

Նախագծման Ձևանմուշներ: Command

Հրաչյա Թանդիլյան

2020

Command

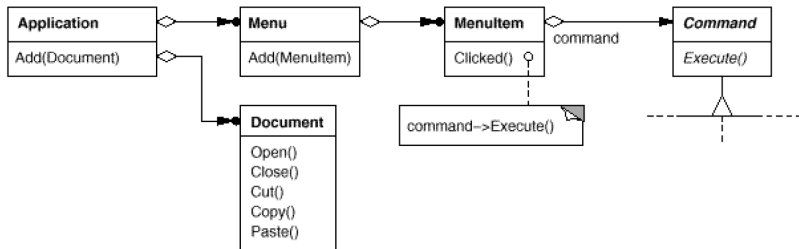
Նպատակը

Հարցումը ինկապտուլացնել օբյեկտի մեջ, դրանով իսկ թույլ տալ օբյեկտները տարբեր հարցումներով պարամետրիզացնել, կազմել հարցումների հերթեր, գրանցել հարցումները (log), իրականացնել վերականգնելի (undoable) գործողություններ:

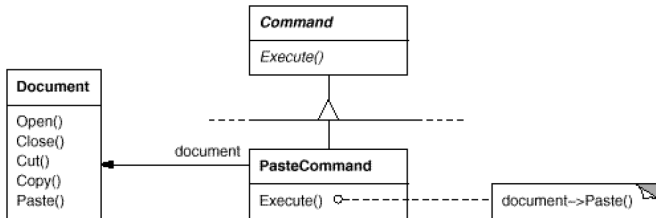
Նաև հայտնի է որպես

- Action
- Transaction

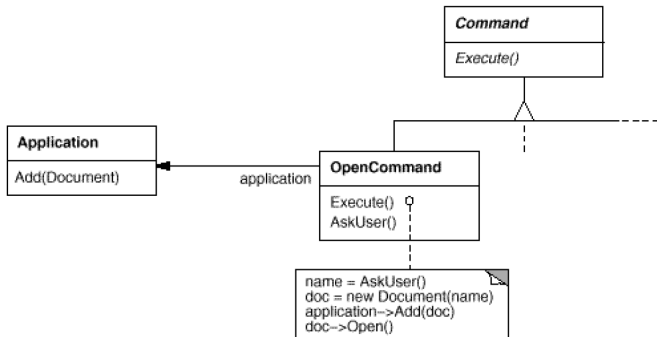
Մոտիվացիան



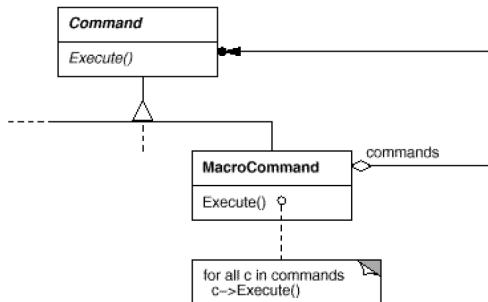
Մոտիվացիան



Մոտիվացիան



Մոտիվացիան



Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

- Անհրաժեշտ է օբյեկտները պարամետրիզացնել կատարվելիք գործողությունով: Հանդիսանում է պրոցեդուրալ լեզուներում կիրառվող callback ֆունկցիաի օբյեկտային համարժեքը:

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

- Ա** Անհրաժեշտ է օբյեկտները պարամետրիզացնել կատարվելիք գործողությունով: Հանդիսանում է պրոցեդուրալ լեզուներում կիրառվող callback ֆունկցիաի օբյեկտային համարժեքը:
- Բ** Անհրաժեշտ է հարցումները տալ, հավաքել և կատարել ժամանակի տարբեր պահերի:

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

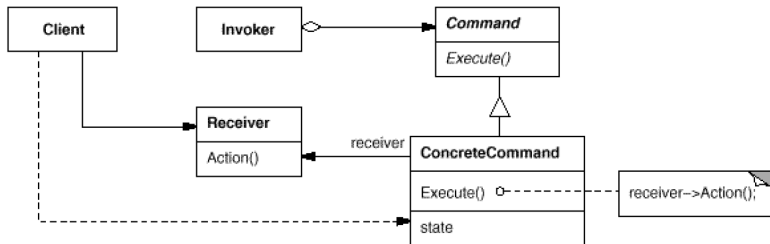
- Ա** Անհրաժեշտ է օբյեկտները պարամետրիզացնել կատարվելիք գործողությունով: Հանդիսանում է պրոցեդուրալ լեզուներում կիրառվող callback ֆունկցիաի օբյեկտային համարժեքը:
- Բ** Անհրաժեշտ է հարցումները տալ, հավաքել և կատարել ժամանակի տարբեր պահերի:
- Գ** Անհրաժեշտ է տրամադրել undo գործողություն:

Կիրառելիությունը

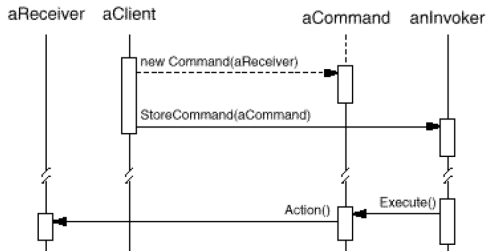
Այս Ն.Ձ. պետք է օգտագործել երբ.

- Ա** Անհրաժեշտ է օբյեկտները պարամետրիզացնել կատարվելիք գործողությունով: Հանդիսանում է պրոցեդուրալ լեզուներում կիրառվող callback ֆունկցիաի օբյեկտային համարժեքը:
- Բ** Անհրաժեշտ է հարցումները տալ, հավաքել և կատարել ժամանակի տարբեր պահերի:
- Գ** Անհրաժեշտ է տրամադրել undo գործողություն:
- Դ** Անհրաժեշտ է գրանցել փոփոխությունները այնպես, որ դրանք հնարավոր լինի կրկնել համակարգի աշխատանքի խափանման պարագայում:

Կառուցվածքը



Կառուցվածքը



Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Ա Առանձնացնում է գործողության կանչն իրականացնող օբյեկտին այն օբյեկտից, որը գիտի թե ինչպես գործողությունը կատարել:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Ա** Առանձնացնում է գործողության կանչն իրականացնող օբյեկտին այն օբյեկտից, որը գիտի թե ինչպես գործողությունը կատարել:
- Բ** Քանի որ գործողությունը մոդելավորվում է սովորական օբյեկտի միջոցով, այն կարելի է դեկլարել և ընդլայնել այլ օբյեկտների նման:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Ա** Առանձնացնում է գործողության կանչն իրականացնող օբյեկտին այն օբյեկտից, որը գիտի թե ինչպես գործողությունը կատարել:
- Բ** Քանի որ գործողությունը մոդելավորվում է սովորական օբյեկտի միջոցով, այն կարելի է դեկլարել և ընդլայնել այլ օբյեկտների նման:
- Գ** Հրամաններ կարելի է հավաքել կոմպոզիտ հրամանների մեջ:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Ա** Առանձնացնում է գործողության կանչն իրականացնող օբյեկտին այն օբյեկտից, որը գիտի թե ինչպես գործողությունը կատարել:
- Բ** Զանի որ գործողությունը մոդելավորվում է սովորական օբյեկտի միջոցով, այն կարելի է դեկավարել և ընդլայնել այլ օբյեկտների նման:
- Գ** Հրամաններ կարելի է հավաքել կոմպոզիտ հրամանների մեջ:
- Դ** Նոր տիպի գործողությունների ավելացումը հեշտ է, քանի որ գոյություն ունեցող գործողությունները փոփոխելու անհրաժեշտություն չկա:

Իրականացումը

- Ա Որքան խելացի պետք է լինի Command-ը:
- Բ Undo / Redo գործողությունների իրականացում:
- Գ Undo / Redo գործողությունների կատարման ժամանակ սխալի կուտակում:
- Դ C++-ի template-ների կիրառում:

Օրինակ

```
class Command {  
  
    public:  
        virtual ~Command();  
        virtual void Execute() = 0;  
  
    protected:  
        Command();  
};
```

Օրինակ

```
class OpenCommand : public Command {  
  
public:  
    OpenCommand(Application* a) : application(a) {}  
    virtual void Execute() {  
        const char* name = AskUser();  
        if (name == 0) return;  
  
        Document* document = new Document(name);  
        application->Add(document);  
        document->Open();  
    }  
  
protected:  
    virtual const char* AskUser();  
  
private:  
    Application* application;  
    char* response;  
};
```

Օրինակ

```
class PasteCommand : public Command {  
  
public:  
    PasteCommand(Document* d) : document(d) {}  
  
    void PasteCommand::Execute () {  
        document->Paste();  
    }  
  
private:  
    Document* document;  
};
```

Օրինակ

```
template <class Receiver>
class SimpleCommand : public Command {

public:
    typedef void (Receiver::* Action) ();

    SimpleCommand(Receiver* r, Action a)
        : receiver(r), action(a) {}

    virtual void Execute() {
        (receiver->*action)();
    }

private:
    Action action;
    Receiver* receiver;
};
```

Օրինակ

```
MyClass* receiver = new MyClass;
```

```
Command* aCommand = new SimpleCommand<MyClass>(receiver,  
                                                &MyClass::Action);  
aCommand->Execute();
```

Օրինակ

```
class MacroCommand : public Command {  
  
public:  
    MacroCommand();  
    virtual ~MacroCommand();  
  
    virtual void Add(Command* c) { cmds->Append(c); }  
    virtual void Remove(Command* c) { cmds->Remove(c); }  
  
    virtual void Execute() {  
        ListIterator<Command*> i(cmds);  
        for (i.First(); !i.IsDone(); i.Next()) {  
            Command* c = i.CurrentItem();  
            c->Execute();  
        }  
    }  
  
private:  
    List<Command*>* cmds;  
};
```

Առնչվող Նախագծման Ձևանմուշները

- Composite
- Memento
- Prototype