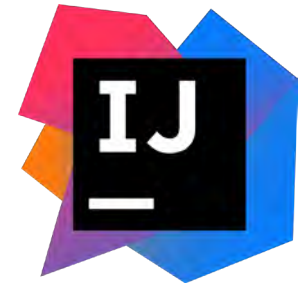
2024 Software Engineering Winter Working Group Meeting

March 4, 2024



What is an IDE?

- Integrated Development Environment
- Helps programmers in code development process
- Combines software editing, building, testing
- Speeds up day-to-day tasks



IDE Usage within CGD

Role

- Scientist
- Software Engineer
- Mix of both



No really,
what's an IDE?



Text editor,
no syntax
highlighting



Text editor,
syntax
highlighting



Some IDE
usage



Moderate
IDE usage



Heavy IDE
usage



IDE Usage within CGD

Role

- Scientist
- Software Engineer
- Mix of both



No really,
what's an IDE?



Text editor,
no syntax
highlighting



"emacs -nw 4 life!"

Text editor,
syntax
highlighting



Some IDE
usage



Moderate
IDE usage



Heavy IDE
usage

Pop quiz

What is $5,772,127 \div 4,513$?



Pop quiz

What is $5,772,127 \div 4,513$?



Scenario

We'd like to rename an defined type

```
! FATES COHORT TYPE
```

```
jessica needham, 5 months ago | 3 authors (You and others)
```

```
type, public :: fates_cohort_type
```

```
You, 11 months ago • move cohort type into its own modul
```

```
! POINTERS
```

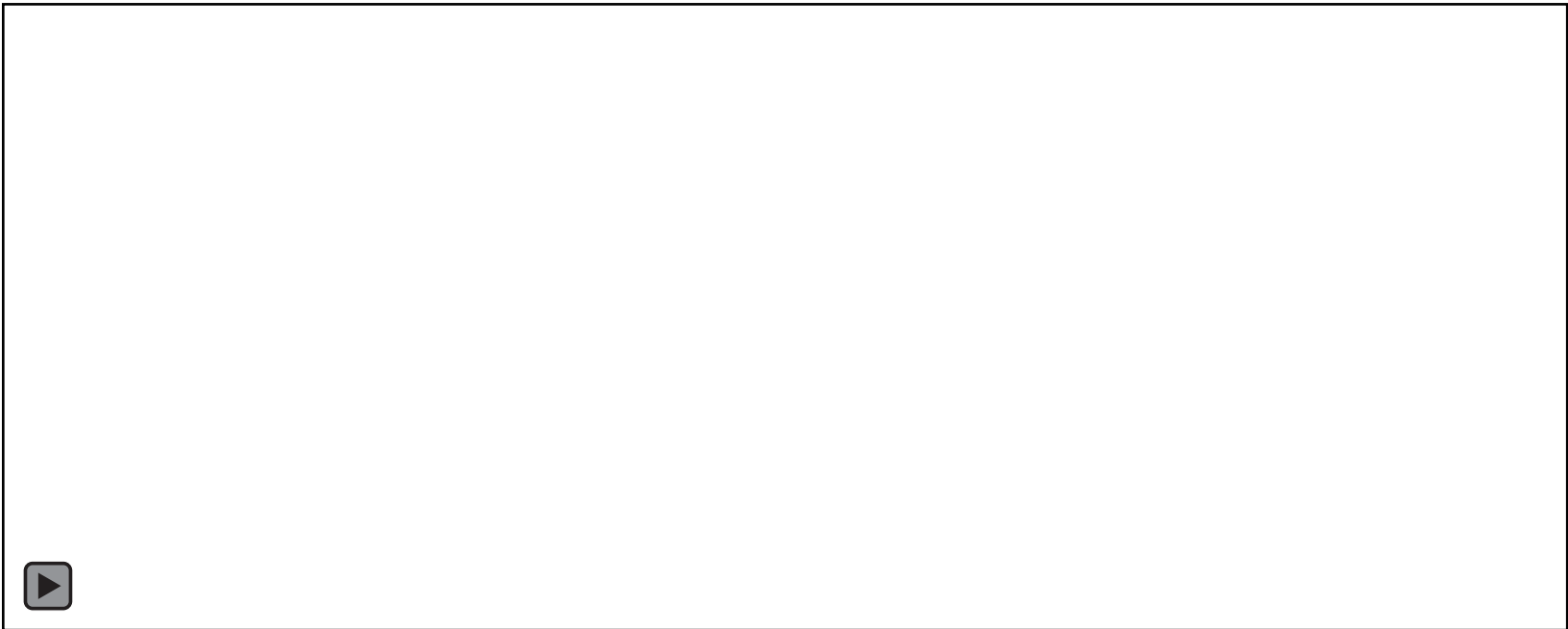
```
type (fates_cohort_type), pointer :: taller => null() !
```

```
type (fates_cohort_type), pointer :: shorter => null() !
```

```
~/Documents/ncar/CTSM/src/fates ± main  
> grep -o "fates_cohort_type" -r * | wc -l  
196
```

Scenario

We'd like to rename an defined type



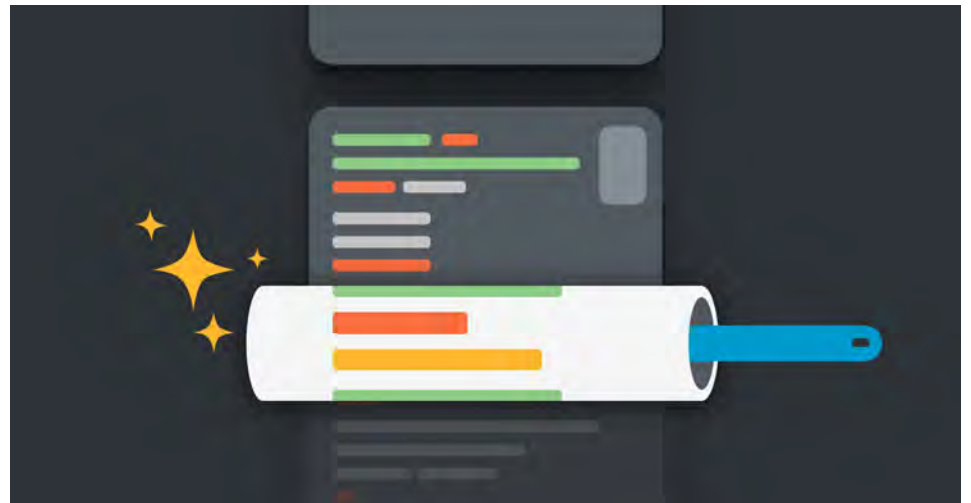
What can IDEs do?

- Syntax highlighting
- Code autocompletion
- Refactoring
- Code navigation
- Build, execution, and debugging tools
- Source control GUI

Linters

Tool to analyze code

- knows language syntax
- can help optimize code
- find typos & syntax errors
- warn about unused variables
- conform to language (or source code) standards



Syntax highlighting

```
! fortran  
a = (/1, 2, 5, 6, 7/)  
n = size(a)  
b = n + 1
```

```
# python  
a = [1, 2, 5, 6, 7]  
n = len(a)  
b = n + 1
```

```
// Java  
int[] a = {1, 2, 5, 6, 7};  
int n = a.length;  
int b = n + 1;
```

Syntax highlighting

```
! fortran  
a = (/1, 2, 5, 6, 7/)  
n = size(a)  
b = n + 1
```

```
# python  
a = [1, 2, 5, 6, 7]  
n = len(a)  
b = n + 1
```

```
// Java  
int[] a = {1, 2, 5, 6, 7};  
int n = a.length;  
int b = n + 1;
```

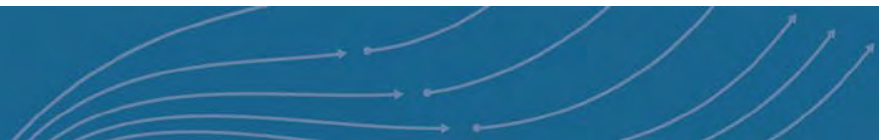
Code Autocompletion and Documentation



Code Autocompletion and Documentation

IntelliSense – intelligent code completion; quick access to documentation

```
E > case >  (function) def isdir(s: FileDescriptorOrPath) -> bool
# load the
case.load_e Return true if the pathname refers to an existing directory.
# create th  See Real World Examples From GitHub
if os.path.isdir(os.path.join(rundir, "timing")):
    shutil.rmtree(os.path.join(rundir, "timing"))
```

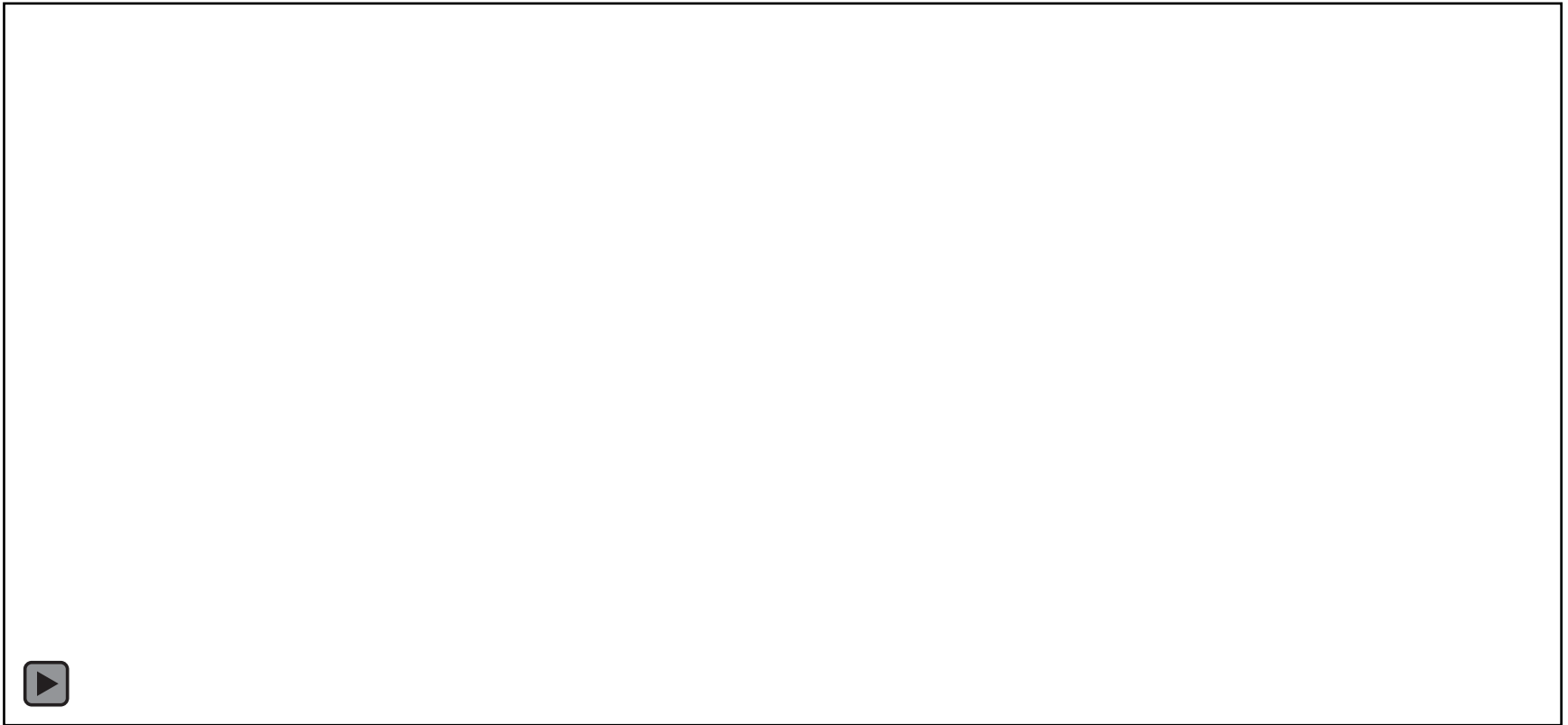


Code Navigation



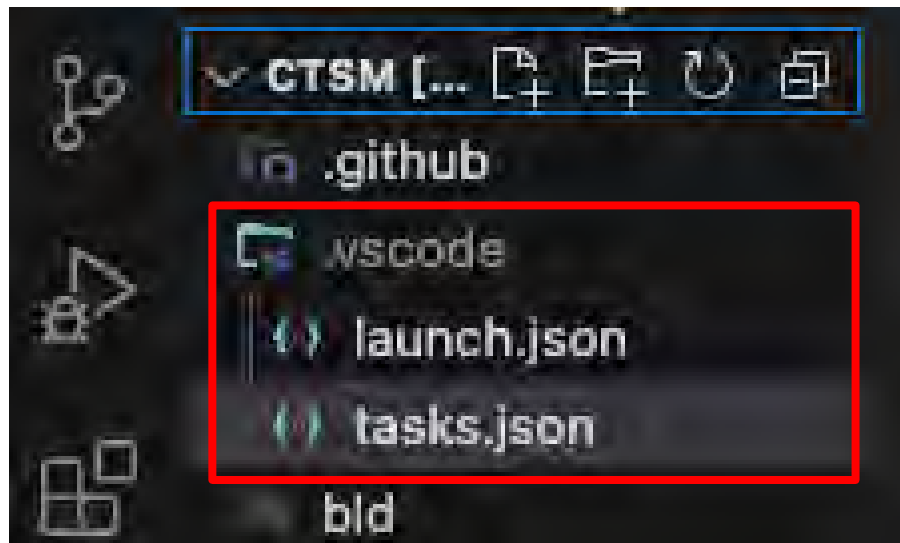
Code Refactoring

PyCharm



Building, Executing, Debugging

Can build, execute, and debug code (yes even Fortran!) from within VS Code



Building, Executing, Debugging

tasks.json – for automating everyday tasks (my use)

```
},
{
  "label": "create",
  "type": "shell",
  "command": "./create_test_case",
  "options": {
    "cwd": "/glade/work/afoster/ctsm_scripts"
  },
  "args": [
    "${input:site-tag}"
  ],
  "problemMatcher": []
},
]
```

"create" – creates a CTSM case

bash script that creates a case using an input config file

script argument (i.e. the config file name)

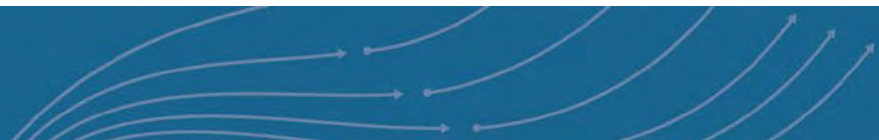
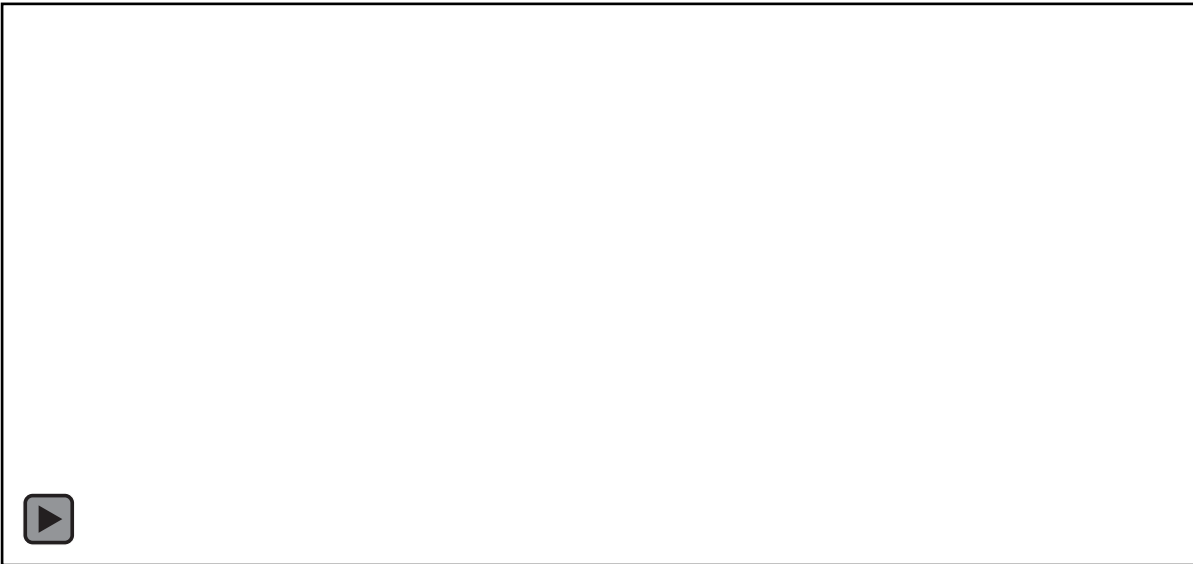
Building, Executing, Debugging

tasks.json – for automating everyday tasks (my use)

```
1 ],
2 "inputs": [
3   {
4     "id": "site-tag",
5     "type": "pickString",
6     "description": "Enter the type of test or site name.",
7     "options": [
8       "BONA",
9       "FATES_f45",
10      "CLM_f45"
11    ],
12     "default": "CLM_f45"
13   },
14   {
```

Building, Executing, Debugging

tasks.json – for automating everyday tasks (my use)



Building, Executing, Debugging

launch.json – debugging!

```
.vscode > {} launch.json > [ ] inputs > {} 0 > description
1
2 {
3   "version": "0.2.0",
4   "configurations": [
5     {
6       "name": "Run GDB",
7       "type": "cppdbg",
8       "request": "launch",
9       "program": "${input:case-dir}/bld/cesm.exe",
10      "args": [],
11      "stopAtEntry": false,
12      "cwd": "${input:case-dir}/run",
13      "externalConsole": false,
14      "MIMode": "gdb",
15    },
16    "inputs": [
17      {
18        "id": "case-dir",
19        "type": "promptString",
20        "description": "Enter the full path to your case output directory.",
21        "default": "/glade/work/afoster/test_cases/output/BONA_test",
22      },
23    ],
24  }
```

executable

run directory



Other stuff!

- Customizing your editor
 - custom themes
 - guide bars
 - interfacing with git
 - mini-window
 - source control pane and outline
- emacs/vim enthusiasts can use emulator
- remote ssh
- gitlens
- lots of other extensions



IDE Usage within CGD

