

Parallel Programming

Recitation Session 10

Thomas Weibel <weibelt@ethz.ch>

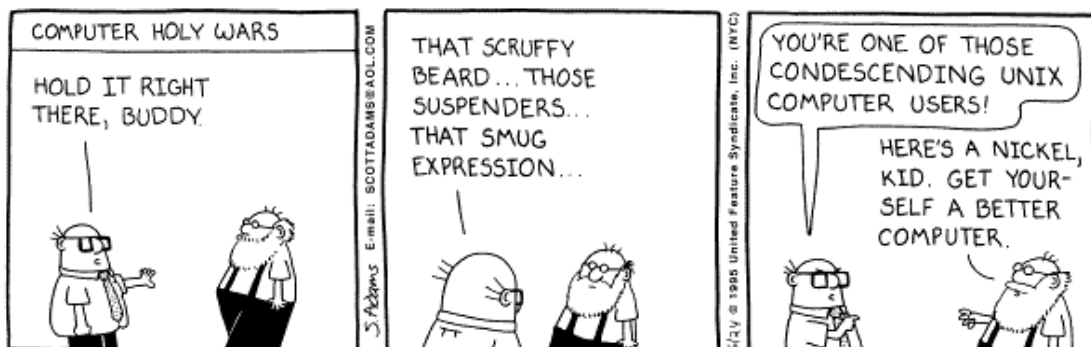
Laboratory for Software Technology,
Swiss Federal Institute of Technology Zürich

May 20, 2010

Introduction

Executive Summary

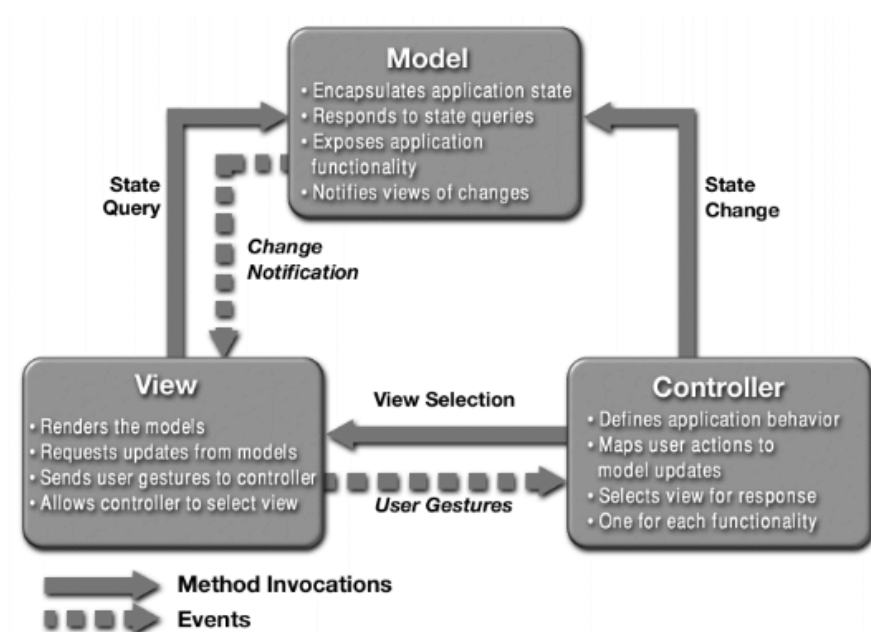
- Model-View-Controller
- Solution to Game of Life
- OpenMP in a nutshell
- JOMP: OpenMP for Java



Outline

- 1 MVC Revised
- 2 Game of Life
- 3 OpenMP
- 4 Assignment 10
- 5 OpenMP in Java

Model-View-Controller



Source: <http://java.sun.com/blueprints/patterns/MVC-detailed.html>

Roles of Participants

- Model:
 - domain specific knowledge
 - adds meaning to raw data
- View:
 - render data
 - capture user gestures
- Controller:
 - respond to events
 - asynchronously invoke changes on model

Outline

- 1 MVC Revisited
- 2 Game of Life**
- 3 OpenMP
- 4 Assignment 10
- 5 OpenMP in Java

Design of the Solution

- Application class:
 - Main window and controls
 - Create and terminate the modeling thread
- Field class:
 - Drawing field: keeps copy of model state
- Controller class:
 - Has data and methods that can be applied on the data

Outline

- 1 MVC Revisited
- 2 Game of Life
- 3 OpenMP**
- 4 Assignment 10
- 5 OpenMP in Java

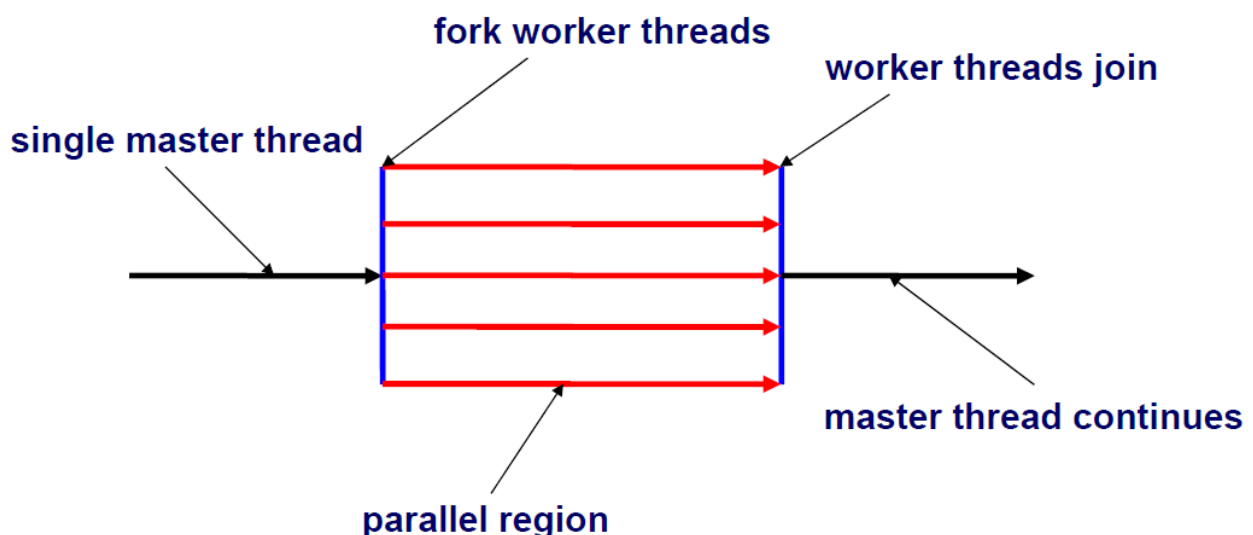
OpenMP in a Nutshell

- OpenMP is an API that consists of three parts
 - Directive-based language extension
 - Runtime library routines
 - Environment variables
- Three categories of language extensions
 - Control structures to express **parallelism**
 - Data environment constructs to express **communication**
 - Synchronization constructs for **synchronization**

Parallel Control Structures

Alter flow of control in a program

→ fork/join model



Parallel Control Structures

- Two kinds of parallel constructs
 - Create multiple threads (parallel directive)
 - Divide work between an existing set of threads
- Parallel directive
 - Start a parallel region
- For directive
 - Exploit data-level parallelism (parallelize loops)
- Sections directive
 - Exploit thread-level parallelism (parallelize tasks)
- Task directive (OpenMP 3.0)
 - Task with ordering (not possible with sections)

Communication & Data Environment

- Master thread (MT) exists the entire execution
- MT encounters a parallel construct
 - Create a set of worker threads
 - Stack is private to each thread
- Data Scoping
 - Shared variable: single storage location
 - Private variable: multiple storage locations (1 per thread)

Synchronization

- Co-ordination of execution of multiple threads
- Critical directive: implement mutual exclusion
 - Exclusive access for a single thread
- Barrier directive: event synchronization
 - Signal the occurrence of an event

Exploiting Loop-Level Parallelism

- Important: program correctness
- Data dependencies:
 - If two threads read from the same location and at least one thread writes to that location
 - Data dependence
 - Example:

```
for (i = 1; i < N; i++)  
    a[i] = a[i] + a[i-1];
```

Loop carried dependence

Examples

```
for (i = 2; i <= n; i+= 2)
    a[i] = a[i] + a[i-1]
```

No dependence

```
for (i = 1; i <= n/2; i++)
    a[i] = a[i] + a[i + n/2]
```

No dependence

```
for (i = 1; i <= n/2 + 1; i++)
    a[i] = a[i] + a[i + n/2]
```

Dependence:
 read(1+n/2)
 write(n/2+1)

Parallel Directive

```
//omp parallel shared (a,b) private (c,d)
```

- Starts a parallel region
- shared: variable is shared across all threads
- private: each thread maintains a private copy

Distribute Loop Iterations

```
//omp for schedule(dynamic)
```

```
//omp for schedule(static)
```

- Distribute loop iterations to worker threads
- *dynamic*: loop-chunks are assigned to threads at runtime
- *static*: loop-chunk assignment before the loop is executed

Critical Section

```
//omp critical
```

- Starts a critical section
- Code section is executed by a single thread at a time

Outline

- 1 MVC Revisited
- 2 Game of Life
- 3 OpenMP
- 4 Assignment 10**
- 5 OpenMP in Java

Tasks

Task 1

- Parallelize an existing implementation with OpenMP
- Which loop nest would you parallelize?
- Do you need a critical section?

Task 2

- Implement a Block Matrix Multiplication
- Divide the source matrices into sub-matrices
- Assign a thread to each sub-matrix

Which one performs better?

Outline

- 1 MVC Revisited
- 2 Game of Life
- 3 OpenMP
- 4 Assignment 10
- 5 OpenMP in Java**

JOMP

- OpenMP is not natively supported by Java
- JOMP: source to source compiler
- See http://www2.epcc.ed.ac.uk/computing/research_activities/jomp/index_1.html for more information

How to use on the Console

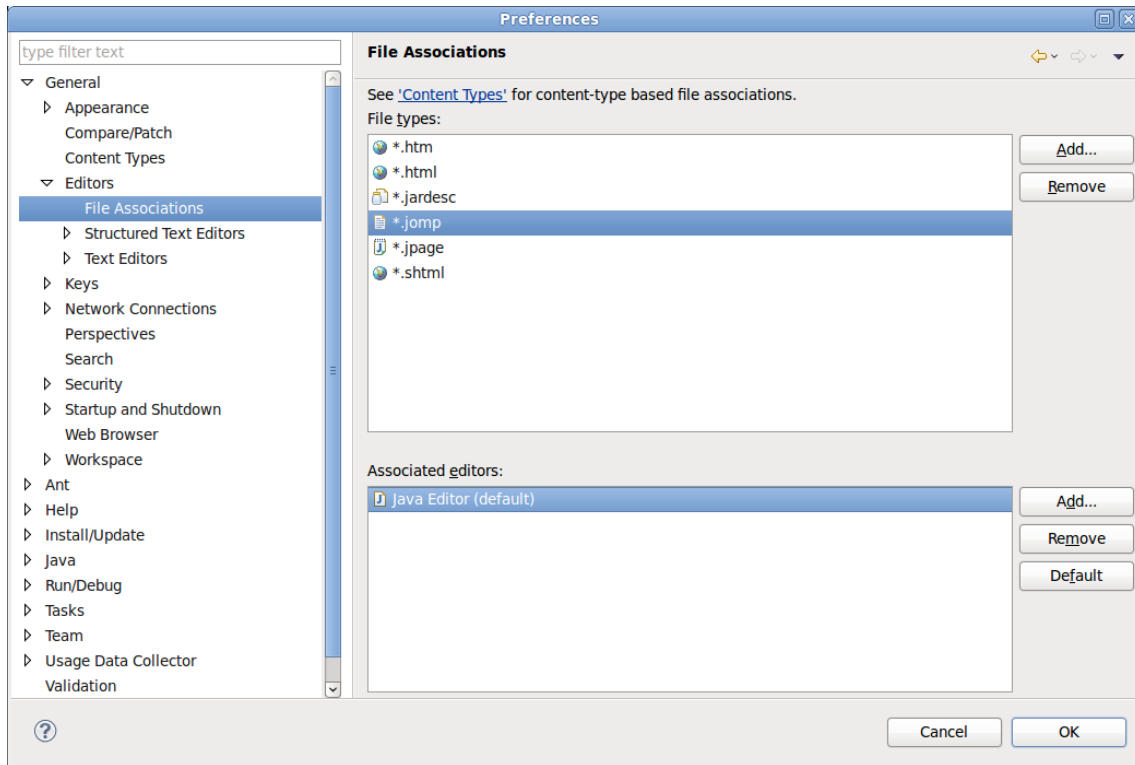
- Download jomp1.0b.jar from the course page
- Import external JAR file to your project (classpath)
 - `export CLASSPATH=$CLASSPATH:/path/to/jomp1.0b.jar`
 - Use option `-cp /path/to/jomp1.0b.jar` when calling Java to transform `file.jomp` into `file.java`
- Perform the following steps in the console
 - `java jomp.compiler.Jomp file.jomp`
 - generates `file.java`
 - `javac file.java`
 - `java file`

Hot to use in Eclipse

- Add `MatrixMultiply.jomp` to your project
- Add a new class `MatrixMultiply.java` to your project
 - this file will be overwritten by JOMP
- Copy `jomp1.0b.jar` to your project's `lib/` directory

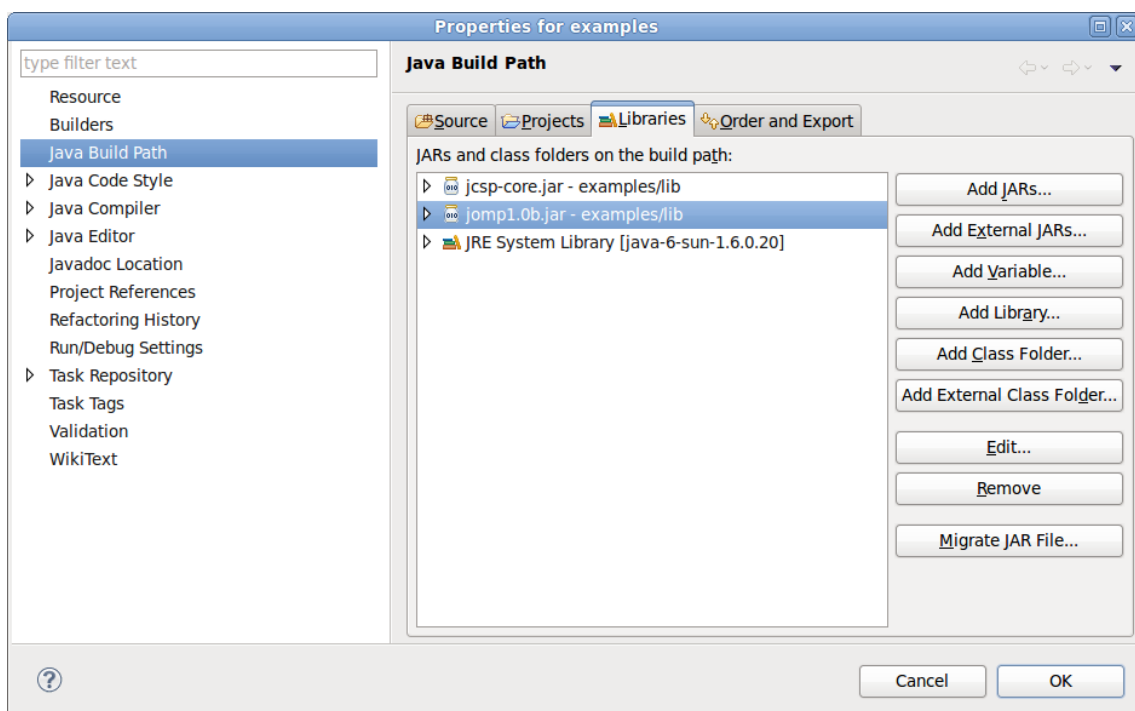
How to use in Eclipse: File Association

Add a file association for *.jomp as Java files (for syntax highlighting and auto completion):



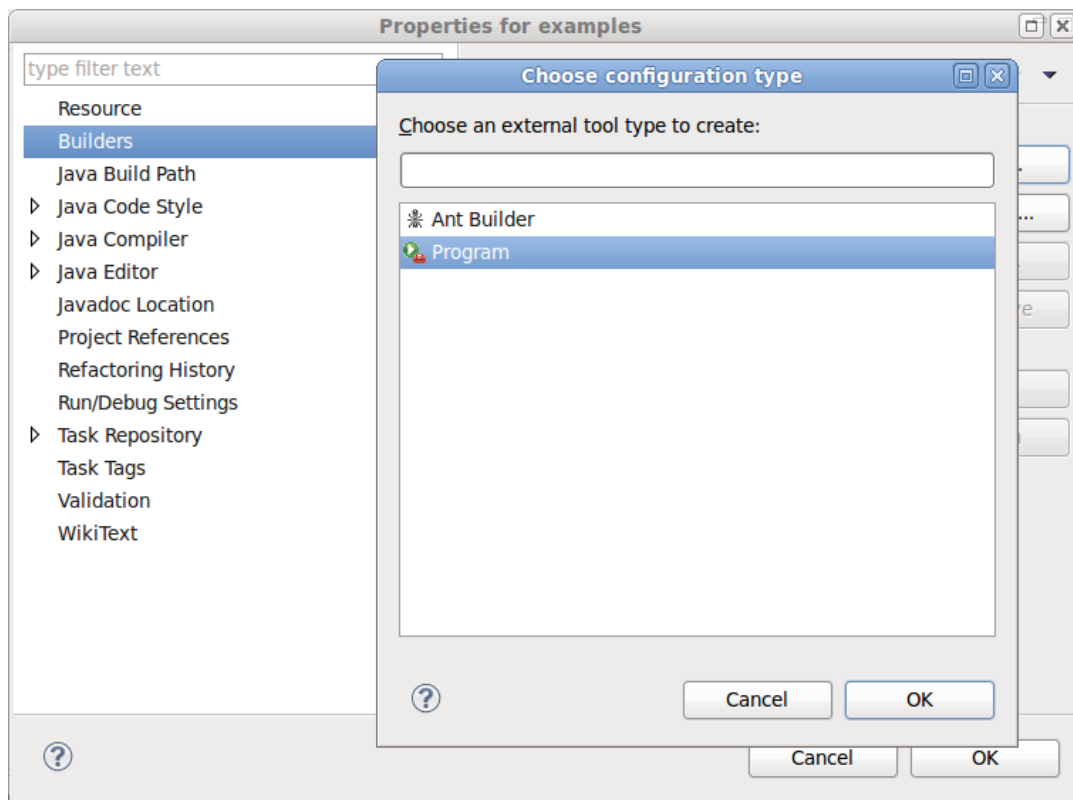
How to use in Eclipse: Java Build Path

Add jomp1.0b.jar to your project's build path:

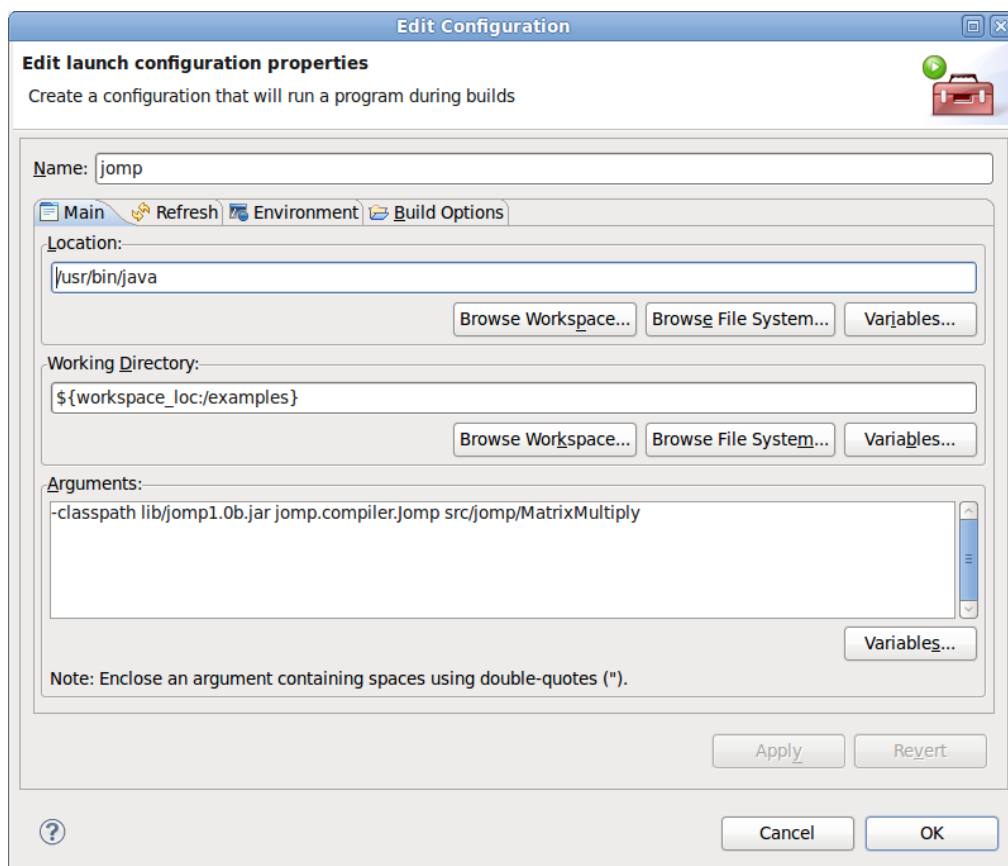


How to use in Eclipse: New Builder

Add a new builder for JOMP:

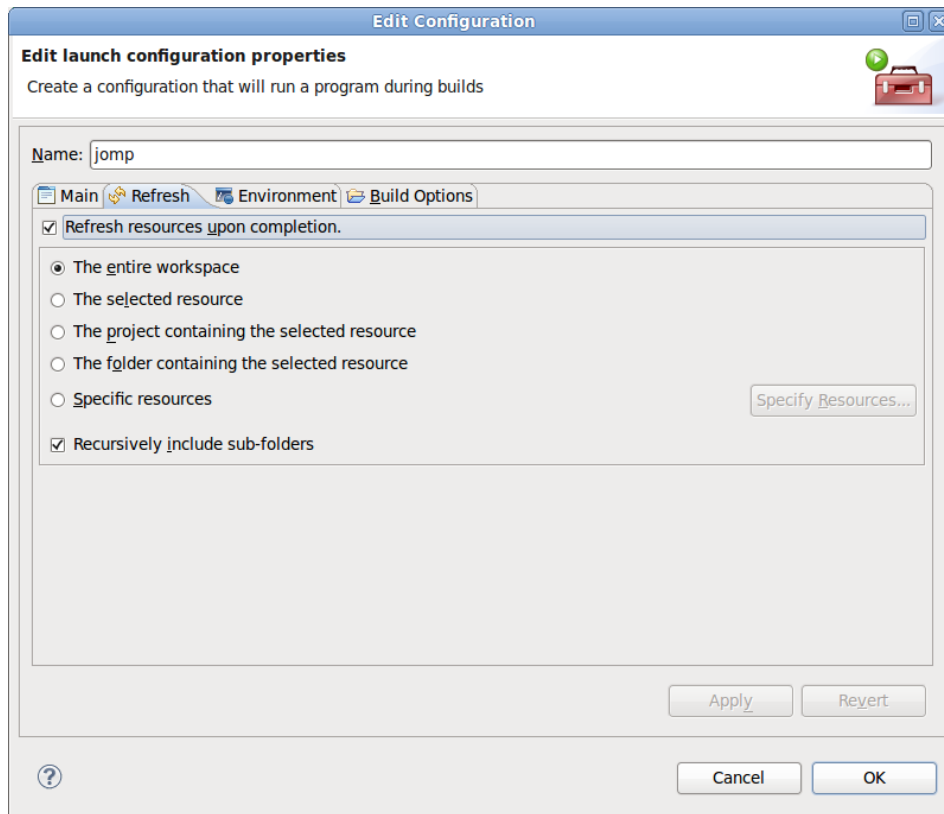


How to use in Eclipse: Configure Builder



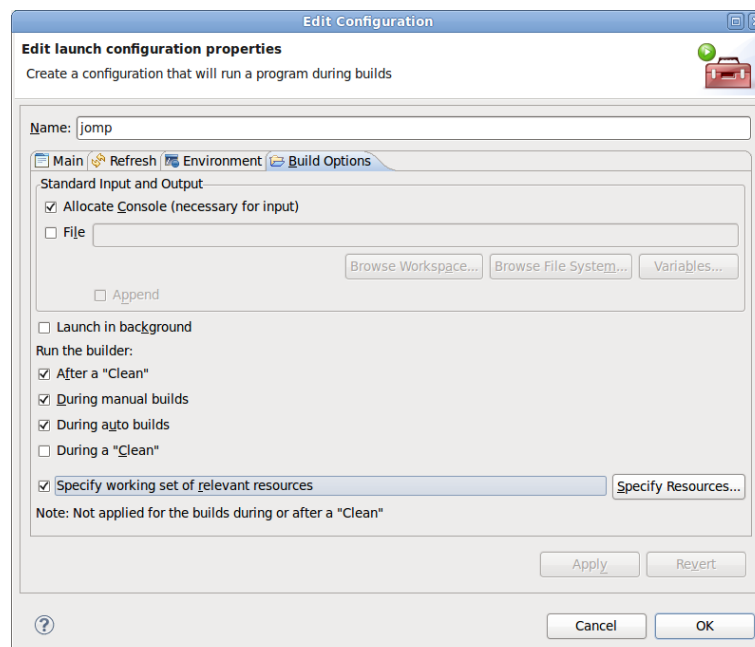
How to use in Eclipse: Configure Builder

Check “Refresh resources upon completion”:



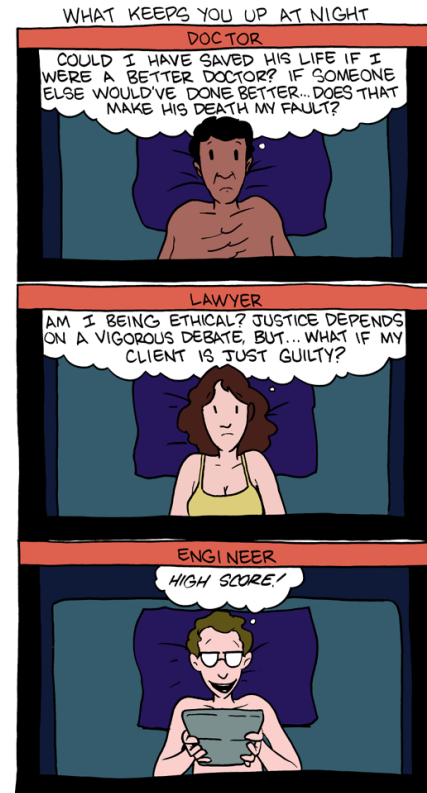
How to use in Eclipse: Configure Builder

- Check “During auto builds”
- Check “Specify working set of relevant resources”
- Click “Specify Resources” and select MatrixMultiply.jomp



Summary

- Model-View-Controller pattern and Game of Life
- OpenMP
- Data dependence
- How to use JOMP



Source: <http://www.smbc-comics.com/index.php?db=comics&id=1845>