Hf Dokumentáció

Mali Bence

Feladat: GNU make klón

A feladat egy GNU make-hez (innen egyszerűen make) hasonlóan működő program, a specifikációban felsorolt alapvető funkciókkal. A feladat elsősorban linux (esetleg más unix) rendszerekre készül.

Specifikáció:

A program futtatásakor a munkakönyvtárban keres egy Makefile (esetleg makefile) nevezetű fájlt, ami az eredeti make szabványoknak kell megfeleljen, de a funkcionalitása erősen korlátozott.

Támogatott funkciók:

- Target-prerequisites függőség (csak akkor fordítódik újra egy target, ha egy dependency később változott meg, mint a target létrejötte)

  targets …: prerequisites …

  recipe

- Az alapértelmezett cél az első szabály első targetje, a futtatáskor paraméterként a kívánt cél nevét átadhatjuk

  pl: ***$ ./makeclone clean***

- Változók létrehozása különböző hozzárendelésekkel

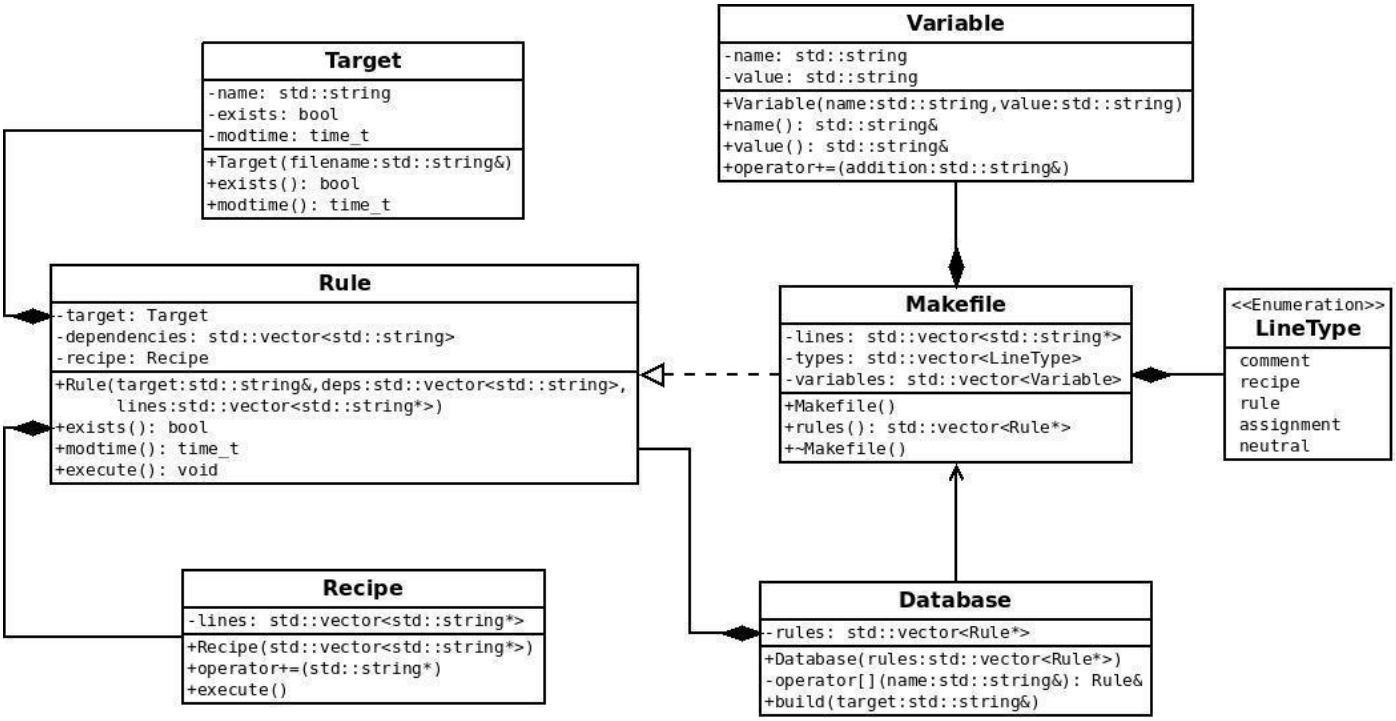
  =, ?=, +=

- Automatikus változók használata ***$@, $<, $^***

- Minta függőségek kezelése

  pl: ***%.o: %.c***

Osztálydiagram:



Működés:

A legfontosabb osztály a Makefile, amely példányosításnál egy Makefile vagy makefile elnevezésű fájlt keres, majd a beolvasott soraiból a hozzárendelések alapján létrehozza a Variable típusú változókat. A rules() tagfüggvény megkeresi a szabályokat a fájlban, majd behelyettesíti az ismert változókat és az összes szabályhoz létrehoz dinamikusan egy Rule objektumot, amelyek pointereit a visszatérési vektorban adja át. Az osztály destruktora felel a dinamikusan tárolt sorok és a pointerként átadott dinamikus Rule objektumok felszabadításáért.

A Rule osztály tárol egy targetet, a függőségek neveit és az előállítás parancsait. Az exists() tagfüggvény visszaadja, hogy a target fájl létezik-e, a modtime() tagfüggvény visszaadja a target fájl utolsó megváltoztatásának időpontját, az execute() tagfüggvény pedig végrehajtja a Recipe-ben szereplő parancsokat.

A Database osztály konstruktora egy (a Makefile osztály rules() tagfüggvényével megkapható) std::vector<Rule*> típusú vektort vesz át. Ez viselkedik “függőségi gráfként”. Az osztály operator[] tagfüggvénye a paraméterként megadott string alapján megkeresi a megegyező nevű target Rule objektum pointerét a vektorban, majd az objektumot referenciaként visszaadja. Amikor mi a program meghívásakor átadjuk az egyik target nevét, akkor a Database osztály build() tagfüggvénye hívódik meg ezzel a névvel.

A build() függvény pszeudokódja:

```
void build(target_name)
    target = this[target_name]
    for all dependencies of target:
        build(dependency_name)
    if !target.exists():
        target.execute()
    else:
        build_needed = false
        for all dependencies of target:
            if target.modtime() < dependency.modtime():
                build_needed = true
                break
        if build_needed:
            target.execute()
```

Ha olyan targetet akarunk build-elni, amelynek nem írtunk szabályt, akkor a build() függvény std::runtime_error kivételt dob, és a felhasználó a következő üzenetet látja: “No rule to make target ‘xxx’.”

Ha a build() függvény rekurzívan hívódik meg egy dependency-re, akkor a következő hibaüzenetet kapjuk: “No rule to make target ‘xxx’ needed by ‘yyy’.”

Megjegyzések:

A változó/target nevek nem tartalmazhatnak whitespace-t, ‘$’ és ‘:’ szimbólumokat.

A szabályok deklarációjánál vegyesen lehetnek minta függőségek és változós, kiírt target és függőség nevek. (pl: %.o main.o: %.cpp main.cpp)