SW 개발 현황
(14 주차)

프로젝트명: MBTI 진단 검사 어플리케이션 구현

참여 개발자			
학과	학번	이름	역할
컴퓨터공학부	202004484	조 용 찬	팀장
	201900776	김 보 석	팀원
	202001165	김 혜 리	팀원
	201902520	이 상 윤	팀원
	202002436	이 사 야	팀원

2022-1 학기 오픈소스 SW 및 실습과목

2022. 06. 09.

1. 목표 및 내용

1) 최종 목표

“ MBTI 진단 검사 Application 구현 ”

- Main UI 설계 (16 가지의 MBTI 소개 및 진단)
- Sub UI 설계 (진단 후 결과화면, 각 MBTI 의 자세한 설명)
- MBTI 관련 자세한 정보 수집
- 간단한 MBTI 관련 아이콘 설정
- 질문에 대한 답변에 따른 MBTI 결과 진단 알고리즘 구현

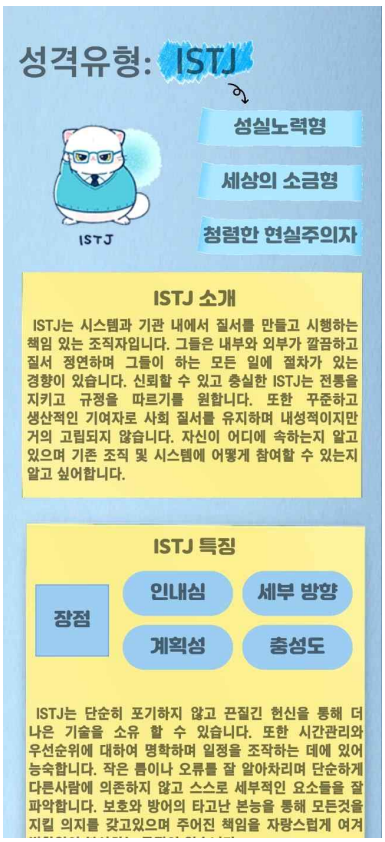
2. 세부 내용 개발 상황

- 14 주차 개발 목표 : 세부 특징 UI 정보 삽입 및 Java algorithms 구현

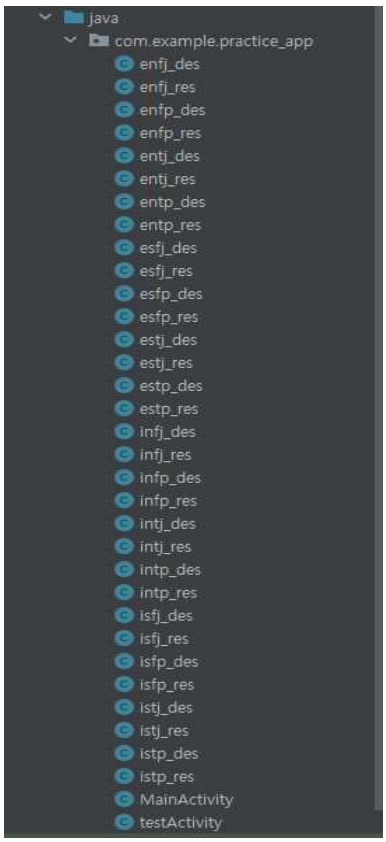
- Main UI 및 Sub UI 설계

팀장 조용찬 : App 의 기본 바탕이 되는 Main UI 의 Lavout 을 제작
팀원 이사야 : Sub UI 중 MBTI 의 세부적 특징 설명 Lavout 을 제작
팀원 김보석 : Sub UI 중 진단을 마친 후 도출되는 결과 화면 제작
Main UI 및 Sub UI 를 포함한 모든 UI 의 Lavout : ScrollView(Scroll bar 존재)

1. Sub UI - 각각의 MBTI 에 대한 세부특징 화면 : 16 페이지 정보입력 완료



2. 모든 UI 및 Java code 구현 완료



• **진단 알고리즘 구현**

팀원 이상윤, 김혜리 : 진단 알고리즘에 구현할 자료조사 및 알고리즘 구상

TestActivitv 설계도 (Test UI 의 동적 구현)

① 각각의 특성에 해당하는 변수 및 Point 설정, 동적연결의 ID 값 부여

- 각 특성의 point, percent 변수 설정

```
int inspection_number = 1;
int e_point, i_point, s_point, n_point, t_point, f_point, j_point, p_point = 0;
double percent_e, percent_i, percent_s, percent_n, percent_t, percent_f, percent_j, percent_p;

private Button test_btn_return;
private Button test_btn_next; // 다음으로
private RadioGroup Radio_group_ei, Radio_group_sn, Radio_group_tf, Radio_group_jp;
/* 특성끼리의 라디오 그룹*/
private RadioButton Radio_btn_e, Radio_btn_i, Radio_btn_s, Radio_btn_n, Radio_btn_t, Radio_btn_f, Radio_btn_j, Radio_btn_p;
/* 각 특성에 해당하는 라디오버튼*/
private TextView test_number_ei, test_number_sn, test_number_tf, test_number_jp; // 문제 상단 숫자
private TextView test_progress_1, test_progress_2; // 진행도 1 = (1/8), 2 = (8/64)
private TextView textView1,textView2,textView3,textView4,textView5,textView6,textView7,textView8;
```

• inspection_number : 질문지를 넘기기위한 변수 값 (=질문 page 번호)

- 해당하는 특성의 문항 및 그룹 ID 값 부여

```
Radio_group_ei=findViewById(R.id.Radio_group_ei);
Radio_group_sn=findViewById(R.id.Radio_group_sn);
Radio_group_tf=findViewById(R.id.Radio_group_tf);
Radio_group_jp=findViewById(R.id.Radio_group_jp);


Radio_btn_e = (RadioButton) findViewById(R.id.Radio_btn_e);
Radio_btn_i = (RadioButton) findViewById(R.id.Radio_btn_i);
Radio_btn_s = (RadioButton) findViewById(R.id.Radio_btn_s);
Radio_btn_n = (RadioButton) findViewById(R.id.Radio_btn_n);
Radio_btn_t = (RadioButton) findViewById(R.id.Radio_btn_t);
Radio_btn_f = (RadioButton) findViewById(R.id.Radio_btn_f);
Radio_btn_p = (RadioButton) findViewById(R.id.Radio_btn_p);
Radio_btn_j = (RadioButton) findViewById(R.id.Radio_btn_j);
```

② '다음으로' Button 의 동적연결

```
        test_btn_next = findViewById(R.id.test_btn_next);
        test_btn_next.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                int inspection_point = 0;//답하지 않은 문제가 있는지 확인(next누를때마다 0으로 초기화해서 4개안돌면 안넘어가고 Toast발
                if(Radio_btn_e.isChecked()){
                    e_point++;
                    inspection_point++;
                }
                if(Radio_btn_i.isChecked()){
                    i_point++;
                    inspection_point++;
                }
                if(Radio_btn_s.isChecked()){
                    s_point++;
                    inspection_point++;
                }
                if(Radio_btn_n.isChecked()){
                    n_point++;
                    inspection_point++;
                }
                if(Radio_btn_t.isChecked()){
                    t_point++;
                    inspection_point++;
                }
                if(Radio_btn_f.isChecked()){
                    f_point++;
                    inspection_point++;
                }
                if(Radio_btn_j.isChecked()){
                    j_point++;
                    inspection_point++;
                }
                if(Radio_btn_p.isChecked()){
                    p_point++;
                    inspection_point++;
                }

                if (inspection_point >=4 ){
                    inspection_number++;
                    next1();
                    Checkoff(); //다음으로 버튼 누르면 라디오버튼 체크가 해제
                }
                else{
                    Toast.makeText(getApplicationContext(), text: "답하지 않은 문제가 있습니다.", Toast.LENGTH_SHORT).show();
                }

            }
        });

    }
```

- 그룹화 되어있는 **Radio Group** 속 각 특성에 해당하는 **Radio Button** 의 기능화

  - 양자택일의 형태
  - 각 특성에 해당하는 답안 선택 시 해당특성 point 증가
  - inspection_point 라는 변수를 이용하여 모든 답안 답변 기능 구현
  - inspection_point >= 4 일 경우 질문지 넘기기, 답안 초기화 구현
  - inspection_point < 4 일 경우 Toast Message 출력

③ check off(), next1() 함수의 질문지 변형 code

```
public void Checkoff(){
    Radio_group_ei.clearCheck();
    Radio_group_sn.clearCheck();
    Radio_group_tf.clearCheck();
    Radio_group_jp.clearCheck();
}

public void next1() {
    final TextView answer_e = (TextView) findViewById(R.id.answer_e);
    TextView answer_i = (TextView) findViewById(R.id.answer_i);
    TextView answer_s = (TextView) findViewById(R.id.answer_s);
    TextView answer_n = (TextView) findViewById(R.id.answer_n);
    TextView answer_t = (TextView) findViewById(R.id.answer_t);
    TextView answer_f = (TextView) findViewById(R.id.answer_f);
    TextView answer_j = (TextView) findViewById(R.id.answer_j);
    TextView answer_p = (TextView) findViewById(R.id.answer_p);
    TextView test_number_ei = (TextView)  findViewById(R.id.test_number_ei);
    TextView test_number_sn = (TextView)  findViewById(R.id.test_number_sn);
    TextView test_number_tf = (TextView)  findViewById(R.id.test_number_tf);
    TextView test_number_jp = (TextView)  findViewById(R.id.test_number_jp);
    TextView test_progress_1 = (TextView)  findViewById(R.id.test_progress_1);
    TextView test_progress_2 = (TextView)  findViewById(R.id.test_progress_2);



    if (inspection_number == 2) {
        test_progress_1.setText("2/8");
        test_progress_2.setText("16/64");
        test_number_ei.setText("⑤");
        test_number_sn.setText("⑥");
        test_number_tf.setText("⑦");
        test_number_jp.setText("⑧");
        answer_e.setText("여러 사람들 사이에 끼어\n함께 대화를 나누는 편이다.");
        answer_i.setText("한 번에 한 사람씩\n대화를 나누는편이다.");
        answer_s.setText("현실 감각이 있는 사람과 잘 어울린다.");
        answer_n.setText("상상력이 풍부한 사람과 잘 어울린다.");
        answer_t.setText("꾸준하고 합리적인\n사람으로 불리는게 더 좋다.");
        answer_f.setText("솔직하고 감정적인\n사람으로 불리는게 더 좋다.");
        answer_j.setText("모임을 미리 여유있게\n계획하는 편이다.");
        answer_p.setText("모임은 상황에 따라 별\n계획없이 자유로운 편이다.");
    }
```

- **Checko off 함수:**
  - Radio Group의 옵션 : check 초기화

- **Next1 함수:**
  - final : 객체자체의 변경은 막지만 내부 값의 변경은 가능
  - inspection number가 증가함에 따라 Test page 내의 값의 변경

④ MBTI 의 각 4 가지 특성의 결정

```
        if (inspection_number == 9) {
            inspection_result();

        }
    }

    public void inspection_result() {
        String result_four_property;
        String first, second, third, fourth;

        if (e_point >= 5) {
            first = "e";
            percent_e = e_point * 12.5;
            percent_i = 100 - percent_e;
        } else {
            first = "i";
            percent_i = i_point * 12.5;
            percent_e = 100 - percent_i;
        }

        if (s_point >= 5) {
            second= "s";
            percent_s = s_point * 12.5;
            percent_n = 100 - percent_s;
        } else {
            second = "n";
            percent_n = n_point * 12.5;
            percent_s = 100 - percent_n;
        }

        if (t_point >= 5) {
            third = "t";
            percent_t = t_point * 12.5;
            percent_f = 100 - percent_t;
        } else {
            third= "f";
            percent_f = f_point * 12.5;
            percent_t = 100 - percent_f;
        }

        if (j_point >= 5) {
            fourth = "j";
            percent_j = j_point * 12.5;
            percent_p = 100 - percent_j;
        } else {
            fourth= "p";
            percent_p = p_point * 12.5;
            percent_j = 100 - percent_p;
        }

        result_four_property = first + second + third + fourth;
        intent_mbti(result_four_property, percent_e,percent_i,percent_s,percent_n,percent_t,percent_f,percent_j,percent_p);
```

- **inspection number =9:** 총 **8** 페이지의 모든 문항 답변시 결과지로의 이동 활성화

- **String(문자열)**을각각의 순서에 따라 **first. second. third. fourth** 로 지정
  - 순서에 맞게 특성의 답변 수에 따라 해당 특성으로 결정
  - 5 문제 이상 답변 시. 해당 특성으로 결정 그 외 반대특성 결정
  - 총 4 가지의 특성 결정 후 result four property 의 String 값으로 결정
  - 각 특성의 답변 수의 point 값을 답변 수 * 12.5 로 결정 및 함수로 송신

⑤ **intent mbti 함수 - 16 개의 intent 구현(해당되는 결과.res UI 로 이동)**

```
public void intent_mbti(String result_four_property, double percent_e, double percent_i, double percent_s,
    if (result_four_property.equals("istp")) {
        Intent intent = new Intent(getApplicationContext(), istp_res.class);
        intent.putExtra( name: "percent_e",percent_e);
        intent.putExtra( name: "percent_i",percent_i);
        intent.putExtra( name: "percent_s",percent_s);
        intent.putExtra( name: "percent_n",percent_n);
        intent.putExtra( name: "percent_t",percent_t);
        intent.putExtra( name: "percent_f",percent_f);
        intent.putExtra( name: "percent_j",percent_j);
        intent.putExtra( name: "percent_p",percent_p);
        startActivity(intent);
    }

    if (result_four_property.equals("istj")) {
        Intent intent = new Intent(getApplicationContext(), istj_res.class);
        intent.putExtra( name: "percent_e",percent_e);
        intent.putExtra( name: "percent_i",percent_i);
        intent.putExtra( name: "percent_s",percent_s);
        intent.putExtra( name: "percent_n",percent_n);
        intent.putExtra( name: "percent_t",percent_t);
        intent.putExtra( name: "percent_f",percent_f);
        intent.putExtra( name: "percent_j",percent_j);
        intent.putExtra( name: "percent_p",percent_p);
        startActivity(intent);
    }

    if (result_four_property.equals("isfp")) {
        Intent intent = new Intent(getApplicationContext(), isfp_res.class);
        intent.putExtra( name: "percent_e",percent_e);
        intent.putExtra( name: "percent_i",percent_i);
        intent.putExtra( name: "percent_s",percent_s);
        intent.putExtra( name: "percent_n",percent_n);
        intent.putExtra( name: "percent_t",percent_t);
        intent.putExtra( name: "percent_f",percent_f);
        intent.putExtra( name: "percent_j",percent_j);
        intent.putExtra( name: "percent_p",percent_p);
        startActivity(intent);
    }
```

- 4 가지 특성이 결정되고 결정된 String 값에 따라 해당 MBTI.res 로 이동

- intent.putExtra : 실수형의 특성 별 percent 값을 객체에 담아 이동

- 16 개의 MBTI res (Result UI) - Activity

① 적용할 **textvirw** 및 **progressbar** 의 동적연결 **ID** 값 부여

```
final TextView e_percent = (TextView) findViewById(R.id.e_percent);
TextView i_percent = (TextView) findViewById(R.id.i_percent);
TextView s_percent = (TextView) findViewById(R.id.s_percent);
TextView n_percent = (TextView) findViewById(R.id.n_percent);
TextView t_percent = (TextView) findViewById(R.id.t_percent);
TextView f_percent = (TextView) findViewById(R.id.f_percent);
TextView j_percent = (TextView) findViewById(R.id.j_percent);
TextView p_percent = (TextView) findViewById(R.id.p_percent);

final ProgressBar progressBar_e = (ProgressBar) findViewById(R.id.progressbar_e);
ProgressBar progressBar_i = (ProgressBar) findViewById(R.id.progressbar_i);
ProgressBar progressBar_s = (ProgressBar) findViewById(R.id.progressbar_s);
ProgressBar progressBar_n = (ProgressBar) findViewById(R.id.progressbar_n);
ProgressBar progressBar_t = (ProgressBar) findViewById(R.id.progressbar_t);
ProgressBar progressBar_f = (ProgressBar) findViewById(R.id.progressbar_f);
ProgressBar progressBar_j = (ProgressBar) findViewById(R.id.progressbar_j);
ProgressBar progressBar_p = (ProgressBar) findViewById(R.id.progressbar_p);
```

- Progressbar : 특정 값의 진척도, 분포도 효과적으로 표현
  • 실수형의 값만 적용 가능

② 수신되는 객체의 값을 받아온 후, **int(정수형)**으로 변형

```
Intent intent = new Intent(this.getIntent());
double percent_e = intent.getExtras().getDouble( key: "percent_e");
double percent_i = intent.getExtras().getDouble( key: "percent_i");
double percent_s = intent.getExtras().getDouble( key: "percent_s");
double percent_n = intent.getExtras().getDouble( key: "percent_n");
double percent_t = intent.getExtras().getDouble( key: "percent_t");
double percent_f = intent.getExtras().getDouble( key: "percent_f");
double percent_j = intent.getExtras().getDouble( key: "percent_j");
double percent_p = intent.getExtras().getDouble( key: "percent_p");

int per_e = (int)Math.ceil(percent_e);
int per_i = (int)Math.floor(percent_i);
int per_s = (int)Math.ceil(percent_s);
int per_n = (int)Math.floor(percent_n);
int per_t = (int)Math.ceil(percent_t);
int per_f = (int)Math.floor(percent_f);
int per_j = (int)Math.ceil(percent_j);
int per_p = (int)Math.floor(percent_p);
```

- intent.getExtras().getDouble : **객체에 담겨진 값을 intent 옵션으로 수신**
  • Double(실수형)의 값을 받아온 후 다른 객체로 선언

- Math.ceil (반올림) / Math.floor(반내림) :
  • Double(실수형)의 값의 소수점 제거로 int 형으로 변형 후 객체로 선언

③ 변환된 int형의 값을 % / porgressbar에 적용

```
e_percent.setText(per_e + "%");
i_percent.setText(per_i + "%");
s_percent.setText(per_s + "%");
n_percent.setText(per_n + "%");
t_percent.setText(per_t + "%");
f_percent.setText(per_f + "%");
j_percent.setText(per_j + "%");
p_percent.setText(per_p + "%");

progressBar_e.setProgress(per_e);
progressBar_i.setProgress(per_i);
progressBar_s.setProgress(per_s);
progressBar_n.setProgress(per_n);
progressBar_t.setProgress(per_t);
progressBar_f.setProgress(per_f);
progressBar_j.setProgress(per_j);
progressBar_p.setProgress(per_p);
```

3. 기타 사항

① **세부 특징 UI 정보입력** : App 에 적용을 위한 간단한 16 개의 MBTI 별 세부적 특징의 자료조사 및 정리를 위한 시간이 많이 소요되어 14 주차의 java code 의 구현과 더불어 자료조사 및 정리를 분담하여 적용할 수 있도록 세부 계획을 수정하고 보완하여 프로젝트 발표를 위한 시간 확보를 위해 세부 계획을 수정하여 분담을 진행하였음.

② **추후 보완사항 :** 총 8 개의 질문지를 생성하여 특성별 5 개의 답안을 선택 할 시, 해당 특성으로 결정되는 알고리즘이 특성별로 4 개씩 절반에 해당 되는 각각의 답안을 선택하면 항상 알고리즘에서 설정한 반대되는 특성으로 결정되는 특징을 보완하기 위해 문제지의 추가를 통해 더 정확한 MBTI 의 진단을 유도하도록 보완할 필요가 있음을 파악함.