Computer Budget

객체지향설계 8조

목차

목적

- 조립식 컴퓨터 -> 가격이 쌈 , 부품에 대한 지식 필요
- 브랜드 컴퓨터 -> 가격이 비쌈 , 지식 불필요
- 부품에 대해 몰라도 싸게 사고싶다!

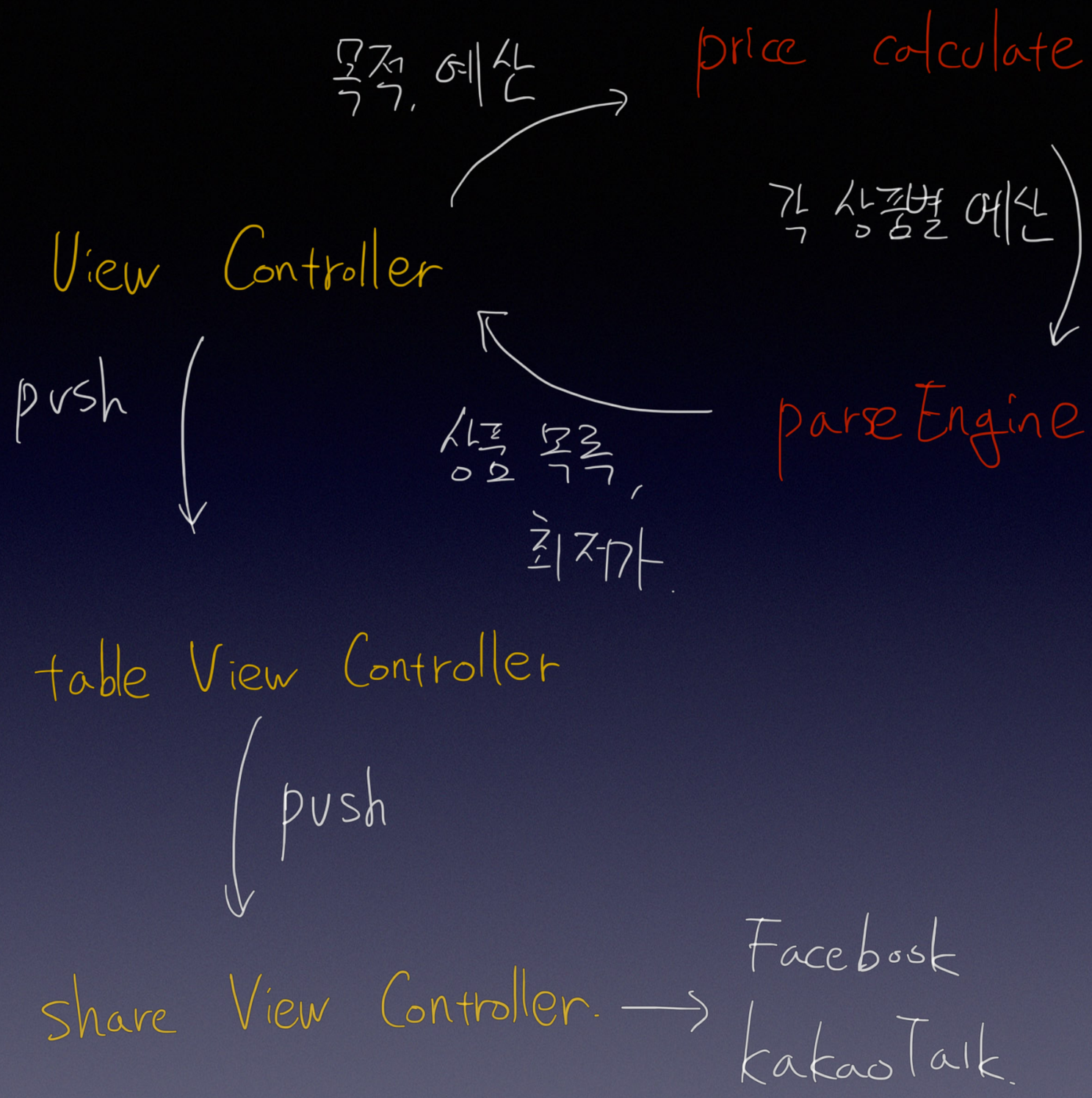
기존 다나와 견적의 단점

PC주요부품	CPU	메인보드	메모리	그래픽카드	SSD	하드디스크	ODD	케이스
	파워	키보드	마우스	사운드/스피커	모니터	쿨러/튜닝	소프트웨어	
주변기기	PC 헤드셋	리더기	복합기	컨트롤러	케이블	공유기/무선랜	IP공유기/허브	영상/TV/PC캠
	프린터	멀티탭						
추가상품	외장HDD	NAS	USB	노트북	노트북 주변기기	디지털TV	베어본	서버
	잉크	토너	공미디어	소모품	스마트패드	플래시/메모리		

CPU

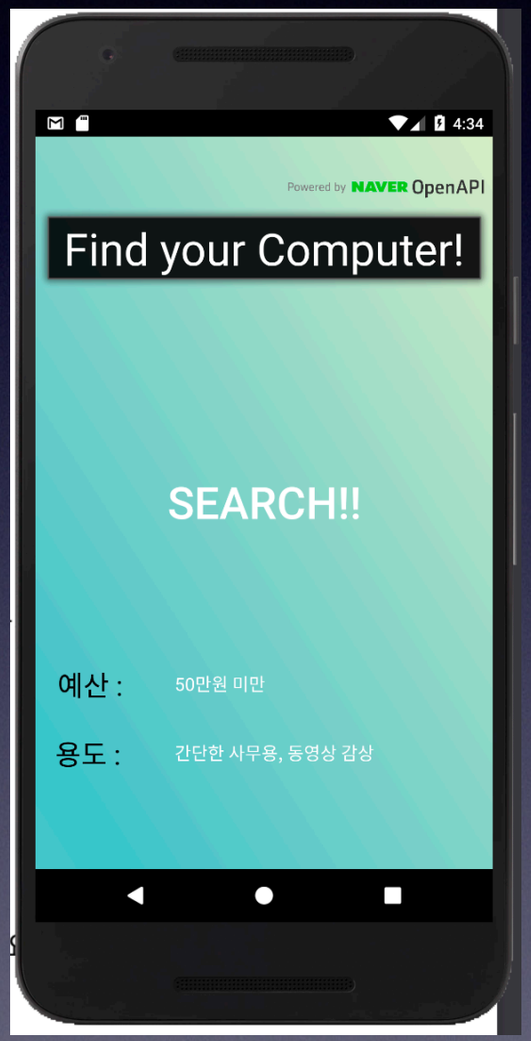
제조사	브랜드 분류	소켓 구분	제조 공정	연산 체계
코어 형태	쓰레드 형태	동작 속도	패키지 형태	GPU 유무

너무나도 복잡한 옵션.



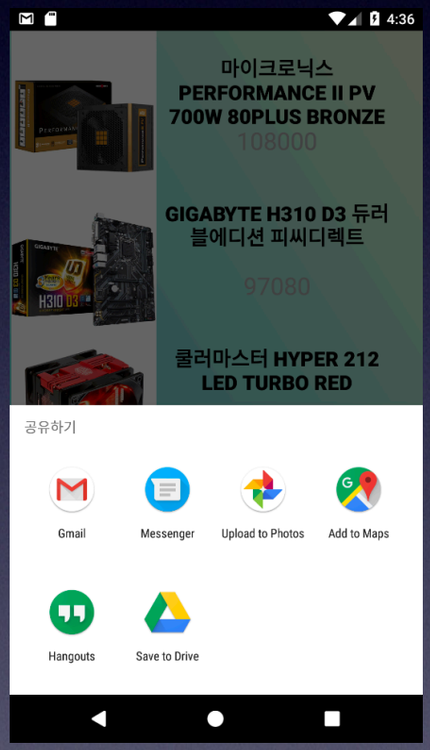
Class Diagram

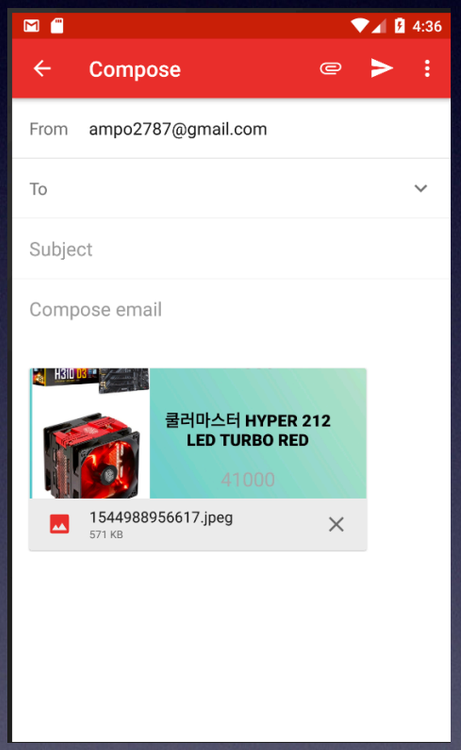
UI


4:34
Powered by NAVER OpenAPI
Find your Computer!
SEARCH!!
예산 :
50만원 미만
용도 :
간단한 사무용, 동영상 감상


4:34
Powered by NAVER OpenAPI
Find your Computer!
50만원 미만
50만원 이상 100만원 미만
100만원 이상 150만원 미만
150만원 이상
예산 :
50만원 미만
용도 :
간단한 사무용, 동영상 감상


4:35
I7 8700
637990
GTX - 1070 8GB
668000
SAMSUNG DDR4 16GB - 21300
143540
삼성전자 970 PRO M.2 2280 (512GB)
242000
마이크로닉스 PERFORMANCE II PV 700W 80PLUS BRONZE


4:36
마이크로닉스 PERFORMANCE II PV 700W 80PLUS BRONZE
108000
GIGABYTE H310 D3 듀러블에디션 피씨디렉트
97080
쿨러마스터 HYPER 212 LED TURBO RED
공유하기
Gmail
Messenger
Upload to Photos
Add to Maps
Hangouts
Save to Drive


4:36
Compose
From ampo2787@gmail.com
To
Subject
Compose email
쿨러마스터 HYPER 212 LED TURBO RED
41000
1544988956617.jpeg
571 KB

예산 산출 및 검색

1. 호출

```
priceCalculate calculate = new priceCalculate();
priceList = calculate.moneyAnalysis(budgetText);
calculate.targetAnalysis(targetText);

NaverAPI apiUser = new NaverAPI();
apiUser.parseAll(priceList);
```

2. 예산 분배

```
public HashMap<String, Integer> moneyAnalysis(int money){
    if(money <= entry){
        priceList.put(cpu, 15);
        priceList.put(gpu, 0);
        priceList.put(ram, 10);
        priceList.put(disk, 10);
        priceList.put(power, 5);
        priceList.put(mainboard, 5);
        priceList.put(cooler, 0);
        priceList.put(ccase, 5);
    }
```

예산 산출 및 검색

3. 목적에 따른 재분배

```
public void targetAnalysis(String target){
    if(target.equals("간단한 사무용, 동영상 감상")){
        priceList.remove(gpu);
        priceList.remove(cooler);
        priceList.put(gpu, 0);
        priceList.put(cooler, 0);
    }
```

4. NAVER 검색

```
public void parseAll(HashMap<String, Integer> priceList){
    parseCPU(priceList.get("cpu"));
    if(!priceList.get("gpu").equals(0)){
        parseGPU(priceList.get("gpu"));
    }else{
       finalPriceList.put("gpu" , 0);
    }
    parseRAM(priceList.get("ram"));
    parseDisk(priceList.get("disk"));

    if(!priceList.get("cooler").equals(0)) {
        parseCooler(priceList.get("cooler"));
    }else{
        finalPriceList.put("cooler" , 0);
    }
    parseMainBoard(productList.get("cpu"), priceList.get("main"));
    parsePower(priceList.get("power"));
    parseCase(priceList.get("case"));
}
```

검색 과정

1. 검색명 설정

```java
public void parseCPU(int price) {
    if (price <= 5) {
        setup("인텔 펜티엄 골드 G5600");
        productList.put("cpu", "인텔 펜티엄 골드 G5600");
    }
```

2. 연결 설정

```java
public void setup(String model){
    try {
        String apiURL = "https://openapi.naver.com/v1/search/shop.json?query=";
        URL url = new URL( spec: apiURL + model);
        con = (HttpURLConnection) url.openConnection();
        con.setRequestMethod("GET");
        con.setRequestProperty("X-Naver-Client-Id", clientId);
        con.setRequestProperty("X-Naver-Client-Secret", clientSecret);
    } catch (Exception e){
        System.out.println(e);
    }
}
```

3. 응답 저장

```java
JSONArray jsonArray = new JSONObject(response.toString()).getJSONArray( name: "items");
JSONObject temp = jsonArray.getJSONObject( index: 0);
finalPriceList.put(productKind , Integer.parseInt(temp.optString( name: "lprice")));
imageList.put(productKind ,temp.optString( name: "image"));
```

검색에 필요한 것 try - catch 와 Thread!

```
public class shoppingTask extends AsyncTask{
    @Override protected void onPreExecute() { super.onPreExecute(); }

    @Override
    protected Object doInBackground(Object[] objects) {
        try {
            animationView.loop(true);

            priceCalculate calculate = new priceCalculate();
            priceList = calculate.moneyAnalysis(budgetText);
            calculate.targetAnalysis(targetText);

            NaverAPI apiUser = new NaverAPI();
            apiUser.parseAll(priceList);

            titleList = apiUser.productList;
            imageURLList = apiUser.imageList;
            priceList = apiUser.finalPriceList;
        }
        catch (Exception e){
            System.out.println("   ??????????????          :        " + e);
        }
        return null;
    }

    @Override
    protected void onPostExecute(Object o) {
        super.onPostExecute(o);
        animationView.cancelAnimation();

        Intent intent = new Intent( packageContext: MainActivity.this, Scroll_Activity.class);
        intent.putExtra( name: "title", titleList);
        intent.putExtra( name: "price", priceList);
        intent.putExtra( name: "imageURL", imageURLList);
        startActivity(intent);
    }
}
```

이미지 쓰레드

```
Thread imageThread = run() → {
        try {
            String apiURL = imageURLList.get("cpu");
            URL url = new URL(apiURL);
            HttpURLConnection con = (HttpURLConnection) url.openConnection();
            con.setDoInput(true);
            con.connect();

            InputStream inputStream = con.getInputStream();
            CPUbitmap = BitmapFactory.decodeStream(inputStream);

            if(imageURLList.get("gpu") != null) {
                apiURL = imageURLList.get("gpu");
                url = new URL(apiURL);
                con = (HttpURLConnection) url.openConnection();
                con.setDoInput(true);
                con.connect();
            }
            inputStream = con.getInputStream();
            GPUbitmap = BitmapFactory.decodeStream(inputStream);
```

각 부품마다 한 이미지!

공유 기능

```
Button shareButton = findViewById(R.id.shareButton);
shareButton.setOnClickListener((view) → {
        View container;
        container = getWindow().getDecorView();
        container.buildDrawingCache();
        Bitmap captureView = container.getDrawingCache();

        String strFolderPath = Environment.getExternalStorageDirectory().getAbsolutePath();
        File folder = new File(strFolderPath);
        if(!folder.exists()){
            folder.mkdirs();
        }
        String filepath = strFolderPath + "/" + System.currentTimeMillis()+ ".jpeg";
        File fileCache = new File(filepath);

        FileOutputStream fos;
        Uri uri = null;
        try{
            fos = new FileOutputStream(fileCache);
            captureView.compress(Bitmap.CompressFormat.JPEG,  quality: 100, fos);
            uri = Uri.fromFile(fileCache);
        }catch(Exception e){
            e.printStackTrace();
        }

       Intent msg = new Intent(Intent.ACTION_SEND);
       msg.putExtra(Intent.EXTRA_STREAM, uri);
       msg.setType("image/*");
       startActivityForResult(Intent.createChooser(msg,  title: "공유하기"),  requestCode: 0);
});
```

이미지를 저장 후 타 앱으로 공유!

계획대로 되었는가?

10월 12일 : 아이디어 구상 및 제안서 작성

10월 19일: 제안서 발표 + 기본 흐름 UI 제작

11월 2일: UI 구현 완료, NAVER 검색 구현

11월 16일: 중간 발표

12월 10일: 공유 기능 , 예산 선정 알고리즘

12월 17일: 최종 발표

실제 구현

10월 12일 : 아이디어 구상 및 제안서 작성

10월 19일: 제안서 발표 + 기본 흐름 UI 제작

11월 2일: UI 구현 완료, NAVER 검색 구현

11월 16일: 중간 발표

중간 발표 후 ~ 12월 : 쓰레드 구현

12월 10일: 공유 기능 , 예산 선정 알고리즘

12월 17일: 최종 발표

구현 완료!!!