Նախագծման Ձևանմուշներ: Proxy

Հրաչյա Թանդիլյան

2020

Proxy

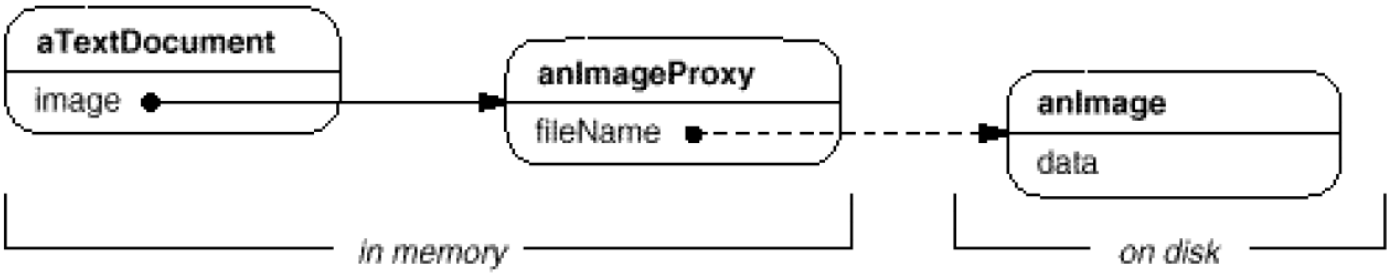
Նպատակը

Տվյալ օբյեկտի համար տրամադրել այնպիսի փոխարինող, որը կվերահսկի նրան դիմումը:

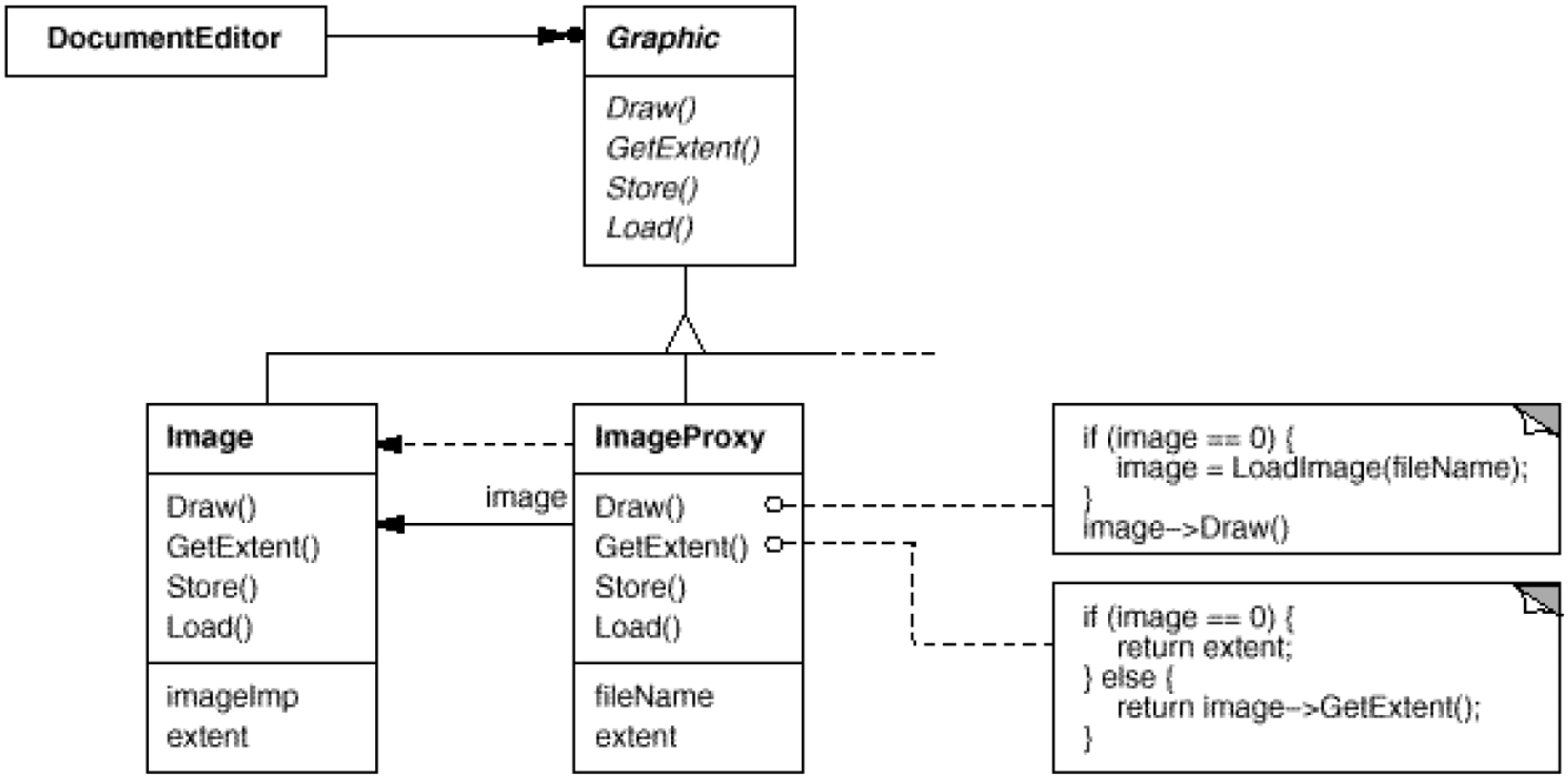
Նաև հայտնի է որպես

- Surrogate

Մոտիվացիան


aTextDocument
image
anImageProxy
fileName
anImage
data
in memory
on disk

Մոտիվացիան


DocumentEditor
Graphic
Draw()
GetExtent()
Store()
Load()
Image
Draw()
GetExtent()
Store()
Load()
imageImp
extent
image
ImageProxy
Draw()
GetExtent()
Store()
Load()
fileName
extent
if (image == 0) {
image = LoadImage(fileName);
}
image->Draw()
if (image == 0) {
return extent;
} else {
return image->GetExtent();
}

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

- Remote Proxy

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

Ա Remote Proxy

Բ Virtual Proxy

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

Ա Remote Proxy

Բ Virtual Proxy

Գ Protection Proxy

Կիրառելիությունը

Այս Ն.Ձ. պետք է օգտագործել երբ.

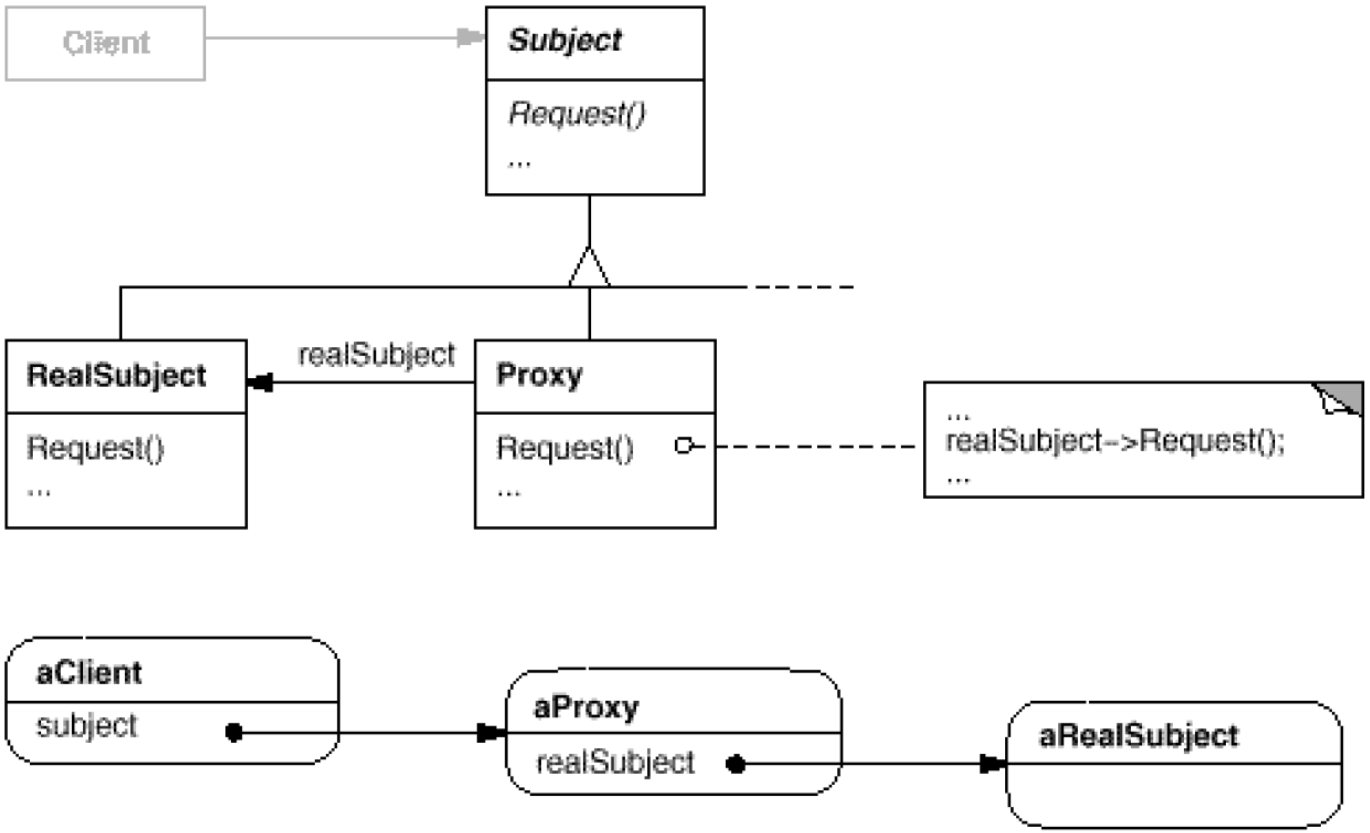
Ա Remote Proxy

Բ Virtual Proxy

Գ Protection Proxy

Դ Smart Reference

Կառուցվածքը



Հետևանքները

Այս Ն.Ձ. ունի հետևյալ առավելություններն ու թերությունները.

Ա Վերհղման (indirection) մակարդակ է ավելացնում: Կախված Proxy-ի տարատեսակից այդ վերհղումը ունի տարբեր նպատակներ

- remote proxy-ին թաքցնում է այն փաստը, որ հղվող օբյեկտը գտնվում է այլ համակարգչի վրա:
- virtual proxy-ին հնարավորություն է տալիս օբյեկտը ստեղծել առաջին դիմման ժամանակ:
- protection proxy-ին և smart reference-ները թույլ են տալիս օբյեկտին դիմելիս հավելյալ գործողություններ կատարել:

Բ copy-on-write օպտիմիզացիաի հնարավորություն է տալիս:

Իրականացումը

Ա. Անդամներին դիմման օպերատորի (operator *, operator ->) վերաբեռնում (C++):

Բ. Proxy-ն պարտադիր չէ, որ իմանա իրական օբյեկտի տիպը:

Օրինակ

```
class Graphic {

public:
    virtual ~Graphic();
    virtual void Draw(const Point& at) = 0;
    virtual void HandleMouse(Event& event) = 0;
    virtual const Point& GetExtent() = 0;
    virtual void Load(istream& from) = 0;
    virtual void Save(ostream& to) = 0;

protected:
    Graphic();
};
```

Օրինակ

```
class Image : public Graphic {

public:
    Image(const char* file); // loads image from a file
    virtual ~Image();
    virtual void Draw(const Point& at);
    virtual void HandleMouse(Event& event);
    virtual const Point& GetExtent();
    virtual void Load(istream& from);
    virtual void Save(ostream& to);

private:
    // private fields
};
```

Օրինակ

```
class ImageProxy : public Graphic {

public:
    ImageProxy(const char* imageFile);
    virtual ~ImageProxy();
    virtual void Draw(const Point& at);
    virtual void HandleMouse(Event& event);
    virtual const Point& GetExtent();
    virtual void Load(istream& from);
    virtual void Save(ostream& to);

protected:
    Image* GetImage();

private:
    Image* image;
    Point extent;
    char* fileName;
};
```

Օրինակ

```
ImageProxy::ImageProxy(const char* f) {
    fileName = strdup(f);
    extent = Point::Zero;
    image = 0;
}

Image* ImageProxy::GetImage() {
    if (image == NULL) {
        image = new Image(fileName);
    }
    return image;
}

const Point& ImageProxy::GetExtent() {
    if (extent == Point::Zero) {
        extent = GetImage()->GetExtent();
    }
    return extent;
}
```

Օրինակ

```
void ImageProxy::Draw(const Point& at) {
    GetImage()->Draw(at);
}

void ImageProxy::HandleMouse(Event& event) {
    GetImage()->HandleMouse(event);
}

void ImageProxy::Save(ostream& to) {
    to << extent << fileName;
}

void ImageProxy::Load(istream& from) {
    from >> extent >> fileName;
}
```

Օրինակ

```
class TextDocument {

public:
    TextDocument();
    void Insert(Graphic*);
    // other methods
};

TextDocument* text = new TextDocument;
text->Insert(new ImageProxy("anImageFileName"));
```

Առնչվող Նախագծման Ձևանմուշները

- Adapter
- Decorator