OOP DESIGN WITH GAME

레이어 분리

Layering

먼저 크게 분리한다.

Layering

먼저 크게 분리한다.

client ⟷ server

presentation⟷domain⟷data source

Layering

먼저 크게 분리한다.

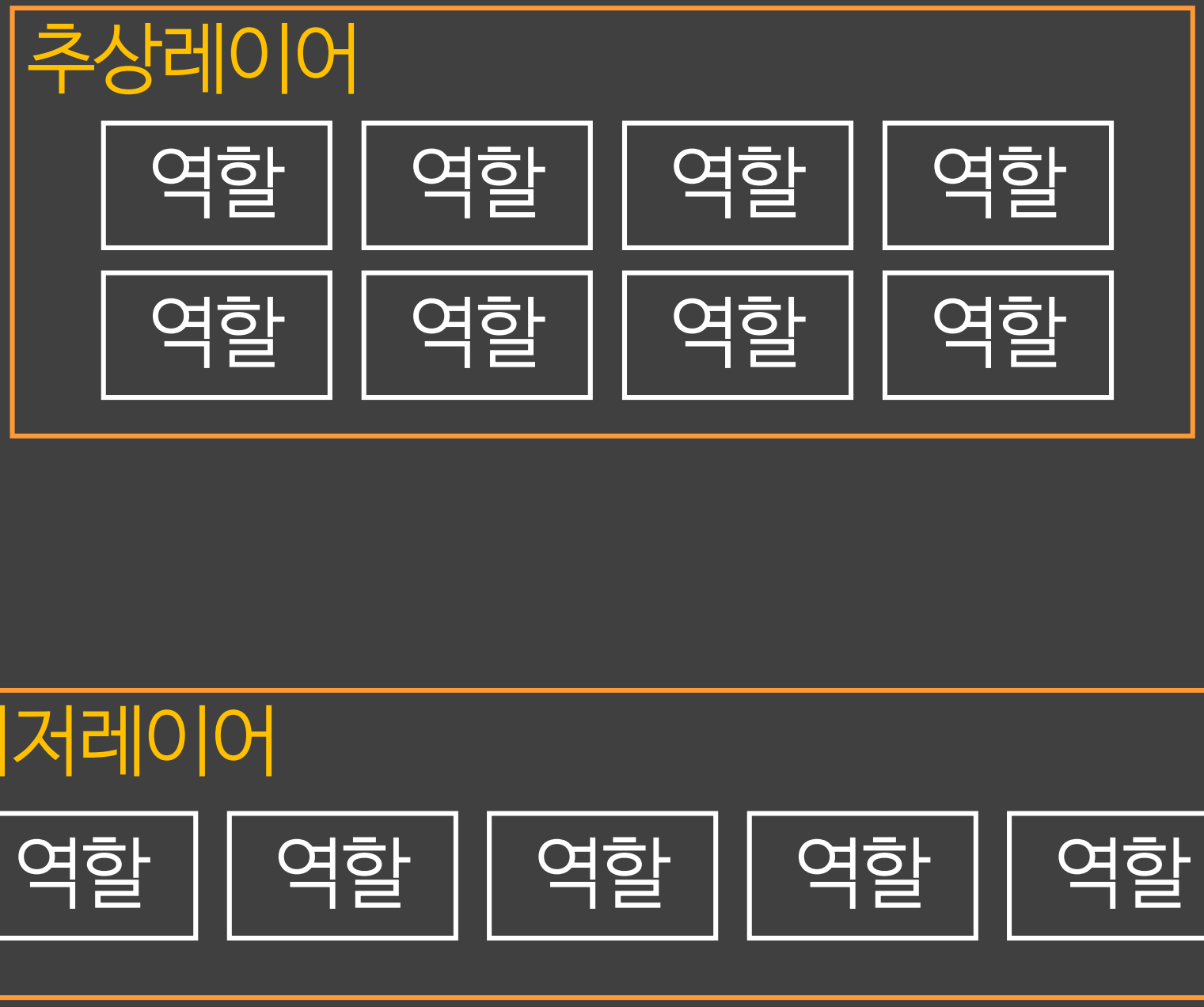
client ⟷ server

presentation↔domain↔data source

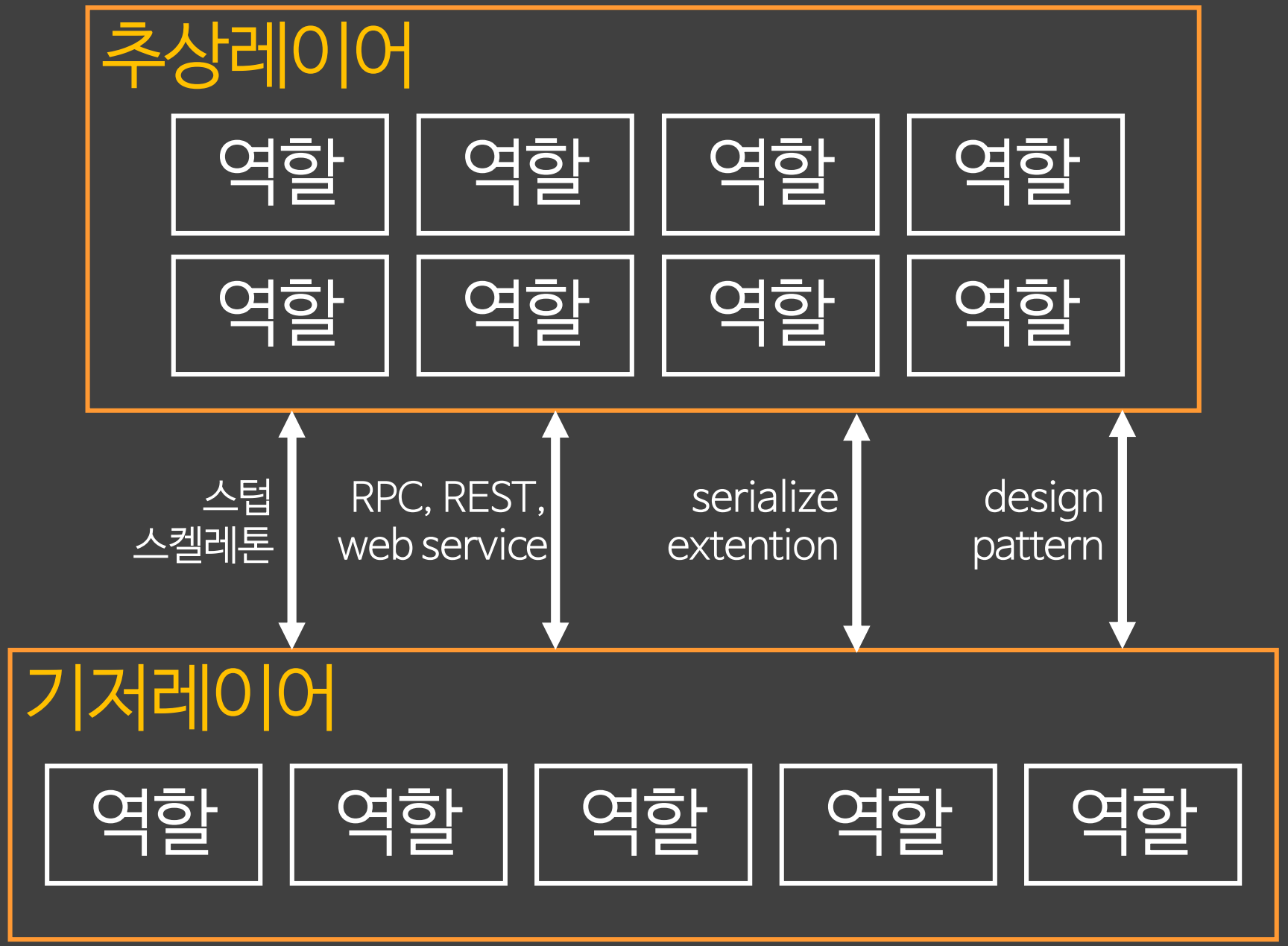
레이어는 계층적이다.
기저레이어는 추상레이어를 모른다.
레이어 안에 다수의 역할이 소속된다.

기저레이어

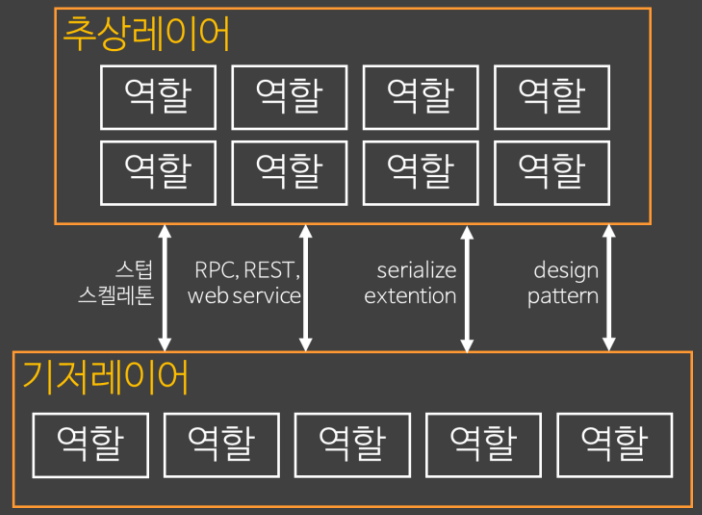

기저레이어
역할
역할
역할
역할
역할


추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
기저레이어
역할
역할
역할
역할
역할

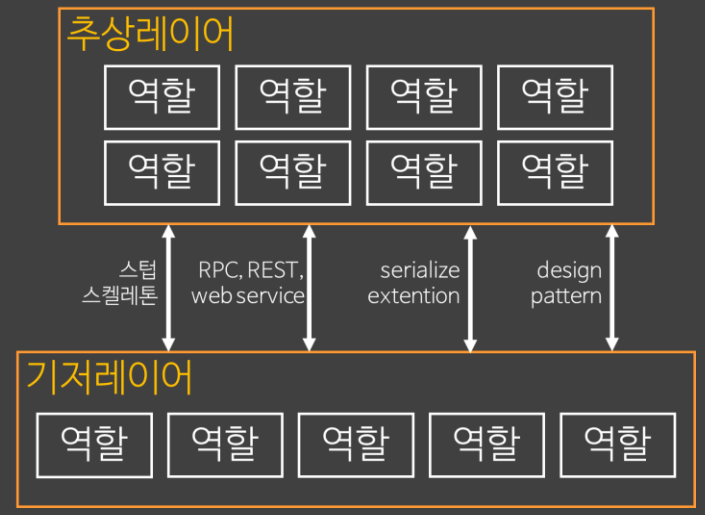

추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
기저레이어
역할
역할
역할
역할
역할

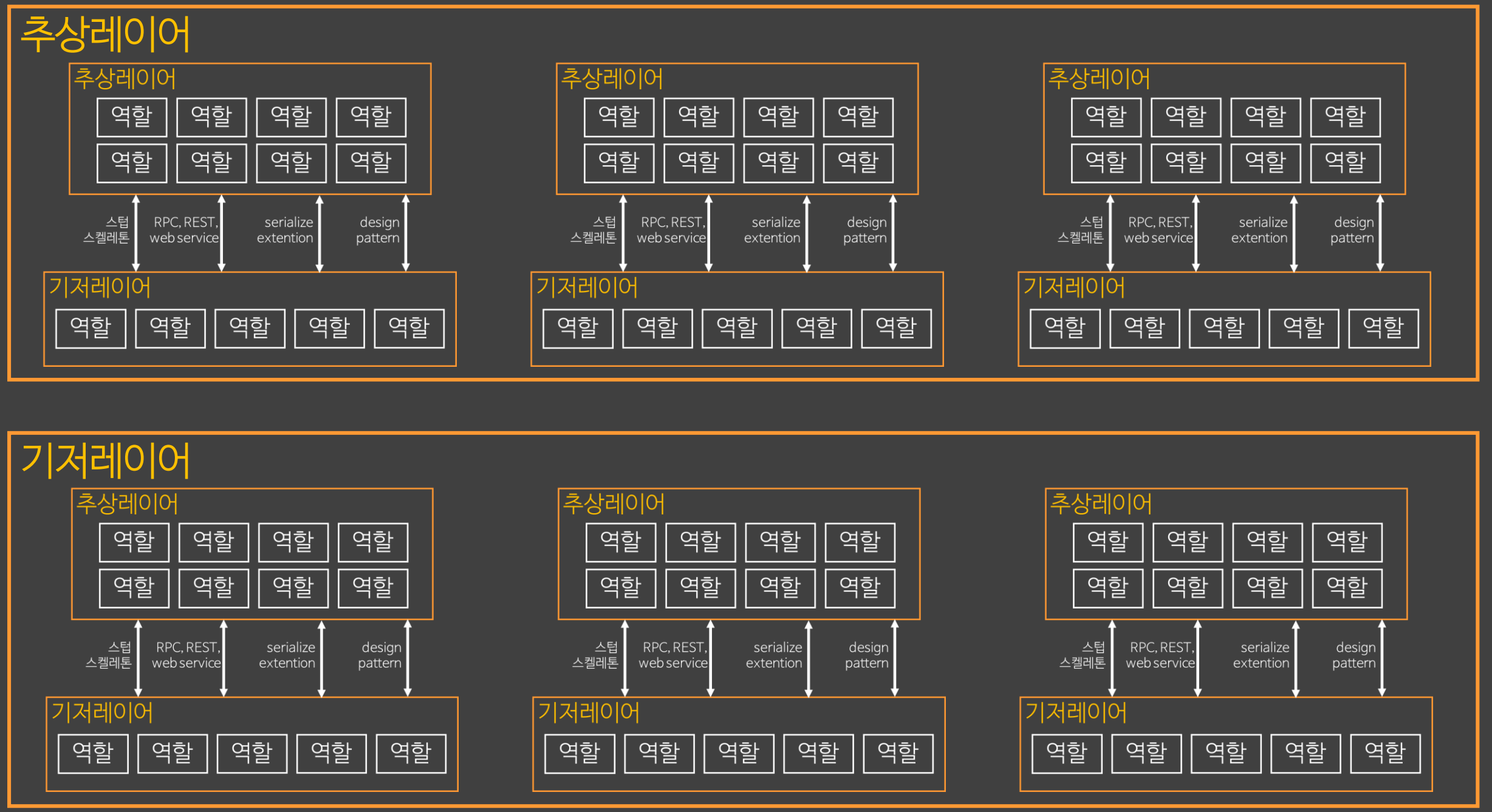

추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
serialize
extention
design
pattern
기저레이어
역할
역할
역할
역할
역할

추상레이어

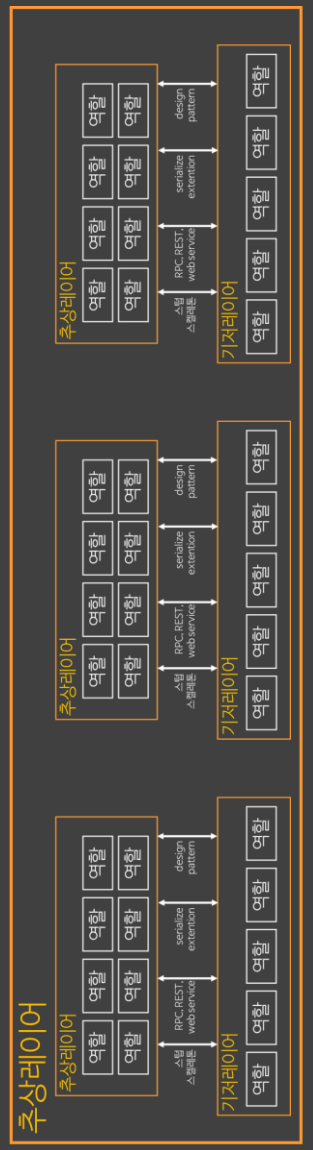

추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
serialize
extention
design
pattern
기저레이어
역할
역할
역할
역할
역할

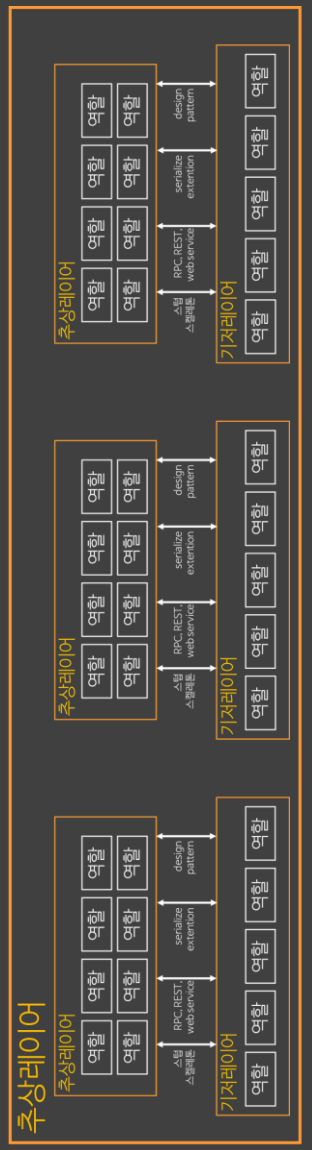
기저레이어

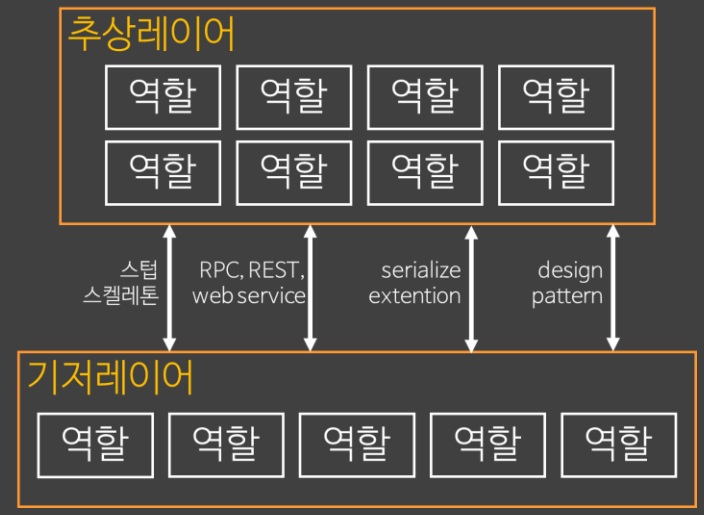

추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
serialize
extention
design
pattern
기저레이어
역할
역할
역할
역할
역할

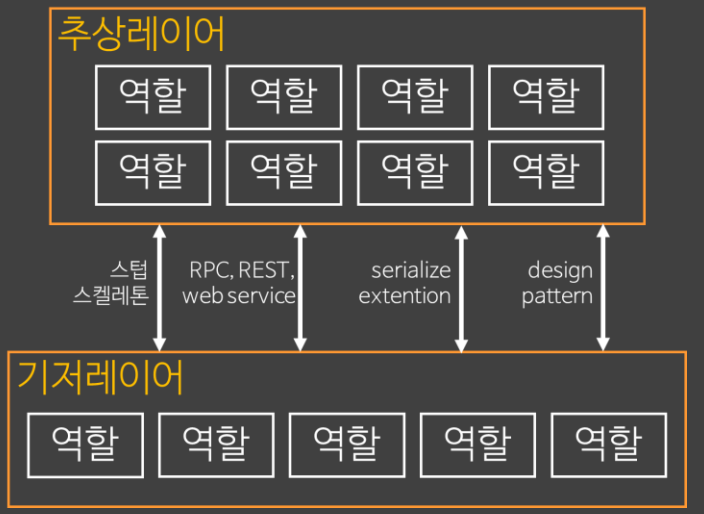

추상레이어
추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
serialize
extention
design
pattern
기저레이어
역할
역할
역할
역할
역할
추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
serialize
extention
design
pattern
기저레이어
역할
역할
역할
역할
역할
추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
serialize
extention
design
pattern
기저레이어
역할
역할
역할
역할
역할
기저레이어
추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
serialize
extention
design
pattern
기저레이어
역할
역할
역할
역할
역할
추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
serialize
extention
design
pattern
기저레이어
역할
역할
역할
역할
역할
추상레이어
역할
역할
역할
역할
역할
역할
역할
역할
스텁
스켈레톤
RPC, REST,
web service
serialize
extention
design
pattern
기저레이어
역할
역할
역할
역할
역할


추상레이어
추상레이어
역할
스텁 스켈레톤
RPC, REST, web service
serialize extention
design pattern
기저레이어


추상레이어
역할
design pattern
serialize extention
RPC, REST, web service
스텁 스켈레톤
기저레이어


추상레이어
역할
design pattern
serialize extention
RPC, REST, web service
스텁 스켈레톤
기저레이어


추상레이어
역할
스텁 스켈레톤
RPC, REST, web service
serialize extention
design pattern
기저레이어


추상레이어
역할
스텁 스켈레톤
RPC, REST, web service
serialize extention
design pattern
기저레이어

유틸리티

베이스클래스

유틸리티

구상클래스

베이스클래스

유틸리티

호스트코드

구상클래스

베이스클래스

유틸리티

호스트코드

인스턴스화

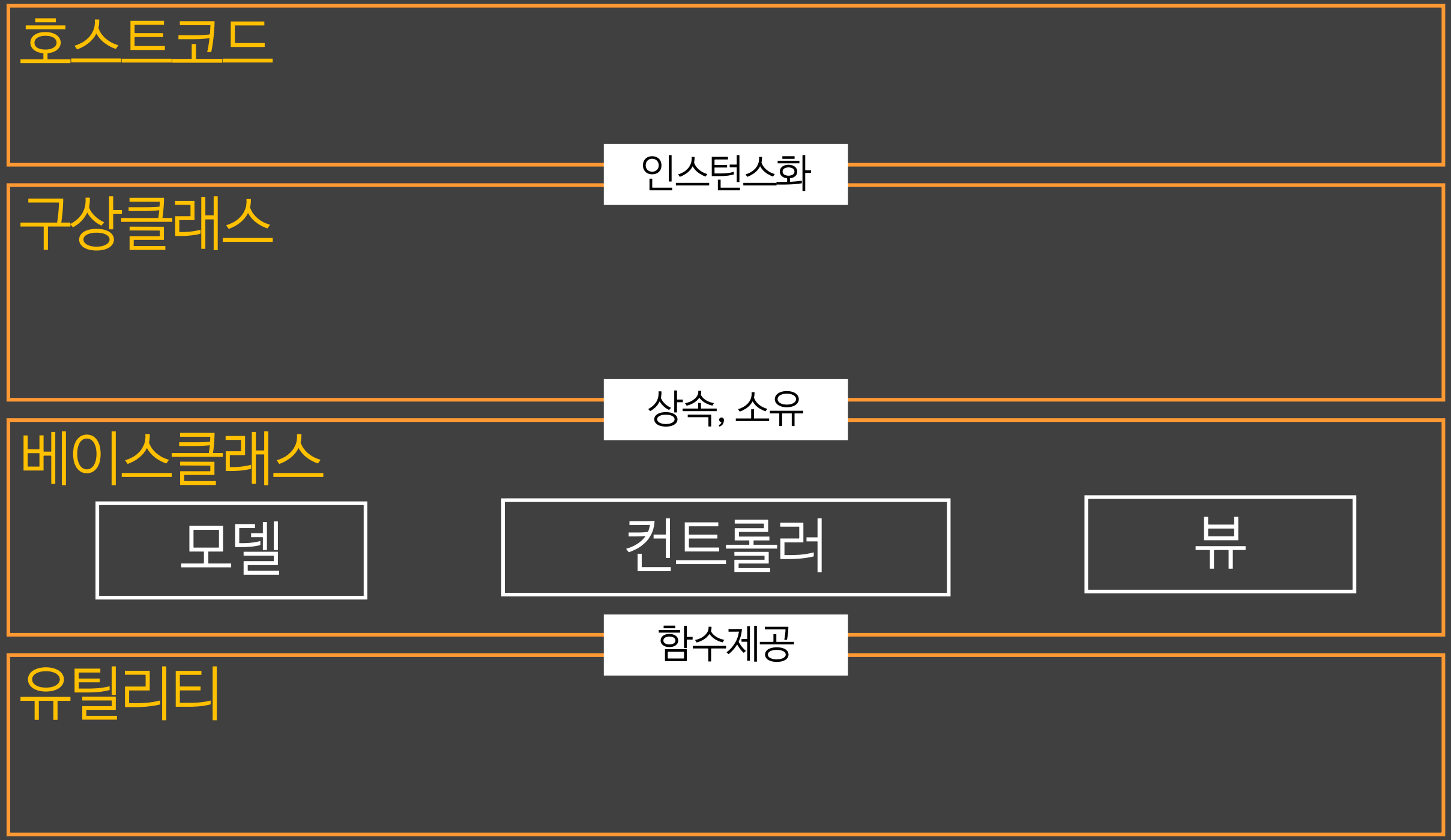
구상클래스

상속, 소유

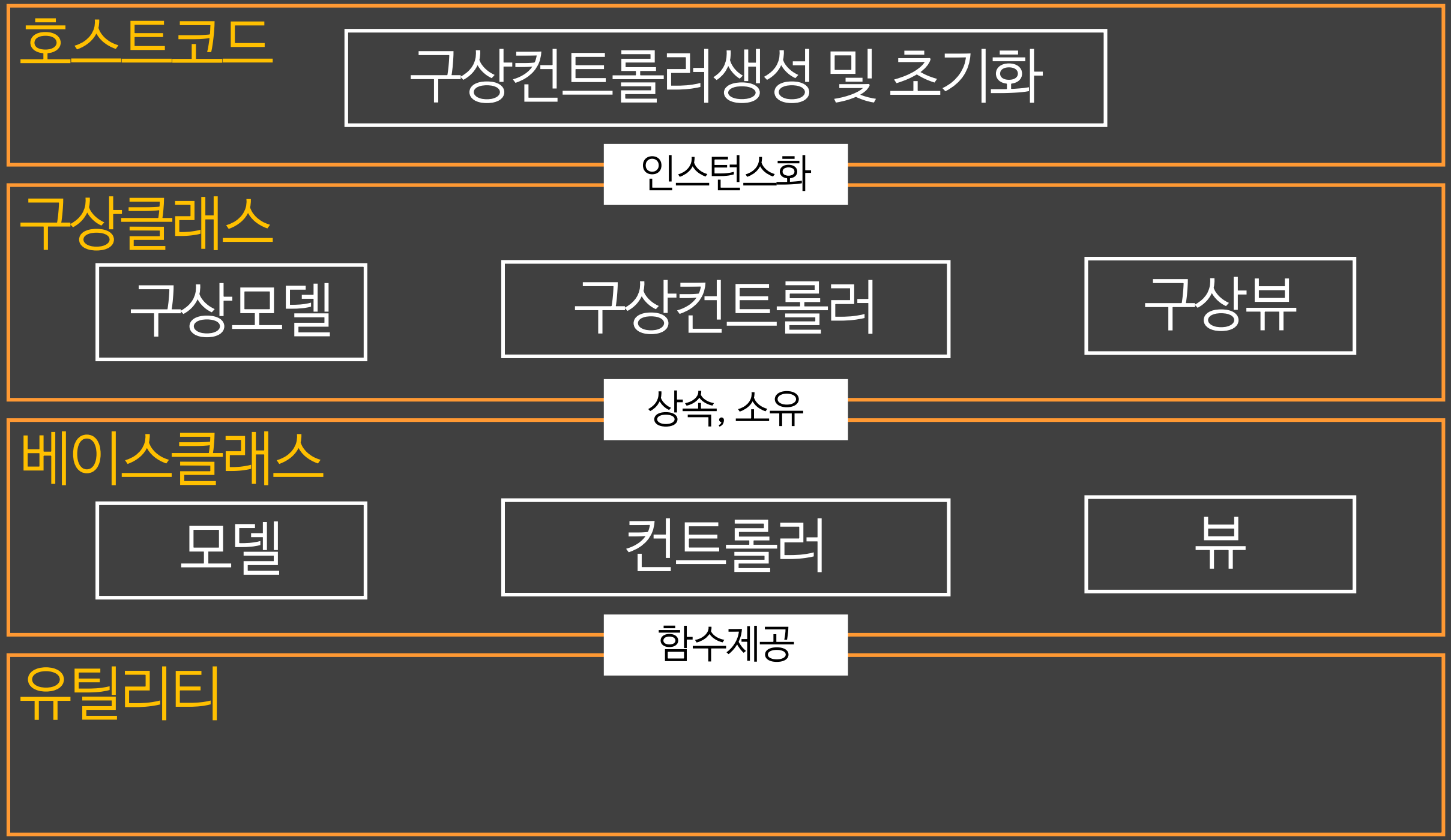
베이스클래스

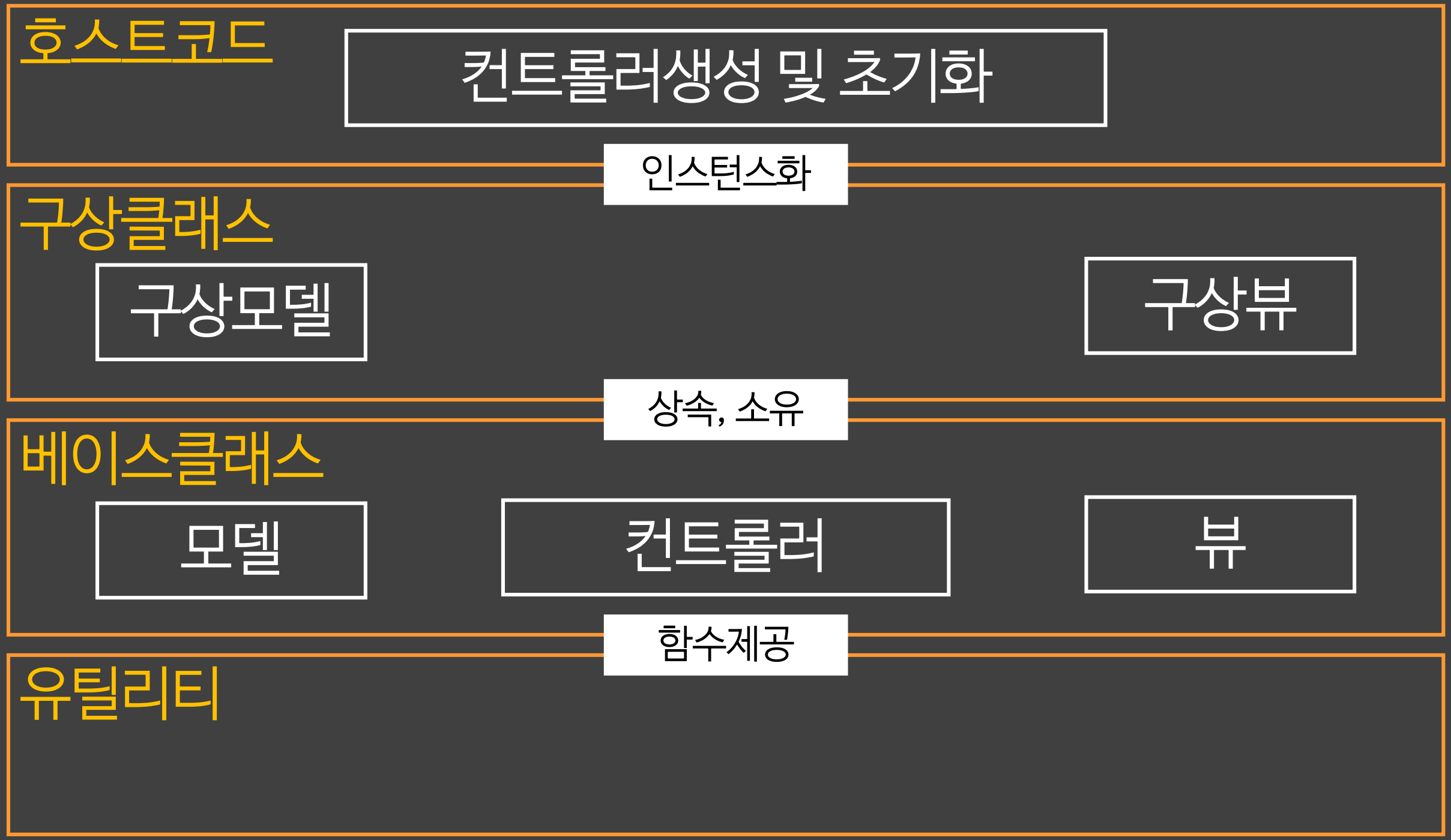
함수제공

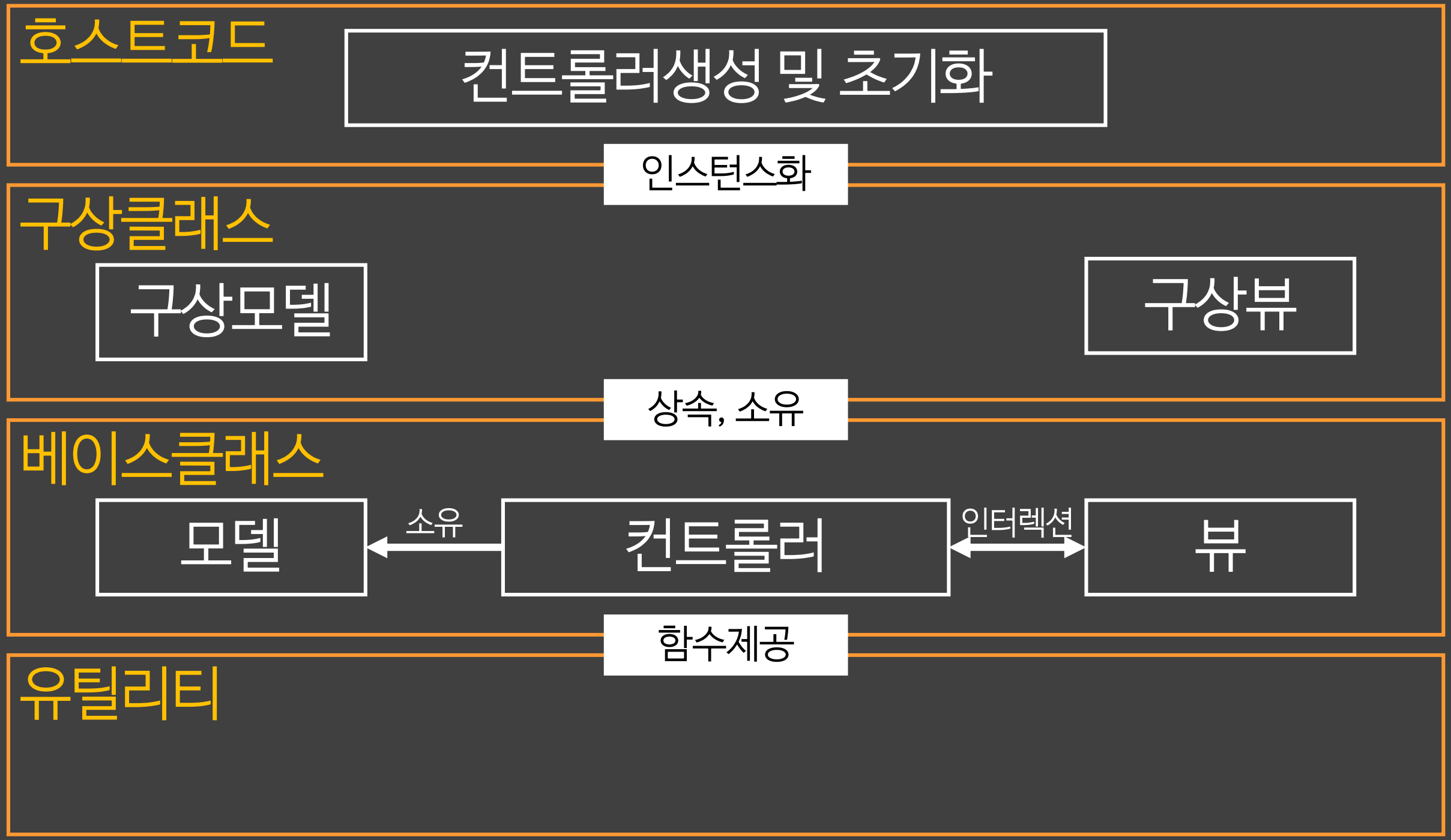
유틸리티

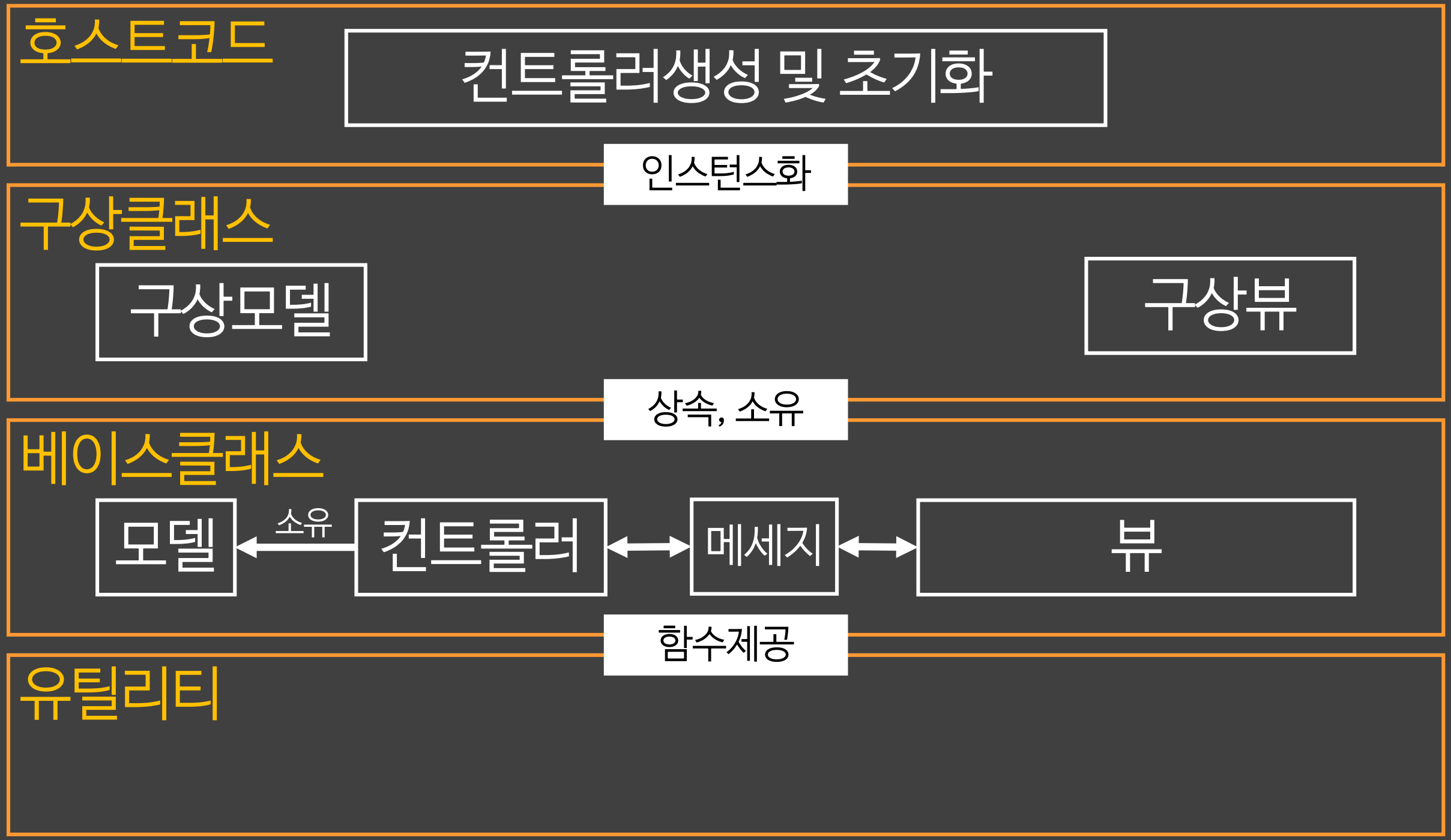

호스트코드
인스턴스화
구상클래스
상속, 소유
베이스클래스
모델
컨트롤러
뷰
함수제공
유틸리티

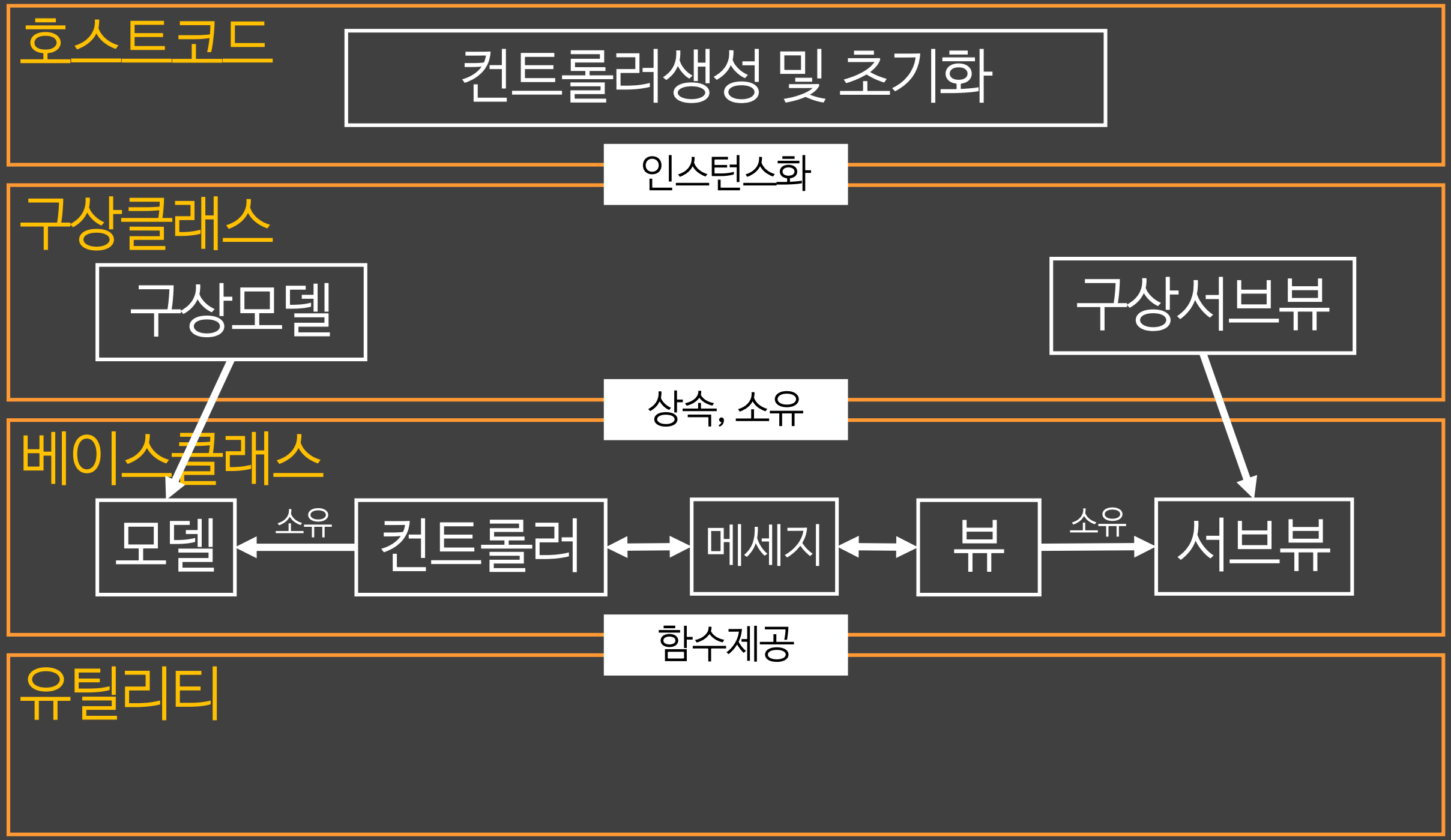

호스트코드
인스턴스화
구상클래스
구상모델
구상컨트롤러
구상뷰
상속, 소유
베이스클래스
모델
컨트롤러
뷰
함수제공
유틸리티


호스트코드
구상컨트롤러생성 및 초기화
인스턴스화
구상클래스
구상모델
구상컨트롤러
구상뷰
상속, 소유
베이스클래스
모델
컨트롤러
뷰
함수제공
유틸리티

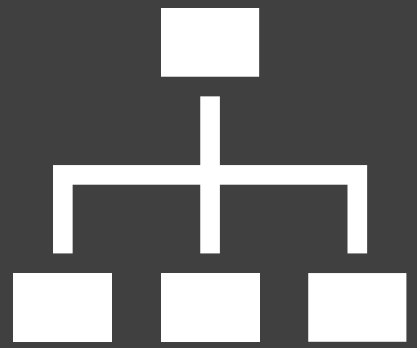

호스트코드
컨트롤러생성 및 초기화
인스턴스화
구상클래스
구상모델
구상뷰
상속, 소유
베이스클래스
모델
컨트롤러
뷰
함수제공
유틸리티


호스트코드
컨트롤러생성 및 초기화
인스턴스화
구상클래스
구상모델
구상뷰
상속, 소유
베이스클래스
모델
소유
컨트롤러
인터렉션
뷰
함수제공
유틸리티


호스트코드
컨트롤러생성 및 초기화
인스턴스화
구상클래스
구상모델
구상뷰
상속, 소유
베이스클래스
모델
소유
컨트롤러
메세지
뷰
함수제공
유틸리티


호스트코드
컨트롤러생성 및 초기화
인스턴스화
구상클래스
구상모델
구상서브뷰
상속, 소유
베이스클래스
모델
소유
컨트롤러
메세지
뷰
소유
서브뷰
함수제공
유틸리티

모델



```
const Block = class{
  static GET(type = parseInt(Math.random() * 5)){return new Block(type);}
  constructor(type){
    this._type = type;
  }
  get image(){return `url('img/block${this._type}.png')`;}
  get type(){return this._type;}
}
```

```
const data = [];
```

```
const data = [];
```

```
return tid=>{
  table = document.querySelector(tid);
  for(let i = 0; i < row; i++){
    const r = [];
    data.push(r);
    for(let j = 0; j < column; j++) r[j] = Block.GET();
  }
  table.addEventListener('mousedown', down);
  table.addEventListener('mouseup', up);
  table.addEventListener('mouseleave', up);
  table.addEventListener('mousemove', move);
  render();
};
```

```
const selected = [];
```

```
const selected = [];
```

```
const down = ({pageX:x, pageY:y})⇒{
  if(isDown) return;
  const curr = getBlock(x, y);
  if(!curr) return;
  isDown = true;
  selected.length = 0;
  selected[0] = startBlock = currBlock = curr;
  render();
};
const move =({pageX:x, pageY:y})⇒{
  if(!isDown) return;
  const curr = getBlock(x, y);
  if(!curr || curr.type ≠ startBlock.type || !isNext(curr)) return;
  if(selected.indexOf(curr) == -1) selected.push(curr);
  else if(selected[selected.length - 2] == curr) selected.pop();
  currBlock = curr;
  render();
};
const up =_⇒selected.length > 2 ? remove() : reset();
```

```
const Item = class{
   static GET(type, x, y){return new Item(type, x, y);}
   constructor(_type, _x, _y){
      prop(this, {_type, _x, _y, _selected:false, _prev:null});
   }
```

Util layer

```
const UTIL = {
  el:v=>document.querySelector(v),
  prop:(...arg)=>Object.assign(...arg)
};
```

```
const Item = class{
   static GET(type, x, y){ return new Item(type, x, y); }
   constructor(_type, _x, _y){
      prop(this, {_type, _x, _y, _selected:false, _prev:null});
   }
   get type(){return this._type;}
   get x(){return this._x;}
   get y(){return this._y;}
   get selected(){return this._selected;}
   get prev(){return this._prev;}
```

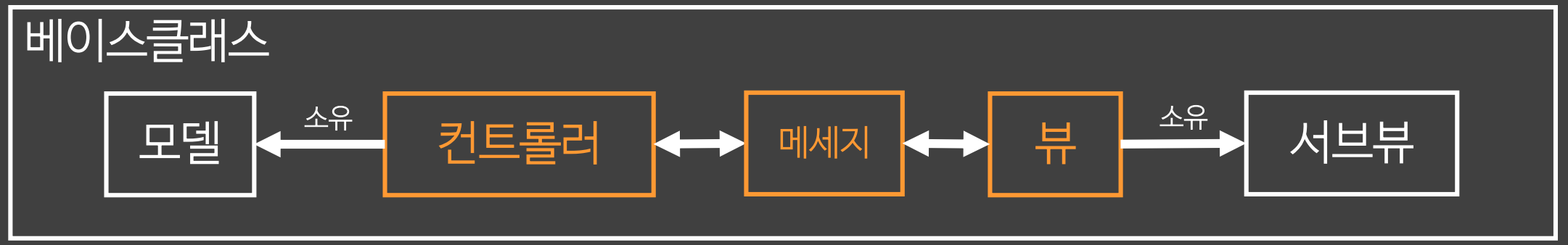
```
const Item = class{
   static GET(type, x, y){ return new Item(type, x, y); }
   constructor(_type, _x, _y){
      prop(this, {_type, _x, _y, _selected:false, _prev:null});
   }
   get type(){return this._type;}
   get x(){return this._x;}
   get y(){return this._y;}
   get selected(){return this._selected;}
   get prev(){return this._prev;}
   pos(x, y){
      this._x = x;
      this._y = y;
   }
   select(item){
      this._selected = true;
      this._prev = item;
   }
   unselect(){
      this._selected = false;
      this._prev = null;
   }
```

```
const Item = class{
   static GET(type, x, y){ return new Item(type, x, y); }
   constructor(_type, _x, _y){
      prop(this, {_type, _x, _y, _selected:false, _prev:null});
   }
   get type(){return this._type;}
   get x(){return this._x;}
   get y(){return this._y;}
   get selected(){return this._selected;}
   get prev(){return this._prev;}
   pos(x, y){
      this._x = x;
      this._y = y;
   }
   select(item){
      this._selected = true;
      this._prev = item;
   }
   unselect(){
      this._selected = false;
      this._prev = null;
   }
                              isSelectedList(item){
                                 if(!this._prev) return false;
                                 if(this._prev === item) return true;
                                 else return this._prev.isSelectedList(item);
                              }
                              isBorder(item){
                                 return Math.abs(this.x - item.x) < 2 && Math.abs(this.y - item.y) < 2;
                              }
                           }
```

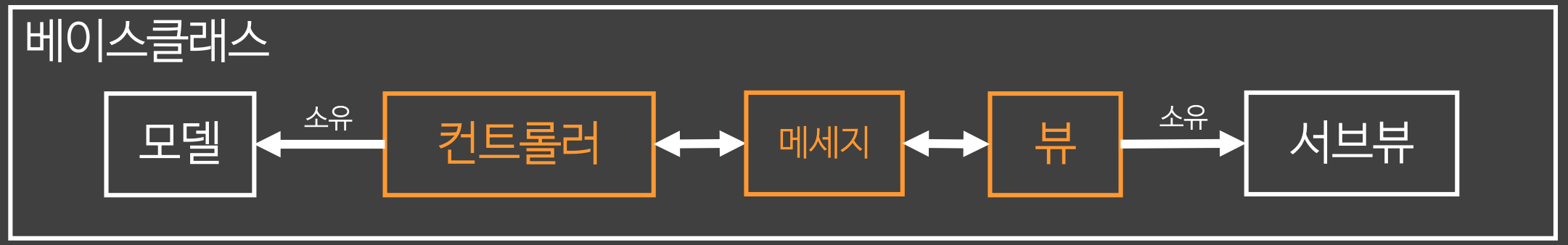
컨트롤러

```
const Game = class{
   constructor(setting){
      prop(this, setting, {
         items:new Set,
         msg2item:new WeakMap,
         item2msg:new WeakMap
      });
```

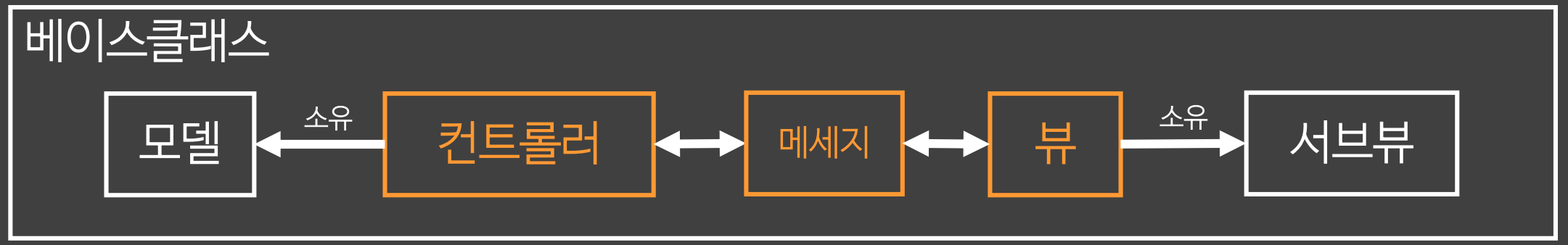
```
const Game = class{
    constructor(setting){
        prop(this, setting, {
            items:new Set,
            msg2item:new WeakMap,
            item2msg:new WeakMap
        });
```



```
const Game = class{
    constructor(setting){
        prop(this, setting, {
            items:new Set,
            msg2item:new WeakMap,
            item2msg:new WeakMap
        });
        const {renderer, row, column, items, item2msg} = this;
        renderer.setGame(this, row, column);
        for(let c = 0; c < column; c++){
            for(let r = 0; r < row; r++) this._add(c, r);
        }
```



```
const Game = class{
    constructor(setting){
        prop(this, setting, {
            items:new Set,
            msg2item:new WeakMap,
            item2msg:new WeakMap
        });
        const {renderer, row, column, items, item2msg} = this;
        renderer.setGame(this, row, column);
        for(let c = 0; c < column; c++){
            for(let r = 0; r < row; r++) this._add(c, r);
        }
        Promise.all(items.map(item=>{
            item.pos(item.x, item.y + row);
            return renderer.move(item2msg.get(item).pos(item.x, item.y)));
        })).then(_=>renderer.activate())
    }
```



```
const Game = class{
   constructor(setting){
      prop(this, setting, {
         items:new Set,
         msg2item:new WeakMap,
         item2msg:new WeakMap
      });
      ...
   }
   _add(c, r){
      const {itemType, row, items, msg2item, item2msg, renderer} = this;
      const item = new Item(itemType[parseInt(Math.random() * itemType.length)], c, r - row);
      const msg = new GameMsg;
      items.add(item);
      msg2item.set(msg, item);
      item2msg.set(item, msg);
      renderer.add(msg);
      return item;
   }
   _delete(item){
      const msg = this.item2msg.get(item);
      this.msg2item.delete(msg);
      this.item2msg.delete(item);
      this.items.delete(item);
   }
```

```
const Game = class{
   ...
   getInfo(msg){
      const item = this.msg2item.get(msg);
      msg.info(item.x, item.y, item.type, item.selected);
      return msg;
   }
```

getInfo(msg)

```
const Game = class{
   constructor(setting){
      prop(this, setting, {
         items:new Set,
         msg2item:new WeakMap,
         item2msg:new WeakMap,
         prevItem:null
      });
      ...
   }
   selectStart(msg){
      const item = this.msg2item.get(msg);
      if(!item) return;
      item.select();
      this.prevItem = item;
   }
```

```
getInfo(msg)
selectStart(msg)
```

```
const Game = class{
   ...
   selectStart(msg){
      const item = this.msg2item.get(msg);
      if(!item) return;
      item.select();
      this.prevItem = item;
   }
   selectNext(msg){
      const item = this.msg2item.get(msg);
      if(!item) return;
      const {prevItem:curr} = this;
      //자신이 아니고 타입이 같아야하며, 인접셀이어야 함
      if(item == curr || item.type != curr.type || !curr.isBorder(item)) return;
      if(!curr.isSelectedList(item)){ //선택된 게 아니면 add
         item.select(curr);
         this.prevItem = item;
      }else{ //선택된 것 중에서 직전 것이면 release
         if(curr.prev === item){
            this.prevItem = curr.prev;
            curr.unselect();
         }
      }
   }
```

```
getInfo(msg)
selectStart(msg)
selectNext(msg)
```

```
const Game = class{


   ...

   selectEnd(){
      const {items, item2msg, renderer} = this;
      const selected = [];
      items.forEach(v=>v.selected && selected.push(item2msg.get(v)));
      if(selected.length > 2) renderer.remove(selected).then(_=>this._clear());
      else items.forEach(v=>v.unselect());
      this.prevItem = null;
   }
```

```
getInfo(msg)
selectStart(msg)
selectNext(msg)
selectEnd()
```

```
const Game = class{
   ...
    _clear(selectedItems){}
   _dropBlocks(){}
   _fillStart(){}
```

```
getInfo(msg)
selectStart(msg)
selectNext(msg)
selectEnd()
_clear(selectedItems)
_dropBlocks()
_fillStart()
```

```
const Game = class{
   ...
   _clear(selectedItem){
      const {items, renderer} = this;
      renderer.deactivate();
      items.forEach(item=>item.selected && this._delete(item));
      this._dropBlocks();
   }
   _dropBlocks(){}
   _fillStart(){}
```

```
getInfo(msg)
selectStart(msg)
selectNext(msg)
selectEnd()
_clear(selectedItems)
_dropBlocks()
_fillStart()
```

```
const Game = class{
   _dropBlocks(){
      const {items, column, row, renderer, item2msg} = this;
      const allItems = [];
      for(let i = row; i--;) allItems.push([]);
      items.forEach(item=>(allItems[item.y][item.x] = item));
      const coll = [];
      for(let c = 0; c < column; c++){
         for(let r = row - 1; r > -1; r--){
            if(allItems[r] && allItems[r][c]){
               let cnt = 0;
               for(let j = r + 1; j < row; j++){
                  if(allItems[j] && ! allItems[j][c]) cnt++;
               }
               if(cnt){
                  const item = allItems[r][c];
                  item.pos(c, r + cnt);
                  coll.push(renderer.move(item2msg.get(item).pos(item.x, item.y)));
               }
            }
         }
      }
      if(coll.length) Promise.all(coll).then(_=>this._fillStart());
   }
```

```
getInfo(msg)
selectStart(msg)
selectNext(msg)
selectEnd()
_clear(selectedItems)
_dropBlocks()
_fillStart()
```

```
const Game = class{
   _fillStart(){
      const {items, column, row, renderer, item2msg} = this;
      const allItems = [];
      for(let i = row; i--;) allItems.push([]);
      items.forEach(item=>(allItems[item.y][item.x] = item));
      const coll = [];
      for(let c = 0; c < column; c++){
         for(let r = row - 1; r > -1; r--){
            if(allItems[r] && ! allItems[r][c]) coll.push(this._add(c, r));
         }
      }
      if(!coll.length) return;
      Promise.all(coll.map (item=>{
         item.pos(item.x, item.y + row);
         return renderer.move(item2msg.get(item).pos(item.x, item.y));
      }).then(_=>renderer.activate())
   }
```

```
getInfo(msg)
selectStart(msg)
selectNext(msg)
selectEnd()
_clear(selectedItems)
_dropBlocks()
_fillStart()
```

서브렌더러

```
const ItemRenderer = class{
   get object(){throw 'override';}
   find(v){throw 'override';}
   remove(){return this._remove();}
   move(x, y){return this._move(x, y);}
   render(x, y, type, selected){ this._render(x, y, type, selected);}
   _remove(){throw 'override';}
   _move(x, y){throw 'override';}
   _render(x, y, type, selected){throw 'override';}
};
```

렌더러

Util layer

```
const UTIL = {
  el:v=>document.querySelector(v),
  prop:(...arg)=>Object.assign(...arg),
  ThrowSet:class extends Set{
   constructor(){
      super();
   }
   some(f){
      try{
         this.forEach((v, i)=>{
            if(v = f(v,i)) throw v;
         });
      }catch(r){
         return r;
      }
   }
  }
};
```

```
const Renderer = class extends ThrowSet{
   constructor(itemFactory){
      super();
      prop(this, {_itemFactory:itemFactory, msg2item:new WeakMap, item2msg:new WeakMap});
   }
   setGame(_game, _row, _col){ prop(this, {_game, _row, _col}); }
   activate(){throw 'override!';}
   deactivate(){throw 'override!';}
```

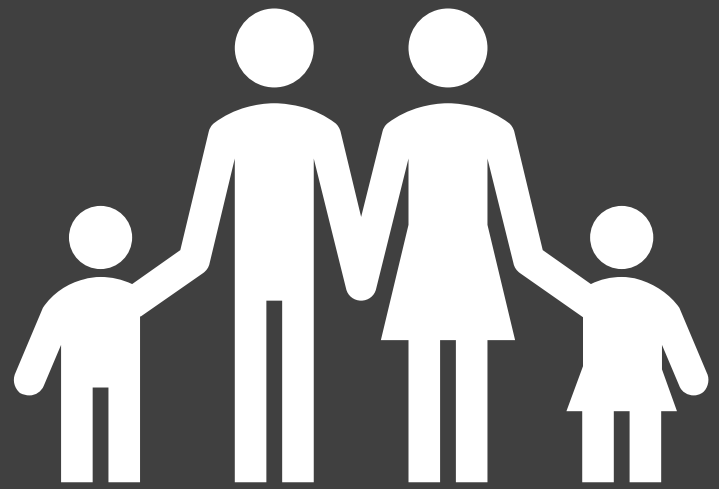
```
const Renderer = class extends ThrowSet{
   add(msg){
      const {msg2item, item2msg, _itemFactory} = this;
      const item = _itemFactory(this, this.bw, this.bh, this.img);
      super.add(item);
      msg2item.set(msg, item);
      item2msg.set(item, msg);
      this._add(item);
   }
   _add(v){throw 'override'}
```

```
const Renderer = class extends ThrowSet{
   remove(msgs){
      if(!msgs.length) return;
      const {msg2item} = this;
      return Promise.all(msgs.map(msg=>{
         const item = msg2item.get(msg);
         msg2item.delete(msg);
         this._delete(item);
         return item.remove();
      }));
   }
   _delete(item){
      this.item2msg.delete(item);
      super.delete(item);
      this._remove(item);
   }
   _remove(item){throw 'override!';}
   move(msg){
      const {x, y} = msg.pos();
      return this.msg2item.get(msg).move(x, y);
   }
```

```
const Renderer = class extends ThrowSet{
   itemStart(item){ this._gameRequest(this._game.selectStart, item); }
   itemNext(item){ this._gameRequest(this._game.selectNext, item); }
   itemEnd(){ this._gameRequest(this._game.selectEnd); }
   _gameRequest(f, item){
      const {_game:game, item2msg} = this;
      if(item) f.call(game, item2msg.get(item));
      else f.call(game);
   }
```

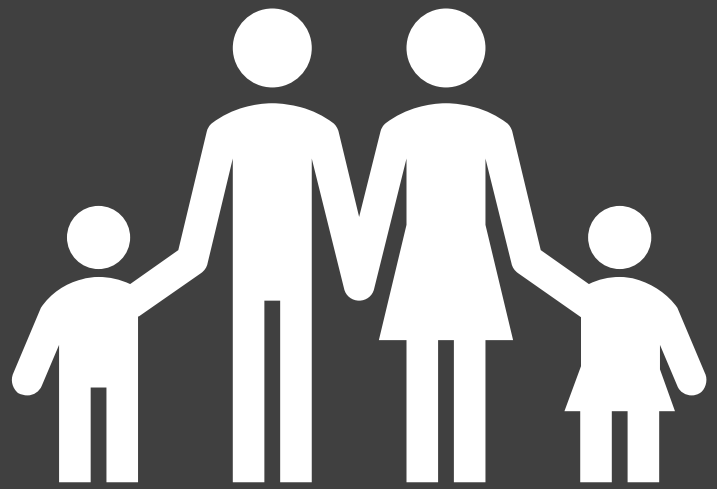
```
const Renderer = class extends ThrowSet{
   _renderLoop(){
      const {_game:game, item2msg} = this;
      this.forEach(item=>{
         const {x, y, type, selected} = game.getInfo(item2msg.get(v)).info();
         item.render(x, y, type, selected);
      });
      this._render();
   }
   _render(){throw 'override'}
```

구상레이어



구상레이어

구상 렌더러



```
const DivRenderer = class extends ItemRenderer{
   constructor(_parent, bw, bh, img){
      prop(this, {_parent, img, bw, bh, div});
      const div =  el('div');
      div.className = 'block';
      div.style.cssText = `width:${bw}px;height:${bh}px;backgroundImage:url(${img})`;
   }
   get object(){return this.div;}
   find(el){return el == this.div;}
```

```
const ItemRenderer = class{
   get object(){throw 'override';}
   find(v){throw 'override';}
   remove(){return this._remove();}
   move(x, y){return this._move(x, y);}
   render(x, y, type, selected){ this._render(x, y, type, selected);}
   _remove(){throw 'override';}
   _move(x, y){throw 'override';}
   _render(x, y, type, selected){throw 'override';}
};
```

```
const DivRenderer = class extends ItemRenderer{
   _remove(){
      const {div, _parent:parent} = this;
      return new Promise((resolve, reject)=>{
         div.style.transition = "transform ease-in 350ms";
         div.style.transform = "scale(0,0)";
         parent.delayTask(resolve, 350);
      });
   }
```

```
const ItemRenderer = class{
   get object(){throw 'override';}
   find(v){throw 'override';}
   remove(){return this._remove();}
   move(x, y){return this._move(x, y);}
   render(x, y, type, selected){ this._render(x, y, type, selected);}
   _remove(){throw 'override';}
   _move(x, y){throw 'override';}
   _render(x, y, type, selected){throw 'override';}
};
```

```
const DivRenderer = class extends ItemRenderer{
   _move(x, y){
      const {div, bw, bh, _parent:parent} = this;
      return new Promise((resolve, reject)=>{
         const time = (y * bh - parseInt(div.style.top)) / bh * 100;
         div.style.transition = `top ease-in ${time}ms`;
         parent.delayTask(resolve, time);
      });
   }
   _render(x, y, type, selected){
      const {div, bw, bh, img} = this;
      div.style.left = bw * x + "px";
      div.style.top = bh * y + "px";
      div.style.backgroundPosition = -(bw * type) + "px";
      div.style.backgroundPositionY = (selected ? -bh : 0) + "px";
   }
```

```
const ItemRenderer = class{
   get object(){throw 'override';}
   find(v){throw 'override';}
   remove(){return this._remove();}
   move(x, y){return this._move(x, y);}
   render(x, y, type, selected){ this._render(x, y, type, selected);}
   _remove(){throw 'override';}
   _move(x, y){throw 'override';}
   _render(x, y, type, selected){throw 'override';}
```

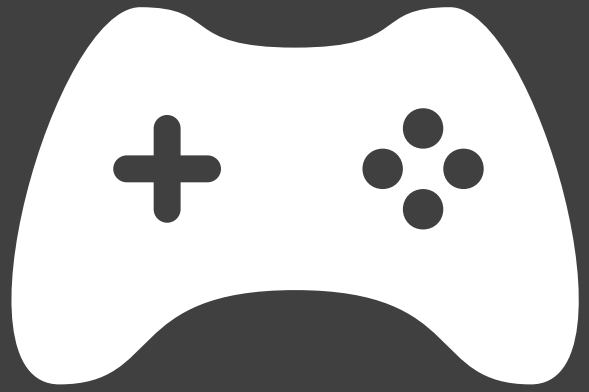
```
const SectionRenderer = class extends Renderer{
  constructor({stage, bg, w, h, c, r, img, itemFactory}){
    super(itemFactory);
    stage = el(stage);
    const bw = parseInt(w/c), bh = parseInt(h/r), _q = [];
    prop(this, {stage, bw, bh, w, h, c, r, img, isdown:false, _q, isAct:null, curr:0});
    stage.style.cssText = `width:${w}px;height:${h}px;background-image:url('${bg}');
       background-size:${bw}px ${bh}px`;
    stage.setAttribute('unselectable', 'on');
    stage.setAttribute('onselectstart', 'return false');
    const f = t=>{
      this.curr = t;
      for(let i = _q.length; i--;){
        const task = _q[i];
        if(task.t <= t){
          _q.splice(i, 1);
          task.f();
        }
      }
      this._renderLoop();
      requestAnimationFrame(f);
    };
    requestAnimationFrame(f);
  }
```

```
const SectionRenderer = class extends Renderer{
  delayTask(f, time){
    this._q.push({f, t:this.curr + time});
  }
  activate(){
    const {stage} = this;
    if(this.isAct === null){
      stage.addEventListener('mousedown', e=>this.isAct && this.dragDown(e));
      stage.addEventListener('mouseup', e=>this.isAct && this.dragUp(e));
      stage.addEventListener('mouseleave', e=>this.isAct && this.dragUp(e));
      stage.addEventListener('mousemove', e=>this.isAct && this.dragMove(e));
    }
    this.isAct = true;
  }
  deactivate(){
    this.isAct = false;
  }

  _add(item){ this.stage.appendChild(item.object);}
  _remove(item){this.stage.removeChild(item.object);}
  _render(){}
```

```
const SectionRenderer = class extends Renderer{
  _getItem(x, y){
    const el = document.elementFromPoint(x, y);
    return this.some(v=>v.find(el));
  }
  dragDown({pageX:x, pageY:y}){
    const item = this._getItem(x, y);
    if(!item) return;
    this.isdown = true;
    this.itemStart(item);
  }
  dragMove({pageX:x, pageY:y}){
    const {isdown} = this;
    if(!isdown) return;
    const item = this._getItem(x, y);
    if(item) this.itemNext(item);
  }
  dragUp({pageX:x, pageY:y}){
    const {isdown} = this;
    if(!isdown) return;
    this.isdown = false;
    this.itemEnd();
  }
```

호스트코드



```
<style>
  #stage{position:relative;display:inline-block;overflow:hidden;border:1px solid #ddd}
  .block{position:absolute;cursor:pointer;overflow:hidden}
</style>
```

```
<style>
  #stage{position:relative;display:inline-block;overflow:hidden;border:1px solid #ddd}
  .block{position:absolute;cursor:pointer;overflow:hidden}
</style>
<section style="text-align:center">
  <h1 style="color:#208AFB">BSIDE TOWER</h1>
  <section id="stage"></section>
</section>
```

```
<style>
  #stage{position:relative;display:inline-block;overflow:hidden;border:1px solid #ddd}
  .block{position:absolute;cursor:pointer;overflow:hidden}
</style>
<section style="text-align:center">
  <h1 style="color:#208AFB">BSIDE TOWER</h1>
  <section id="stage"></section>
</section>

<script src="util.js"></script>
<script src="base.js"></script>
<script src="concreate.js"></script>
```

```
<style>
  #stage{position:relative;display:inline-block;overflow:hidden;border:1px solid #ddd}
  .block{position:absolute;cursor:pointer;overflow:hidden}
</style>
<section style="text-align:center">
  <h1 style="color:#208AFB">BSIDE TOWER</h1>
  <section id="stage"></section>
</section>

<script src="util.js"></script>
<script src="base.js"></script>
<script src="concreate.js"></script>

<script>
const game = new Game({
  column:6, row:6, itemType:'01234'.split(''),
  renderer:new SectionRenderer({
    stage:'#stage', bg:'../img/bg01.gif',
    img:'../img/tower.png', w:400, h:160, r:2, c:5,
    itemFactory:(parent, bw, bh, img)=>new DivRenderer(parent, bw, bh, img)
  })
});
</script>
```

PRACTICE #1

Renderer 클래스에는 레이어를 위반하는 코드가 포함되어있다.
위반하는 부분을 찾아 고치시오.