

Նախագծման Ձևանմուշներ: Prototype

Հրաչյա Թանդիլյան

2020

Prototype

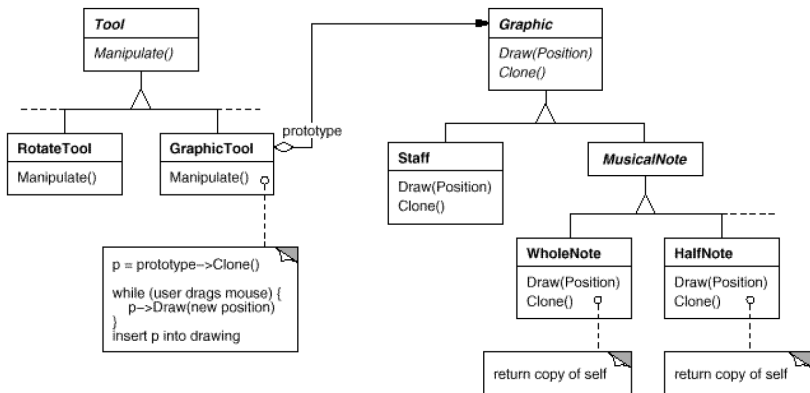
Նպատակը

Նշում է ստեղծվելիք օբյեկտի տիպը օգտագործելով նախատիպային օբիեկտ (instance) և ստեղծում է նոր օբյեկտներ այն պատճենելով:

Նաև հայտնի է որպես

- Virtual Copy Constructor

Մոտիվացիան



Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

- Համակարգը պետք է անկախ լինի օբյեկտների ստեղծման և ներկայացման ձևից:

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

- Ա Համակարգը պետք է անկախ լինի օբյեկտների ստեղծման և ներկայացման ձևից:
- Բ Անհրաժեշտ է խուսափել օբյեկտների հիերարխիային զուգահեռ Factory դասերի հիերարխիա կառուցելուց:

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

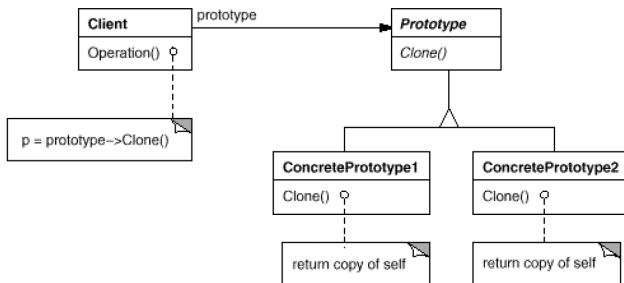
- Ա** Համակարգը պետք է անկախ լինի օբյեկտների ստեղծման և ներկայացման ձևից:
- Բ** Անհրաժեշտ է խուսափել օբյեկտների հիերարխիային զուգահեռ Factory դասերի հիերարխիա կառուցելուց:
- Գ** Երբ ստեղծվելիք օբյեկտի տիպը նշվում է ծրագրի կատարման ընթացքում (օրինակ ներբեռնելով)

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

- Ա** Համակարգը պետք է անկախ լինի օբյեկտների ստեղծման և ներկայացման ձևից:
- Բ** Անհրաժեշտ է խուսափել օբյեկտների հիերարխիային զուգահեռ Factory դասերի հիերարխիա կառուցելուց:
- Գ** Երբ ստեղծվելիք օբյեկտի տիպը նշվում է ծրագրի կատարման ընթացքում (օրինակ ներբեռնելով)
- Դ** Երբ օբյեկտը կարող է ունենալ քիչ քանակով իրարից տարբեր ներքին վիճակներ երբեմն ավելի նպատակահարմար է նախօրոք ստեղծել բոլոր իրարից տարբեր օբյեկտները և անհրաժեշտության դեպքում դրանք պատճենել նոր օբյեկտ ստեղծելու փոխարեն:

Կառուցվածքը



Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

Ա Առանձնացնում է կոնկրետ դասերը:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Ա Առանձնացնում է կոնկրետ դասերը:
- Բ Ստեղծել նոր օբյեկտներ փոփոխելով արժեքները:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Ա Առանձնացնում է կոնկրետ դասերը:
- Բ Ստեղծել նոր օբյեկտներ փոփոխելով արժեքները:
- Գ Ստեղծել նոր օբյեկտներ փոփոխելով կառուցվածքը:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Ա Առանձնացնում է կոնկրետ դասերը:
- Բ Ստեղծել նոր օբյեկտներ փոփոխելով արժեքները:
- Գ Ստեղծել նոր օբյեկտներ փոփոխելով կառուցվածքը:
- Դ Նվազեցնել ենթադասերի քանակը:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Ա Առանձնացնում է կոնկրետ դասերը:
- Բ Ստեղծել նոր օբյեկտներ փոփոխելով արժեքները:
- Գ Ստեղծել նոր օբյեկտներ փոփոխելով կառուցվածքը:
- Դ Նվազեցնել ենթադասերի քանակը:
- Ե Դիսամիկ կերպով կոնֆիգուրացնել կիրառությունը:

Իրականացումը

Ա Օգտագործելով prototype manager:

Իրականացումը

Ա Օգտագործելով prototype manager:

Բ Clone գործողության իրականացումը:

Իրականացումը

- Ա Օգտագործելով prototype manager:
- Բ Clone գործողության իրականացումը:
- Գ Պատճենների ինիցիալիզացիան:

Օրինակ

```
class MazePrototypeFactory : public MazeFactory {  
  
public:  
    MazePrototypeFactory(Maze*, Wall*, Room*, Door*);  
  
    virtual Maze* MakeMaze() const;  
  
    virtual Room* MakeRoom(int) const;  
  
    virtual Wall* MakeWall() const;  
  
    virtual Door* MakeDoor(Room*, Room*) const;  
  
private:  
    Maze* prototypeMaze; Room* prototypeRoom;  
    Wall* prototypeWall; Door* prototypeDoor;  
};
```

Օրինակ

```
MazePrototypeFactory::MazePrototypeFactory(Maze* m, Wall* w,  
                                              Room* r, Door* d) {  
    prototypeMaze = m; prototypeWall = w;  
    prototypeRoom = r; prototypeDoor = d;  
}
```

```
Wall* MazePrototypeFactory::MakeWall() const {  
  
    return prototypeWall->Clone();  
}
```

```
Door* MazePrototypeFactory::MakeDoor(Room* r1, Room *r2) const {  
  
    Door* door = prototypeDoor->Clone();  
    door->Initialize(r1, r2);  
    return door;  
}
```

Օրինակ

```
MazeGame game;  
MazePrototypeFactory simpleMazeFactory(new Maze, new Wall,  
                                         new Room, new Door);
```

```
Maze* maze = game.CreateMaze(simpleMazeFactory);
```

```
MazeGame game;  
MazePrototypeFactory bombedMazeFactory(new Maze, new BombedWall,  
                                         new RoomWithABomb, new Door);
```

```
Maze* maze = game.CreateMaze(bombedMazeFactor);
```

Օրինակ

```
class Door : public MapSite {  
  
public:  
    Door();  
  
    Door(const Door& other) ) {  
        room1 = other.room1; room2 = other.room2;  
    }  
  
    virtual void Initialize(Room* r1, Room* r2) ) {  
        room1 = r1; room2 = r2;  
    }  
  
    virtual Door* Clone() const { return new Door(*this);}  
  
    virtual void Enter();  
  
private:  
    Room* room1; Room* room2;  
};
```

Առնչվող Նախագծման Ձևանմուշները

- Abstract Factory
- Composite
- Decorator