ApiDate

API

- 완성된 코드를 가져다 쓰는 것

SDK

- 로직은 만들어져있지만 구체적인 디테일을 위해 추가적인 개발이 필요한 것

DATE 클래스를 사용하여 현재 시간 구하기

```
Date dateToday = new Date(); // 1)

 // Date 객체가 생성될 때 [= new Date()]
 // Date 객체 내부에 new Date()가 실행되었을 때의 시간이 저장된다.
        System.out.println(dateToday); // 2)

//  Thread.sleep(2000);

        System.out.println(dateToday); // 3) = 1)이 실행되었을때
```

Date 객체에 저장된 날짜를 원하는 날짜 포맷형식으로 변환하기

```
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm
System.out.println(sdf.format(dateToday));
// DATE 클래스를 사용하여 현재 시간 구하기에 사용했던 dateToday 객체를
// sdf라는 원하는 형태로 변환했다.

String strToday = sdf.format(dateToday);
        System.out.println(strToday);
```

Calendar 클래스 사용하여 현재 시간 구하기

```
Calendar calToday = Calendar.getInstance(); // new ~~ 실행된다는

        System.out.println(calToday);
```

Calendar 객체로부터 얻은 Date 객체를 날짜 포맷형식으로 변환하기

```
Date calToDate = calToday.getTime();
System.out.println(calToDate); // long 타입의 결과값 리턴

System.out.println(sdf.format(calToDate));
// 1970년부터 지나온 밀리세컨드를 출력하기때문에
// sdf.format()를 사용하여 보기 좋게 출력함.

System.out.println(sdf.format(calToday.getTime()));
// 위의 코드와 같은 결과
```

System 클래스 사용 및 날짜 포멧팅

```
// System 클래스 사용하여 현재 시간 구하기(밀리초로 리턴)
long longToday = System.currentTimeMillis();
        System.out.println(longToday);

// 날짜 포멧팅
System.out.println(sdf.format(longToday));
```

다양한 날짜 타입 만들기

```
// 1. 2024/01/19 12:12:30
        // 2. 24-01-19 오후 00:12:30
        // 3. 금요일 12:12:30
        // @. 2024.01.19 PM 00:12:30
        sdf = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss");
        System.out.println(sdf.format(dateToday));

        sdf = new SimpleDateFormat("yy-MM-dd a KK:mm:ss");
        System.out.println(sdf.format(dateToday));

        sdf = new SimpleDateFormat("E요일 kk:mm:ss");
        System.out.println(sdf.format(dateToday));

        sdf = new SimpleDateFormat("yyyy.MM.dd a KK:mm:ss", L
        System.out.println(sdf.format(dateToday));
```

날짜형 문자열(String) 을 Date 객체로 변환하기

```
// 날짜형 문자열을 담은 strTomorrow을 변환하고자 함.
String strTomorrow = "2024-01-23 09:07:20";

// 변환하고자 하는 문자열과 똑같은 날짜 포멧을 생성
sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

// sdf.parse()에 문자열 객체를 담아 포멧팅함
Date dateTomorrow = sdf.parse(strTomorrow);
        System.out.println(dateTomorrow);
```

시간형 문자열(String) 을 Date 객체로 변환하기

```
String strTime = "09:13:40";
        sdf = new SimpleDateFormat("HH:mm:ss");
        Date dateTime = sdf.parse(strTime);
```

```
        System.out.println(dateTime);
// Thu Jan 01 09:13:40 KST 1970 을 출력
// 날짜를 지정하지 않으면 기본값인 1970.01.01로 출력된다.
```

Calendar 객체 날짜 세팅하기

- Calendar 객체를 꺼낼때는 get,

  새로운 값으로 설정할때는 set을 사용한다.

```
Calendar cal = Calendar.getInstance(); // 실행되는 시점의 시간을 담

// 1998년 1월 31일
// month 부분은 1월이 0, 2월이 1, 3월이 2, ... 12월이 11
cal.set(1998, 0, 31);

// 1998년 1월 31일 21시 38분 52초
 cal.set(1998, 0, 31, 21, 38, 52);

 sdf = new SimpleDateFormat("yyyy.MM.dd HH:mm:ss");
 System.out.println(sdf.format(cal.getTime()));
// set으로 설정하고 get으로 꺼내야함


// Date 객체로부터 Calendar 객체 얻기
        Calendar calTemp = Calendar.getInstance();
        // Date 객체의 시간이 Calendar 객체에 저장된다.

        calTemp.setTime(dateTomorrow);
// dateTomorrow는 날짜형 문자열을 Date로 바꿨을때 사용한 객체
// Date dateTomorrow = sdf.parse(strTomorrow);
// System.out.println(dateTomorrow);
// "2024-01-23 09:07:20"를 Date 형식으로 변환한 값을 담고 있음
        System.out.println(sdf.format(calTemp.getTime()));
```

날짜 꺼내기

```
// 년도
        System.out.println(calTemp.get(Calendar.YEAR));

        // 월
        System.out.println(calTemp.get(Calendar.MONTH) + 1);

        // 일
        System.out.println(calTemp.get(Calendar.DATE));

        // 시간
        System.out.println(calTemp.get(Calendar.HOUR)); // hh
        System.out.println(calTemp.get(Calendar.HOUR_OF_DAY))

        // 분
        System.out.println(calTemp.get(Calendar.MINUTE));

        // 초
        System.out.println(calTemp.get(Calendar.SECOND));
```

Date 객체를 이용하여 디데이 계산기 만들기

2023.12.18 에 대해 + 35
2023.02.09 에 대해 -18
2024.01.20 에 대해 +2
2024.01.24 에 대해 -2

이 출력되도록 하는 디데이 계산기 만들어보기

```
Date todayDate = new Date(); // 오늘 2024.01.22

        sdf = new SimpleDateFormat("yyyy.MM.dd");

        String todayStr = sdf.format(todayDate); // 2024.01.2
```

```
        todayDate = sdf.parse(todayStr); // 2024.01.22 00:00:

        String before = "2024.01.20"; // 이틀전
        String after = "2024.01.24"; // 이틀후
        String start = "2023.12.18"; // 훈련시작일
        String newDay = "2024.02.09"; // 설


        Date beforeDate = sdf.parse(before);
        long calDiff = todayDate.getTime() - beforeDate.getTi
        calDiff = calDiff / (1000*60*60*24);
        System.out.println(calDiff);

        Date afterDate = sdf.parse(after);
        long dday = todayDate.getTime() - afterDate.getTime()
        dday = dday / (1000*60*60*24);
        System.out.println(dday);
        Date startDate = sdf.parse(start);
        Date newDate = sdf.parse(newDay);

        System.out.println((todayDate.getTime() - beforeDate.
        System.out.println((todayDate.getTime() - afterDate.g
        System.out.println((todayDate.getTime() - startDate.g
        System.out.println((todayDate.getTime() - newDate.get
```

Calendar의 날짜 연산

```
Calendar toCal = Calendar.getInstance();
        sdf = new SimpleDateFormat("yyyy.MM.dd HH:mm:ss");
        System.out.println(sdf.format(toCal.getTime())); // A

        // 3일 뒤 날짜로
        toCal.add(Calendar.DATE, 3);
        System.out.println(sdf.format(toCal.getTime())); // B
```

```
        // 20일 뒤 날짜로
        toCal.add(Calendar.DATE, 20);
        System.out.println(sdf.format(toCal.getTime())); // C

       // 7일 전 날짜로
                toCal.add(Calendar.DATE, -7);
        System.out.println(sdf.format(toCal.getTime())); // D

        // 1달 뒤 날짜로
        toCal.add(Calendar.MONTH, 1);
        System.out.println(sdf.format(toCal.getTime())); // F
```

달력 만들기

```
int year = 2024;
int month = 2;

  Calendar calendar = Calendar.getInstance();
  calendar.set(year, month - 1, 1); // Calendar 객체에서 Month는
  System.out.println(sdf.format(calendar.getTime()));

  // 해당 월의 1일이 무슨 요일에 시작하는지 확인
  // 1: 일요일, 2: 월요일, 3: 화요일, 4: 수요일, ... 7: 토요일
  int startDay = calendar.get(Calendar.DAY_OF_WEEK);
  System.out.println(startDay); // 5 출력, 일요일부터 시작하므로

  // 해당 월의 마지막 일자가 언제인지
  int lastDay = calendar.getActualMaximum(Calendar.DAY_OF_MON
  System.out.println(lastDay);

  System.out.println(year +"년 " + month + "월 달력");
  System.out.println("일\t월\t화\t수\t목\t금\t토");

  int day = 1;
  for (int i = 0; i < 42; i++){
```

```
        // startDay가 5(목요일) 일때 i=4에 해당
        // i가 0, 1, 2, 3 일때는 그리지 않음
        if (i < startDay - 1){
            System.out.print("\t");
        } else {
            System.out.print(day + "\t");

            if (day == lastDay){
                break;
            }
            day++;

        } // i가 6,13, 20, 27, ... 일때 줄바꿈 넣기
        if (i % 7 == 6){
            System.out.println();
        }

    }
```