

Նախագծման Ձևանմուշներ: AbstractFactory

Հրաչյա Թանդիլյան

2020

AbstractFactory

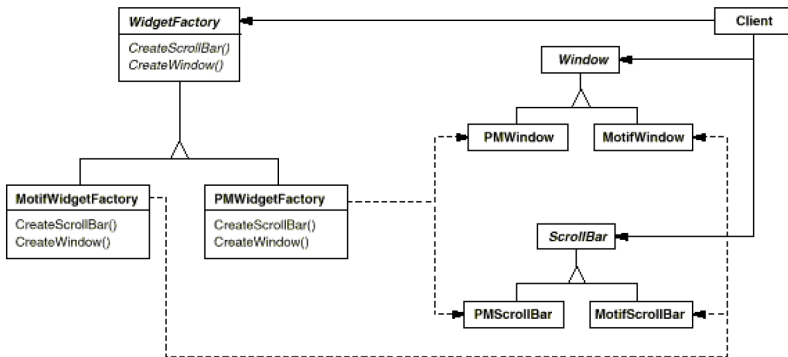
Նպատակը

Տրամադրում է ինտերֆեյս միմյանց հետ կապված կամ միմյանցից կախված օբյեկտների ընտանիքներ ստեղծելու համար, առանց նշելու նրանց կոնկրետ տիպը:

Նաև հայտնի է որպես

- Kit

Մոտիվացիան



Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

- Համակարգը պետք է անկախ լինի իր կոմպոնենտների ստեղծումից և ներկայացումից:

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

- Ա** Համակարգը պետք է անկախ լինի իր կոմպոնենտների ստեղծումից և ներկայացումից:
- Բ** Համակարգը պետք է աշխատի կոմպոնենտների բազմաթիվ ընտանիքներից միայն մեկի հետ:

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

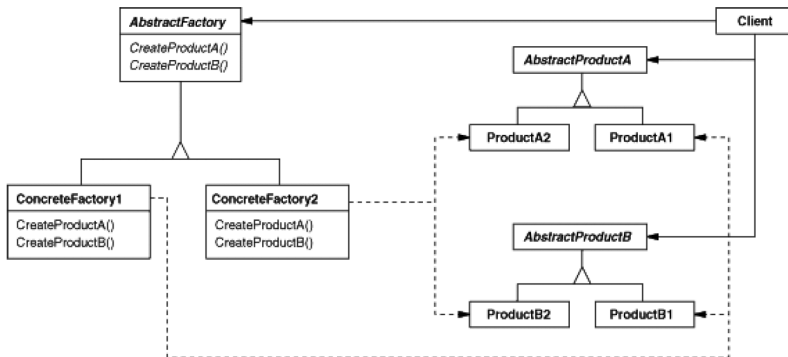
- Ա** Համակարգը պետք է անկախ լինի իր կոմպոնենտների ստեղծումից և ներկայացումից:
- Բ** Համակարգը պետք է աշխատի կոմպոնենտների բազմաթիվ ընտանիքներից միայն մեկի հետ:
- Գ** Անհրաժեշտ է ստիպել, որ միևնույն ընտանիքի կոմպոնենտներին համապատասխան օբյեկտները աշխատեն միայն միմյանց հետ:

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

- Ա** Համակարգը պետք է անկախ լինի իր կոմպոնենտների ստեղծումից և ներկայացումից:
- Բ** Համակարգը պետք է աշխատի կոմպոնենտների բազմաթիվ ընտանիքներից միայն մեկի հետ:
- Գ** Անհրաժեշտ է ստիպել, որ միևնույն ընտանիքի կոմպոնենտներին համապատասխան օբյեկտները աշխատեն միայն միմյանց հետ:
- Դ** Անհրաժեշտ է տրամադրել կոմպոնենտների դասերի գրադարան, այնպես որ բացահայտվի միայն նրանց ինտերֆեյսը և ոչ թե իրականացումը:

Կառուցվածքը



Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

Ա Առանձնացնում է կոնկրետ դասերը:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Առանձնացնում է կոնկրետ դասերը:
- Հեշտացնում է կոմպոնենտների ընտանիքների փոխարինումը:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Առանձնացնում է կոնկրետ դասերը:
- Հեշտացնում է կոմպոնենտների ընտանիքների փոխարինումը:
- Ապահովում է կայունություն կոմպոնենտների միջև:

Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

- Առանձնացնում է կոնկրետ դասերը:
- Հեշտացնում է կոմպոնենտների ընտանիքների փոխարինումը:
- Ապահովում է կայունություն կոմպոնենտների միջև:
- Դժվարեցնում է նոր տիպի կոմպոնենտների ավելացումը:

Իրականացումը

Ա Factory-ները որպես Singleton:

Իրականացումը

Ա Factory-ները որպես Singleton:

Բ Կոմպոնենտների ստեղծումը:

Իրականացումը

Ա Factory-ները որպես Singleton:

Բ Կոմպոնենտների ստեղծումը:

Գ Ընդլայնելի Factory-ների ստեղծումը:

Օրինակ

```
class MazeFactory {  
  
    public:  
        MazeFactory();  
  
        virtual Maze* MakeMaze() const { return new Maze; }  
  
        virtual Wall* MakeWall() const { return new Wall; }  
  
        virtual Room* MakeRoom(int n) const { return new Room(n); }  
  
        virtual Door* MakeDoor(Room* r1, Room* r2) const {  
            return new Door(r1, r2);  
        }  
};
```


Օրինակ

```
Maze* MazeGame::CreateMaze(MazeFactory& factory) {  
    Maze* aMaze = factory.MakeMaze();  
  
    Room* r1 = factory.MakeRoom(1);  
    Room* r2 = factory.MakeRoom(2);  
    aMaze->AddRoom(r1); aMaze->AddRoom(r2);  
  
    Door* aDoor = factory.MakeDoor(r1, r2);  
    r1->SetSide(East, aDoor); r2->SetSide(West, aDoor);  
  
    r1->SetSide(North, factory.MakeWall());  
    r1->SetSide(South, factory.MakeWall());  
    r1->SetSide(West, factory.MakeWall());  
    r2->SetSide(North, factory.MakeWall());  
    r2->SetSide(East, factory.MakeWall());  
    r2->SetSide(South, factory.MakeWall());  
  
    return aMaze;  
}
```

Օրինակ

```
class EnchantedMazeFactory : public MazeFactory {  
  
public:  
    EnchantedMazeFactory();  
  
    virtual Room* MakeRoom(int n) const {  
        return new EnchantedRoom(n, CastSpell());  
    }  
  
    virtual Door* MakeDoor(Room* r1, Room* r2) const {  
        return new DoorNeedingSpell(r1, r2);  
    }  
  
protected:  
    Spell* CastSpell() const;  
};
```

Օրինակ

```
class BombedMazeFactory : public MazeFactory {  
  
public:  
    BombedMazeFactory();  
  
    virtual Room* MakeRoom(int n) const {  
        return new RoomWithABomb(n);  
    }  
  
    Wall* MakeWall () const {  
        return new BombedWall;  
    }  
};  
  
MazeGame game;  
BombedMazeFactory factory;  
game.CreateMaze(factory);
```

Առնչվող Նախագծման Ձևանմուշները

- Factory Method

- Prototype

- Singleton