ES6+ DESIGN PATTERN VIEW PATTERN

DESIGN PATTERN CATEGORY

GOF의 디자인패턴 분류

생성패턴
구조패턴
행동패턴

GOF의 디자인패턴 분류

생성패턴
구조패턴
행동패턴

이미 객체지향설계에 숙달한 사람을 대상으로 함

객체지향설계를 학습할 수 있는 분류

캡슐화

다형성

객체 관계

변화율

역할모델

객체지향설계를 학습할 수 있는 분류

다형성

캡슐화

객체 관계

변화율

역할모델

알고리즘이 변화하는 이유?

비지니스변화

연관 라이브러리 변화

호스트측 변화

알고리즘이 변화하는 이유?

비지니스변화

연관 라이브러리 변화

호스트측 변화

대부분 통제불가요소

기존 제어문 기반의 알고리즘이 갖는 문제

"수정하면 전체가 컴파일 됨"

기존 제어문 기반의 알고리즘이 갖는 문제

"수정하면 전체가 컴파일 됨"

알고리즘이 변화한 부분만 수정하고
나머지는 건드리고 싶지 않다면?

기존 제어문 기반의 알고리즘이 갖는 문제

"수정하면 전체가 컴파일 됨"

알고리즘이 변화한 부분만 수정하고
나머지는 건드리고 싶지 않다면?

최대한 개별 알고리즘을 함수로 분리한다.

```
if(case == 1){
   .......
}else if(case == 2){
   .......
}else if(case == 3){
   .......
}
```

최대한 개별 알고리즘을 함수로 분리한다.

```
if(case == 1){
   .......
}else if(case == 2){
   .......
}else if(case == 3){
   .......
}
```

최대한 개별 알고리즘을 함수로 분리한다.

```
if(case == 1){
   .......
}else if(case == 2){
   .......
}else if(case == 3){
   .......
}
```

```
if(case == 1){
   case1();
}else if(case == 2){
   case2();
}else if(case == 3){
   case3();
}
```

최대한 개별 알고리즘을 함수로 분리한다.

```
if(case == 1){
   .......
}else if(case == 2){
   .......
}else if(case == 3){
   .......
}
```

```
if(case == 1){
   case1();
}else if(case == 2){
   case2();
}else if(case == 3){
   case3();
}
```

최대한 개별 알고리즘을 함수로 분리한다.

```
if(case == 1){
   case1();
}else if(case == 2){
   case2();
}else if(case == 3){
   case3();
}else if(case == 4){
   case4();
}
```

문제1. 경우가 변경될 때

최대한 개별 알고리즘을 함수로 분리한다.

```
if(case == 1){
   case1();
}else if(case == 2){
   case2();
}else if(case == 3){
   case3();
}else if(case == 4){
   case4();
}
```

```
if(case == 1){
   common();
   case1();
}else if(case == 2){
   common();
   case2();
}else if(case == 3){
   case3();
}
```

문제1. 경우가 변경될 때
문제2. 함수 간 공통부분

최대한 개별 알고리즘을 함수로 분리한다.

알고리즘 분화 시 객체지향에서 선택할 수 있는 두 가지 방법

알고리즘 분화 시 객체지향에서 선택할 수 있는 두 가지 방법

상속 위임

내부계약관계로 추상층에서
공통 요소를 해결하고
상태를 공유할 수 있음

알고리즘 분화 시 객체지향에서 선택할 수 있는 두 가지 방법

상속 위임

내부계약관계로 추상층에서
공통 요소를 해결하고
상태를 공유할 수 있음

소유 위임

외부계약관계로 각각이
독립적인 문제를 해결하며
메세지를 주고 받는 것으로
문제를 해결함

알고리즘 분화 시 객체지향에서 선택할 수 있는 두 가지 방법

상속 위임

내부계약관계로 추상층에서
공통 요소를 해결하고
상태를 공유할 수 있음

소유 위임

외부계약관계로 각각이
독립적인 문제를 해결하며
메세지를 주고 받는 것으로
문제를 해결함

GoF DP 방향성

부가적인 형이 많이 생김

상속 위임

```
const Github = class{
   constructor(id, repo){
      this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
   }
   load(path){
      const id = 'callback' + Github._id++;
      const f = Github[id] = ({data:{content}})=>{
         delete Github[id];
         document.head.removeChild(s);
         this._loaded(content);
      };
      const s = document.createElement('script');
      s.src = `${this._base + path}?callback=Github. ${id}`;
      document.head.appendChild(s);
   }
   _loaded(v){throw 'override!';}
};
Github._id = 0;
```

상속 위임

```
const Github = class{
   constructor(id, repo){
      this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
   }
   load(path){
      const id = 'callback' + Github._id++;
      const f = Github[id] = ({data:{content}})=>{
         delete Github[id];
         document.head.removeChild(s);
         this._loaded(content);
      };
      const s = document.createElement('script');
      s.src = `${this._base + path}?callback=Github. ${id}`;
      document.head.appendChild(s);
   }
   _loaded(v){throw 'override!';}
};
Github._id = 0;
const ImageLoader = class extends Github{
   _loaded(v){...}
}
```

상속 위임

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
    }
    load(path){
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._loaded(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    _loaded(v){throw 'override!';}
};
Github._id = 0;
const ImageLoader = class extends Github{
    _loaded(v){...}
}
```

공통부분

상속 위임

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
    }
    load(path){
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._loaded(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    _loaded(v){throw 'override!';}
};
Github._id = 0;
const ImageLoader = class extends Github{
    _loaded(v){...}
}
```

공통부분

위임부분

상속 위임

TEMPLATE METHOD PATTERN

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
    }
    load(path){
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._loaded(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    _loaded(v){throw 'override!';}
};
Github._id = 0;
const ImageLoader = class extends Github{
    _loaded(v){...}
}
```

TEMPLATE METHOD

공통부분

위임부분

HOOK

위임구현

상속 위임

TEMPLATE METHOD PATTERN

정의시점 - 변하지 않는 부분

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
    }
    load(path){
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._loaded(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    _loaded(v){throw 'override!';}
};
Github._id = 0;
const ImageLoader = class extends Github{
    _loaded(v){...}
}
```

TEMPLATE METHOD

공통부분

위임부분

HOOK

실행시점선택지 - 변하는 부분

위임구현

상속 위임

TEMPLATE METHOD PATTERN

정의시점 - 변하지 않는 부분

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
    }
    load(path){
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._loaded(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    _loaded(v){throw 'override!';}
};
Github._id = 0;
const ImageLoader = class extends Github{
    _loaded(v){...}
}
```

TEMPLATE METHOD

공통부분

위임부분

HOOK

실행시점 - 변하는 부분

위임구현

ImageLoader

```
const ImageLoader = class extends Github{
    constructor(id, repo, target){
        super(id, repo);
        this._target = target;
    }
    _loaded(v){
        this._target.src = 'data:text/plain;base64,' + v;
    }
};
```

```
const Github = class{
     constructor(id, repo){
          this._base = `https://api.github.com/repo... `;
     }
     load(path){
          const id = 'callback' + Github._id++;
          const f = Github[id] = ({data:{content}})=>{
               delete Github[id];
               document.head.removeChild(s);
               this._loaded(content);
          };
          const s = document.createElement('script');
          s.src = `${this._base + path}?callback=Github. ${id}`;
          document.head.appendChild(s);
     }
     _loaded(v){throw 'override!';}
};
Github._id = 0;
```

ImageLoader

```
const ImageLoader = class extends Github{
    constructor(id, repo, target){
        super(id, repo);
        this._target = target;
    }
    _loaded(v){
        this._target.src = 'data:text/plain;base64,' + v;
    }
};

const s75img = new ImageLoader(
    'hikaMaeng',
    'codespitz75',
    document.querySelector('#a')
);
s75img.load('einBig.png');
```

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repo... `;
    }
    load(path){
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._loaded(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    _loaded(v){throw 'override!';}
};
Github._id = 0;
```

MdLoader

```
const MdLoader = class extends Github{
    constructor(id, repo, target){
        super(id, repo);
        this._target = target;
    }
    _loaded(v){
        this._target.innerHTML = this._parseMD(v);
    }
    _parseMD(v){
        return d64(v).split('\n').map(v=>{
            let i = 3;
            while(i--){
                if(v.startsWith('#'.repeat(i + 1))) return `<h${i + 1}>${v.substr(i + 1)}</h${i + 1}>`;
            }
            return v;
        }).join('<br>');
    }
};
const d64 =v=>decodeURIComponent(
    atob(v).split('').map(c=>'%' + ('00' +c.charCodeAt(0).toString(16)).slice(-2)).join('')
);
```

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repo... `;
    }
    load(path){
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._loaded(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    _loaded(v){throw 'override!';}
};
Github._id = 0;
```

MdLoader

```
const MdLoader = class extends Github{
    constructor(id, repo, target){
        super(id, repo);
        this._target = target;
    }
    _loaded(v){
        this._target.innerHTML = this._parseMD(v);
    }
    _parseMD(v){
        return d64(v).split('\n').map(v=>{
            let i = 3;
            while(i--){
                if(v.startsWith('#'.repeat(i + 1))) return `<h${i + 1}>${v.substr(i + 1)}</h${i + 1}>`;
            }
            return v;
        }).join('<br>');
    }
};
const d64 =v=>decodeURIComponent(
    atob(v).split('').map(c=>'%' + ('00' +c.charCodeAt(0).toString(16)).slice(-2)).join('')
);

const s75md = new MdLoader('hikaMaeng', 'codespitz75', document.querySelector('#b'));
s75md.load('README.md');
```

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repo... `;
    }
    load(path){
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._loaded(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    _loaded(v){throw 'override!';}
};
Github._id = 0;
```

정의시점과 실행시점

```
<script src="Github.js"></script>
<script src="ImageLoader.js"></script>
<script src="MdLoader.js"></script>
<script>
const img = new ImageLoader(...);
img.load(...);

const md = new MdLoader(...);
md.load(...);
</script>
```

정의시점

실행시점선택

소유 위임

```
const Github = class{
   constructor(id, repo){
      this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
   }
   load(path){
      if(!this._parser) return;
      const id = 'callback' + Github._id++;
      const f = Github[id] = ({data:{content}})=>{
         delete Github[id];
         document.head.removeChild(s);
         this._parser(content);
      };
      const s = document.createElement('script');
      s.src = `${this._base + path}?callback=Github. ${id}`;
      document.head.appendChild(s);
   }
   set parser(v){this._parser = v;}
};
Github._id = 0;
```

소유 위임

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
    }
    load(path){
        if(!this._parser) return;
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._parser(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    set parser(v){this._parser = v;}
};
Github._id = 0;
```

위임부분

소유 위임

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
    }
    load(path){
        if(!this._parser) return;
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._parser(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    set parser(v){this._parser = v;}
};
Github._id = 0;
```

위임부분

위임객체

소유 위임

STRATEGY PATTERN

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
    }
    load(path){
        if(!this._parser) return;
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._parser(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    set parser(v){this._parser = v;}
};
Github._id = 0;
```

위임부분

위임객체

STARATEGY OBJECT

소유 위임

STRATEGY PATTERN

정의시점 - 변하지 않는 부분

```
const Github = class{
    constructor(id, repo){
        this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
    }
    load(path){
        if(!this._parser) return;
        const id = 'callback' + Github._id++;
        const f = Github[id] = ({data:{content}})=>{
            delete Github[id];
            document.head.removeChild(s);
            this._parser(content);
        };
        const s = document.createElement('script');
        s.src = `${this._base + path}?callback=Github. ${id}`;
        document.head.appendChild(s);
    }
    set parser(v){this._parser = v;}
};
Github._id = 0;
```

위임부분

위임객체

STARATEGY OBJECT

실행시점선택지 - 변하는 부분

Concrete Loader

```
const el =v=>document.querySelector(v);
const parseMD =v=>...;
const loader = new Github('hikaMaeng', 'codespitz75');
```

```
const Github = class{
  constructor(id, repo, parser){
    this._base = `https://api.github.com/repos/...`;
    this.parser = parser;
  }
  load(path){
    const id = 'callback' + Github._id++;
    const f = Github[id] = ({data:{content}})=>{
      delete Github[id];
      document.head.removeChild(s);
      this._parser(content);
    };
    const s = document.createElement('script');
    s.src = `${this._base + path}?callback=Github. ${id}`;
    document.head.appendChild(s);
  }
  set parser(v){this._parser = v;}
};
Github._id = 0;
```

Concrete Loader

```
const el =v=>document.querySelector(v);
const parseMD =v=>...;
const loader = new Github('hikaMaeng', 'codespitz75');

//img
const img =v=>el('#a').src = 'data:text/plain;base64,' + v;
loader.parser = img;
loader.load('xx.png');
```

```
const Github = class{
  constructor(id, repo, parser){
    this._base = `https://api.github.com/repos/...`;
    this.parser = parser;
  }
  load(path){
    const id = 'callback' + Github._id++;
    const f = Github[id] = ({data:{content}})=>{
      delete Github[id];
      document.head.removeChild(s);
      this._parser(content);
    };
    const s = document.createElement('script');
    s.src = `${this._base + path}?callback=Github. ${id}`;
    document.head.appendChild(s);
  }
  set parser(v){this._parser = v;}
};
Github._id = 0;
```

Concrete Loader

```
const el =v=>document.querySelector(v);
const parseMD =v=>...;
const loader = new Github('hikaMaeng', 'codespitz75');

//img
const img =v=>el('#a').src = 'data:text/plain;base64,' + v;
loader.parser = img;
loader.load('xx.png');

//md
const md =v=>el('#b').innerHTML = parseMD(v);
loader.parser = md;
loader.load('xx.md');
```

```
const Github = class{
  constructor(id, repo, parser){
    this._base = `https://api.github.com/repos/...`;
    this.parser = parser;
  }
  load(path){
    const id = 'callback' + Github._id++;
    const f = Github[id] = ({data:{content}})=>{
      delete Github[id];
      document.head.removeChild(s);
      this._parser(content);
    };
    const s = document.createElement('script');
    s.src = `${this._base + path}?callback=Github. ${id}`;
    document.head.appendChild(s);
  }
  set parser(v){this._parser = v;}
};
Github._id = 0;
```

자유변수를 통한 확장

```
const Github = class{
   constructor(id, repo){
      this._base = `https://api.github.com/repos/${id}/${repo}/contents/`;
   }
   load(path){
      if(!this._parser) return;
      const id = 'callback' + Github._id++;
      const f = Github[id] = ({data:{content}})=>{
         delete Github[id];
         document.head.removeChild(s);
         this._parser[0](content, ...this._parser[1]);
      };
      const s = document.createElement('script');
      s.src = `${this._base + path}?callback=Github. ${id}`;
      document.head.appendChild(s);
   }
   setParser(f, ...arg){this._parser = [f, arg];}
};
Github._id = 0;
```

Concreate Loader

```
const el =v=>document.querySelector(v);
const parseMD =v=>...;
const loader = new Github('hikaMaeng', 'codespitz75');

//img
const img =(v, el)=>el.src = 'data:text/plain;base64,' + v;
loader.setParser(img, el('#a'));
loader.load('xx.png');

//md
const md =(v, el)=>el.innerHTML = parseMD(v);
loader.setParser(md, el('#b'));
loader.load('xx.md');
```

```
const Github = class{
  constructor(id, repo, parser){
    this._base = `https://api.github.com/repos/...`;
    this.parser = parser;
  }
  load(path){
    const id = 'callback' + Github._id++;
    const f = Github[id] = ({data:{content}})=>{
      delete Github[id];
      document.head.removeChild(s);
      this._parser(content);
    };
    const s = document.createElement('script');
    s.src = `${this._base + path}?callback=Github. ${id}`;
    document.head.appendChild(s);
  }
  set parser(v){this._parser = v;}
};
Github._id = 0;
```

정의시점과 실행시점

```
<script src="Github.js"></script>
<script src="img.js"></script>
<script src="md.js"></script>
```

정의시점

```
<script>
const loader = new Github('hikaMaeng', 'codespitz75');

loader.setParser(img, el('#a'));
loader.load('xx.png');

loader.setParser(md, el('#b'));
loader.load('xx.md');
</script>
```

실행시점선택

실행시점 선택 위임

```
const loader = new Github('hikaMaeng', 'codespitz75');

loader.setParser(img, el('#a'));
loader.load('xx.png');

loader.setParser(md, el('#b'));
loader.load('xx.md');
```

실행시점 선택 위임

```
const Loader = class{
    constructor(id, repo){
        this._git = new Github (id, repo);
        this._router = new Map;
    }
    add(ext, f, ...arg){
        ext.split(',').forEach(v=>this._router.set(v, [f, ...arg]));
    }
    load(v){
        const ext = this._v.split('.').pop();
        if(!this._router.has(ext)) return;
        this._git.setParser(...this._router.get(ext));
        this._git.load(v);
    }
};
```

실행시점 선택 위임

```
const Loader = class{
    constructor(id, repo){
        this._git = new Github (id, repo);
        this._router = new Map;
    }
    add(ext, f, ...arg){
        ext.split(',').forEach(v=>this._router.set(v, [f, ...arg]));
    }
    load(v){
        const ext = this._v.split('.').pop();
        if(!this._router.has(ext)) return;
        this._git.setParser(...this._router.get(ext));
        this._git.load(v);
    }
};
```

라우팅테이블

확장자 경우에 따라 자동분기

실행시점 선택 위임

```
const Loader = class{
    constructor(id, repo){
        this._git = new Github (id, repo);
        this._router = new Map;
    }
    add(ext, f, ...arg){
        ext.split(',').forEach(v=>this._router.set(v, [f, ...arg]));
    }
    load(v){
        const ext = this._v.split('.').pop();
        if(!this._router.has(ext)) return;
        this._git.setParser(...this._router.get(ext));
        this._git.load(v);
    }
};
```

```
const loader = new Loader('hikaMaeng', 'codespitz75');
loader.add('jpg,png,gif', img, el('#a'));
loader.add('md', md, el('#b'));

loader.load('xx.jpg');
loader.load('xx.md');
```

실행시점 선택 위임

```
const Loader = class{
    constructor(id, repo){
        this._git = new Github (id, repo);
        this._router = new Map;
    }
    add(ext, f, ...arg){
        ext.split(',').forEach(v=>this._router.set(v, [f, ...arg]));
    }
    load(v){
        const ext = this._v.split('.').pop();
        if(!this._router.has(ext)) return;
        this._git.setParser(...this._router.get(ext));
        this._git.load(v);
    }
};
```

```
const loader = new Loader('hikaMaeng', 'codespitz75');
loader.add('jpg,png,gif', img, el('#a'));
loader.add('md', md, el('#b'));

loader.load('xx.jpg');
loader.load('xx.md');
```

발생가능한 경우의 수를 값으로 기술

실행시점 선택 위임

```
const Loader = class{
    constructor(id, repo){
        this._git = new Github (id, repo);
        this._router = new Map;
    }
    add(ext, f, ...arg){
        ext.split(',').forEach(v=>this._router.set(v, [f, ...arg]));
    }
    load(v){
        const ext = this._v.split('.').pop();
        if(!this._router.has(ext)) return;
        this._git.setParser(...this._router.get(ext));
        this._git.load(v);
    }
};
```

```
const loader = new Loader('hikaMaeng', 'codespitz75');
loader.add('jpg,png,gif', img, el('#a'));
loader.add('md', md, el('#b'));

loader.load('xx.jpg');
loader.load('xx.md');
```

발생가능한 경우의 수를 값으로 기술

LOADER의 코드가 변경되지 않음

상태에 대한 분기는 사라지지 않는다.

"그 분기가 필요해서 태어났기 때문"

상태에 대한 분기는 사라지지 않는다.

"그 분기가 필요해서 태어났기 때문"

정의 시점에 제거하는 방법은?

상태에 대한 분기는 사라지지 않는다.

"그 분기가 필요해서 태어났기 때문"

정의 시점에 제거하는 방법은?

1. 분기 수 만큼 객체를 만들고
2. 실행시점에 경우의 수를 공급

실행시점으로 분기를 옮길 때의 장단점

실행시점으로 분기를 옮길 때의 장단점

장점

1. 정의 시점에 모든 경우를 몰라도 됨
2. 정의 시점에 그 경우를 처리하는 방법도 몰라도 됨

일정한 통제 범위 내에서 확장가능한 알고리즘설계 가능

실행시점으로 분기를 옮길 때의 장단점

장점

1. 정의 시점에 모든 경우를 몰라도 됨
2. 정의 시점에 그 경우를 처리하는 방법도 몰라도 됨

일정한 통제 범위 내에서 확장가능한 알고리즘설계 가능

단점

1. 실행 시점에 모든 경우를 반드시 기술해야 함
2. 실행 시점마다 알고리즘의 안정성을 담보해야 함

매 호스트코드마다 안정성을 따로 담보해야 함

실행시점으로 분기를 옮길 때의 장단점

장점

1. 정의 시점에 모든 경우를 몰라도 됨
2. 정의 시점에 그 경우를 처리하는 방법도 몰라도 됨

일정한 통제 범위 내에서 확장가능한 알고리즘설계 가능

단점

1. 실행 시점에 모든 경우를 반드시 기술해야 함
2. 실행 시점마다 알고리즘의 안정성을 담보해야 함

매 호스트코드마다 안정성을 따로 담보해야 함

캡슐화패턴 **FACTORY, BUILDER PATTERN**

PRACTICE #1

소유기반의 코드를 구현하되 강의에 있는
전략함수 대신 전략 클래스를 적용하여 개발하라.
전략 클래스는 추상층과 구상층을 구분한다.
완성된 전략클래스를 이용하면 다음과 같이 작성할 수 있다.

```
const loader = new Github('hikaMaeng', 'codespitz75');

const img = new ImageLoader('#a'); 전략객체
loader.setParser(img);
loader.load('xx.png');

const md = new MdLoader('#b'); 전략객체
loader.setParser(md);
loader.load('xx.md');
```

PRACTICE #2

Loader의 라우팅 테이블을 2단계로 확장하여
다양한 저장소별 매핑이 가능하도록 작성하라.
완성된 Loader는 다음에 대응할 수 있다.
(연습문제1의 구조를 활용하거나 강의시의 전략함수기반 중 아무거나 선택)

```
const loader = new Loader();
loader.addRepo('s74', 'hikaMaeng', 'codespitz74');
loader.addRouter('s74', 'jpg,png,gif', img, el('#a'));
loader.addRouter('s74', 'md', md, el('#b'));

loader.addRepo('s75', 'hikaMaeng', 'codespitz75');
loader.addRouter('s75', 'jpg,png,gif', img, el('#c'));
loader.addRouter('s75', 'md', md, el('#d'));

loader.load('s74', 'xx.jpg');
loader.load('s75', 'xx.md');
```

s74 레포지토리 등록

s74용 라우터 등록

s75 레포지토리 등록

s75용 라우터 등록

레포지토리별 로드