OOP is for Boomers

Милош Суботић

20. јануар 2024.

Дефиниција

- Master Linker је почасна титула.
- Изражена потреба код old-school језика који се преводе у машински код.
- Захтева **истинско разумевање** процеса
  - Интерних детаља програмских језика
  - Билдовања (енгл. build) SW
  - Дистрибуције SW
  - Коришћења SW

Мотивација

- Билдовања треба бити
  - поновљиво (не ad-hoc)
  - скалабилно
- Утиче на развој. Немогуће га је избећи. Алати су битни.

Пример

Сениор инжењер који држи курс C језика јуниору:
90% *инжењера, не студената, кад добију* `undefined symbol` *покушава да нешто инклудује!*

IDE

- CodeBlocks, Visual Studio, Eclipse...
- Углавном некако везано за Виндозу
- Симпатични за почетнике, али:
  - Гломазни
  - Ограничени, крути
  - Нема аутоматизације
  - Непортабилни

Едитор + терминал

- Lightweight
- Аутоматизација
- Неограничене могућност

Међутим

- Едитор + Терминал ништа не значе без (доброг) система за билдовање.
- Постоје IDE-и који омогућавају коришћење.
  - Већина не подржава :(Отуда та ограниченост.

У козноли

- Ово није прихватљиво:

```
1 g++ -o prog main.cpp
```

- Није довољно само накуцати C++ код, мора се документовати начин компајлирања.

Па како онда

- Прођемо кроз `make` језик на конкретном примеру
- Уједно ћемо ући у дубине C++-а, **ABI**-ја, билдања...

Make

- Датотека је `Makefile`
- Команда која се изврши у директоријуму где се налази `Makefile`:

```
1 make
```

- Наравно да има своје опције.

Основа

Ово је једно правило (енгл. rule):

```
1 prog: main.o
2     g++ -o prog main.o
```

Невидљиво правило

```
1 main.o: main.cpp
2     g++ -c -o main.o main.cpp
```

Сва невидљива се могу довити са:

```
1 make -p -f/dev/null > Makefile.defalt
```

Пречице

```
1 prog: main.o
2     g++ -o $@ $^
```

Променљиве

```
1 prog: main.o
2     ${CXX} -o $@ $^
```

Право невидљиво правило

```
1 %.o: %.cpp
2     ${CXX} -c -o $@ $^
```

Најтипичније

```
1  prog: main.o
2      ${CXX} -o $@ $^
3
4  .PHONY: clean
5  clean:
6      rm -f *.o test
```

Коришћење:

```
1  make clean
```

Hands-on time

```
make
make run1
make run2
make run3
touch main.cpp
make
make run2
make clean
make CONFIG_INC_TEST=1
make run2
make clean
make PLUGIN_IDX=1
make run2
```

Ректо ко ради

```
1 touch legacy/legacy.h
2 make
```

- Захтева да знате ко шта инклудује. Врло напорно.
- Било би згодно да неки алат то одради за нас.

Линковање

```
objdump -t main.o
objdump -Ct main.o
objdump -CT plugins/lib/libplugin0.so
ldd ./test
LD_LIBRARY_PATH=helpers ldd ./test
readelf -d `which julia` | grep RPATH
```

Динамичко vs Runtime

```
1 JULIA_PATH=$HOME/.local/opt/julia/64b/julia
    -1.6.7/
2 ldd $JULIA_PATH/lib/libjulia.so
3 ldconfig -p
4 ldd $JULIA_PATH/lib/libjulia.so
```

Библиотеке

`PREFIX` укључује најчешће до најређег:

- `bin/`
- `lib/`
- `include/`
- `share/`

Могући префикси:

- `/`
- `/usr/`
- `/usr/local/`
- `$HOME/.local/`

Такође:

- `/opt/нешто`
- `/tools/нешто`

Помоћ за библиотеке

```
pkg-config --libs sfml-all
pkg-config --libs sfml-graphics
```

Мета-билдовање

- autoconf
- cmake

Мета-билдовање

- `configure`
- `install`
- `distclean`
- `dist`

Тарбали

- `libsfml.tar.gz` или `libsfml.tar.bz2`
- Билдање кода.

Билдовање + Мета-билдовање

- `waf`

Пакети менаџери

- `apt`, `dfn`, `pacman`
- Ту Линукс бриљира

Framework

- MSYS2
  - pacman као пакет менаџер

Закључак

- Инсталирајте Линукс
- Користите конзолу

Хвала на пажњи