2021 소프트웨어 나눔 축제 강의자 대본

발표자 : 양현준, 최재혁

(대충 정돈된 후)

PPT 1페이지

지금부터 애플파이와 함께 안드로이드 스튜디오와 자바를 이용한 날씨앱 만들기 수업을 시작하도록 하겠습니다

PPT 2페이지

먼저 목차부터 보겠습니다

첫 번째로는, 축제를 운영하고 있는 애플파이 동아리와, 이번 나눔축제 강의를 맡게 된 강의자들을 소개해드리겠습니다

두 번째로는, 강의자들을 소개하였으니, 저희가 여러분을 알아가야 할 차례일 것 같습니다. 간단하게 준비한 자기소개 미니게임을 통해 서로 알아가는 시간을 가져보아요

세 번째로는, 본격적으로 안드로이드 스튜디오 강의를 시작합니다. 안드로이드 스튜디오가 무엇인지부터 차근차근 배워나가 보도록 하겠습니다

네 번째로는, XML과 미니 테스트 과정을 가져보겠습니다. 흔히 말하는 레이아웃에 해당하는 코드들은 어떻게 작성되는지 XML을 통해 알아보고, 배운 내용을 바탕으로 직접 A4용지 만들기 미니 테스트 과정을 작성 해보겠습니다

다섯 번째로는, 자바코드를 설명하겠습니다. 앱에서 기능이 작동하려면 화면도 물론 중요하지만, 그 뒤에서 작동하는 명령이 있어야지 실행을 할 수 있을 것입니다. 미리 작성해둔 자바 코드를 보면서 날씨앱이 어떤 형식으로 작동하는지 알아보도록 하겠습니다.

여섯 번째로는, 앱을 완성하는 작업입니다. XML파일과 자바 소스코드를 활용하여 최종적인 날씨 어플리케이션을 만들어 내어 제대로 구현해 내었는지 확인하는 것이 마지막 목표입니다

PPT 3페이지

이 강의를 진행하는 동아리와 강의자들을 소개하겠습니다! 먼저, 동아리부터 소개하도록 할게요. 애플파이는 선린인터넷고등학교 동아리 중에서 신나고 즐거운 분위기와 함께 개발을 해나갈 수 있는 소프트웨어과의 최고의 동아리입니다! 애플파이는 어플리케이션을 만드는 선린 최초의 모바일 컨텐츠 개발 및 창업 동아리입니다. 자세한 애플파이의 활동이나 수업 과정 등을 확인하고 싶으시다면 사이트를 참고해주세요

PPT 4페이지 (양현준 소개)
소프트웨어과 최단신으로 불리는 대표 탈모인, 양현준입니다.
키가 지금 수업을 듣고 있는 여러분들보다 작을 수 있다는 점 특징 알아두고 가면서
수업을 들으시면 좋을 것 같아요

PPT 5페이지 (최재혁 소개)
소프트웨어과에서 디자인을 유사하게 하는 것으로 활용하고 있으며, 앞에 소개했던 친구보다는 키가 크고, 탈모를 걸리지 않았다고 볼 수 있어요!

PPT 6페이지 (개인 별 소개)
이제 여러분들이 누구인지 알아보아야 할 시간인 것 같아요
간단하게 준비한 미니게임을 통해 저희가 여러분을 알아보도록 하겠습니다

자기소개 게임을 설명해볼게요
1번부터 마지막 번호까지 순서대로 자기소개를 시작하는데, 자기소개를 하나씩 순서대로 늘려나갈 거에요! 예를 들어 보자면, 첫 번째 사람이 이름을 얘기하였다면, 두 번째 사람은 이름도 말하고, 다른 나이나, 학교나, 진로희망이나, 키라던가, 이상형 등등을 정해서 자기소개를 해볼게요

(끝났다고 치고)
소개를 잘 들었습니다!
간단하게 발표자도, 수업을 듣는 여러분도 같이 알아갈 수 있었던 시간인 것 같아요

이제 수업을 본격적으로 시작 해볼텐데,
수업을 들으면서 모르는 것이나 안되는 것이 있다면 바로바로 채팅을 쳐서 질문하거나,
강의자가 물어보는 것에 대해 적극적으로 대답해 주시면 좋겠습니다!

PPT 7페이지 (안드로이드 스튜디오 소개)
지금부터 만지게 될 환경을 소개 해볼게요. 저희는 '안드로이드 스튜디오' 라는 프로그램을 이용하여 코드를 확인해보게 될 것인데, 안드로이드라는 운영체제는 모바일에서 선두주자로 달리고 있는데, 구글이 인수하여 추후 IntelliJ IDEA 기반으로 만든 IDE가 안드로이드 스튜디오입니다

여기서 약간 어지러울 수 있는 단어가 있을 것 같아요.

먼저 운영체제는 대표적인 것이 윈도우, 안드로이드 등의 우리가 사용하고 있는 컴퓨터나 스마트폰을 사용하기 위해 기본적인 소프트웨어를 제공해주는 것이라고 생각하면 되고,

IntelliJ IDEA는 JetBrains 이라는 회사에서 제작한 자바, 코틀린의 개발 환경이에요.
쉽게 생각하자면 엔트리나 스크래치 같은 개발 환경을 제공해주는 회사라고 생각하시면 됩니다.

자바를 들어보신 적이 있으신 분들이 있나요? *(여기서 대충 난이도 조절해야됨)*

(많다면) 의외로 많은 분들이 자바를 아신다고 하시는데, 혹시 설명해줄 수 있는 분이 있을까요?
<없으면> 없는 것 같으니, 설명을 직접 해주도록 하겠습니다. 자바라는 것은 프로그래밍 언어로써, 우리가 일상생활에서 사용하는 각국의 언어처럼 기계에도 명령을 내릴 수 있게 하는 언어가 존재합니다. 그러한 언어 중에 하나에 해당하는 것이 자바라고 생각하시면 될 것 같습니다.
<있으면> 설명 조금만 보충해주기 (프로그래밍 언어에 대해서)

(적다면) 위 지문을 활용하여 강의하기.

마지막으로 IDE는, 아까 개발 환경을 제공해주는 회사가 JetBrains라고 했죠? 그런 것처럼 개발을 할 수 있는 환경을 제공해주는 곳을 통합 개발 환경(IDE)라고 부른답니다

여기까지 정리해보면, 우리는 안드로이드 스튜디오라는 프로그램을 사용하게 되는데,
이 프로그램은 우리가 개발을 할 수 있도록 도움을 주는 프로그램이고,
Java와 Kotlin 이라는 언어를 사용하여 프로그램이 작동하도록 명령을 내릴 예정입니다

이 중에서 우리가 사용할 언어는 Java가 되는 것이에요!

여기까지 예상시간 : 1교시 반 ~ 1교시 끝.
(13:30 ~ 13:50 정도에 끝났다면 정상 / 13:50분이라면 10분 휴식 가지기)

PPT 8페이지 ~ 16페이지 (안드로이드 스튜디오 설치)
안드로이드 스튜디오를 사전에 설치를 해오라고는 하였는데, 설치하는 과정을 다시 한 번 보면서 잘못된 점이 없는지 확인해보도록 하겠습니다

(슬라이드를 모두 확인한 뒤) 여기까지 모두 다 되어있는 것 맞나요? *(안되어있으면 소회의실)*

PPT 17페이지 (XML 소개부분)
이제 우리는, 앱에서 화면을 구성하는 요소들을 확인하고, 직접 코드를 짜보면서 활용을 해보도록 하겠습니다. 그렇다면 먼저 용어에 대해 알아봐야겠죠? XML이라는 것에 대해 먼저 알아보겠습니다.

XML은 eXtensible Markup Language 의 줄임말로 안드로이드에서 주로 사용하는 태그를 이용하는 언어입니다. 여기서 태그란 부등호로 이루어진 괄호 안에 내용이 적혀진 것이라고 생각하시면 됩니다. 다음 페이지에서 예시를 보여 드릴테니, 형태만 알고 가시면 좋을 것 같습니다.

앱 구성 화면은 크게 두 가지로 나누어지는데, 첫 번째로는 앱의 화면구성으로 볼 수 있습니다.
우리가 시각적으로 보는 폰트나, 사진 같은 디자인이라는 요소에 해당하는 것이 있고,
두 번째로는 앱의 기능으로 볼 수 있습니다. 여러분들이 학교에 갈 때마다 사용하는 자가진단 앱에도 비밀번호 입력과 로그인 기능이 있듯이, 기능이라는 요소까지 이렇게 두 가지로 나눌 수 있습니다.

그렇다면 이 XML은 어디에 해당하는 언어냐, 바로 디자인을 하게 해줄 수 있는 언어에 해당하는 것이 되겠습니다.

PPT 18페이지 (태그 및 속성)
전페이지에서 태그 관련 이야기를 하다가, 이번 페이지에서 보여준다는 것 기억이 나시나요?
XML은 꺽쇠괄호로 감싼 태그 안에서 항목명과 값 형식을 지정하는 속성으로 이루어져 있습니다
우측 상단에 있는 코드를 보면서 설명해보자면, 태그에 해당하는 것은 Linear Layout이고, 속성에 해당하는 것은 xmls:android ~ 부분인 것을 알 수 있습니다.
당연하게도, 지금 태그와 속성에 해당하는 것들을 그림만 보고는 알 수 없을 테니, 태그가 어떤 형식으로 작성해야 하는지 조금 더 자세히 보도록 하겠습니다

PPT 19페이지 (시작태그 및 종료태그)
전 페이지에서 꺽쇠괄호로 감싼 태그 안에서 코드를 작성한다고 했었죠?
첫 번째로, 시작태그입니다. 시작태그는 꺽쇠괄호를 여는 것이 시작하는 것이고, 연 괄호에 태그의 종류와 속성을 나열한 뒤 다 나열을 마쳤다면, 닫는 꺽쇠 괄호를 이용해 닫게 됩니다.
태그안의 종류와 속성을 모두 나열하였다면 종료를 하여 마쳐야겠죠?
종료태그는 꺽쇠를 열어 / (슬래쉬)를 입력하여 태그의 종류를 입력한 뒤, 꺽쇠 괄호를 닫아주면

정상적으로 태그가 종료되게 됩니다.
태그의 종류를 입력하여 종료할 수도 있지만, 시작태그와 종료태그 사이에 아무 코드가 없을 경우 / (슬래쉬)를 붙이고 꺽쇠를 닫음으로서 종료태그를 생략할 수도 잇습니다
오른쪽 예시를 자세히 살펴보자면 LinearLayout으로 시작한 시작태그가 있고, xmls:android~로 속성을 적어준 뒤에, 바로 종료태그가 아닌, 꺽쇠를 닫아주었고 다음줄에 LinearLayout의 종료태그가 있다는 것을 알 수 있는 부분입니다.

PPT 20~23페이지 (레이아웃의 종류)
방금 전, LinearLayout이라는 태그를 봤었는데 이 레이아웃이라는 것이 무엇일까요?
(없으면 답변으로 넘어가도록 하기)

> 레이아웃은, 사전을 찾아보면 이런 뜻으로 나오게 됩니다.
명사 (책·정원·건물 등의) 레이아웃[배치] -- 즉, 무엇인가를 배치하는 것인데, 우리는 앱에서 화면을 구현하기에 화면의 요소들을 배치하는 역할을 하는 것이 레이아웃 이라고 생각하시면 됩니다. 배치하는 것에 따라 종류가 굉장히 많다고 할 수 있는데, 자세하게 알아보겠습니다

< 20P : ConstraintLayout >
첫 번째로, ConstraintLayout으로는 연결선이라는 요소를 이용한 배치입니다. 그림을 자세히 보시면 A와 B의 각 끝에 연결선이 있고, 그 연결선이라는 요소를 통해 배치된 것을 알 수 있는 레이아웃의 형태입니다.

< 21P : RelativeLayout >
두 번째로는, 상대적인 거리를 이용한 배치인 RelativeLayout입니다. 전페이지와는 사뭇 다른 느낌을 주는 그림을 보여주었는데요, 간단하게 예시를 들어 설명해보겠습니다.
들어가야 하는 레이아웃에 날짜가 있고, 시간이 있다고 했을 때 날짜라는 요소가 이미 배치되어서 기준이 되어 준다면, 시간이라는 요소를 그 왼쪽에 배치하는 것이 현재 페이지의 그림입니다
상대적인 거리를 이용하여 뷰들의 위치를 정해주는 레이아웃이라고 생각하시면 되겠습니다.

< 22P : FrameLayout >
세 번째로는, FrameLayout입니다. 이 레이아웃은 여러 뷰를 중첩할 때 쓰게 되는데요, 보이시는 그림과 같이 여러 뷰들을 중첩해야 할 때 사용하기 좋은 레이아웃이라고 생각하시면 됩니다.

< 23P : LinearLayout >
마지막으로 볼 레이아웃은, LinearLayout입니다. 이 레이아웃은 지정한 위치에서, 지정한 방향으로 뷰를 쌓아가는 레이아웃이라고 보시면 됩니다. 그림을 보면 Horizontal 같은 경우는, 왼쪽에서 시작하여 오른쪽(가로)로 배열되는 레이아웃인 반면, Vertical 같은 경우 아래(세로)로 배열되는 레이아웃임을 볼 수 있습니다.

우리는 마지막으로 살펴본 LinearLayout을 이용하여 예제를 풀어보도록 할 것입니다
관련된 속성들을 함께 알아보면서 예제를 풀기 위한 준비를 시작해보도록 하겠습니다!

PPT 24페이지 (레이아웃의 속성)
현재 PPT 페이지에서 보이는 것처럼 속성은 굉장히 적다고 생각할 수 있지만, 이외에도 많은 것이 있으나 오늘 사용할 필수적인 요소들만 모아서 설명을 드린 후, 추가적으로 필요한 부분이 있다면 설명을 더 드리도록 하겠습니다

< Layout 1 : 너비와 높이 정하기 >
레이아웃의 속성에서 처음으로 알아볼 것은 레이아웃의 너비와 높이 정하기입니다.
이것은 우리가 공예품이나, 무엇을 만드려고 할 때 기획안에서 가로 세로 높이는 정해놓고 만드는 것처럼 앱 또한 디자인을 할 때 어느 한 레이아웃의 높이와 너비를 정해주는 것은 기본적인 일이라 할 수 있습니다. 여기서 너비는 width 속성으로, 높이는 weight 속성으로 정해주게 되며, 정하는 방법으로는 다음과 같습니다!

match_parent -> 부모 태그가 가진 화면을 꽉 채워서 너비 혹은 높이를 정함
wrap_content -> 해당 레이아웃이 그려질 수 있도록 필요한 길이만 사용하는 것입니다.
이외에도, 직접 단위를 정해줄 수도 있는데 이 때 단위로는 px, dp, sp가 있습니다

다음으로는, 뷰가 쌓이는 방향에 대해 알아보겠습니다!

< Layout 2 : 뷰가 쌓이는 방향 정하기 >
아까 LinearLayout을 설명하면서 orientation 관련한 내용을 잠시 봤었는데 기억 나시나요?
(대답을 받았다면 다음으로 넘어가기 - 방향까지 기억한다면 아주 좋음.)

그렇습니다. 아까 전 LinearLayout을 소개해보면서 Horizontal과 Vertical의 차이점을 확실히 알 수 있었는데요 Horizontal은 가로 방향으로 뷰를 쌓는 속성이고, Vertical은 세로 방향으로 뷰를 쌓는 속성이었습니다.

방향도 정했으니, 이제 비중을 설정해볼까요?

< Layout 3 : 비중 정하기 >
비중이란 무엇일까요? 쉽게 생각하면 비율이라고 볼 수 있을 것 같습니다.
우리가 스마트폰을 켰을 때 바로 나오는 메인화면을 구현한다고 생각 해봤을 때, 자신이 설정한 위젯의 크기와 앱을 바로가게 해주는 크기는 서로 다른 비중(비율)을 차지한 것을 확인할 수 있습니다.

이렇게 스마트폰에서도 쉽게 비중을 찾을 수 있듯이, 앱 내부에서도 최대 비중과, 부모 레이아웃에서의 자신의 비중을 설정할 수 있는데요, 활용 방법은 다음과 같습니다

android:layout_weight="1" android:weightSum="2"
최대 비중을 2로 설정하고, 부모 레이아웃에서 자신의 비중을 1로 설정한 코드입니다

PPT 25페이지 (A4예제 만들기)
교재를 한페이지 넘겨보시면, 코드가 굉장히 길게 되어 있는 부분을 볼 수 있습니다. 우리가 만들게 될 코드인데요, 교재 상에서 문제가 되어 있는 부분부터 고치도록 하겠습니다

```
<?xml version="1.0" encoding="utf-8"?>
 <LinearLayout xmins:android="http://schemas.android.com/apk/res/android"
 xmlns:app="http://schemas.android.com/apk/res-auto"(기존 없었던 부분)
 xmlns:tools="http://schemas.android.com/tools"
 a(밑줄로 내려야함)ndroid:layout_width="match_parent"
 android:layout_height(기존 heigtht)="match_parent"
 tools:context=".MainActivity"
 android:orientation(기존 orientaation)="vertical"
 android:weightSum="2">

 <LinearLayout
 android:layout_width(기존 띄어쓰기 되어있음)="match_parent"(기존 띄어쓰기 되어있음)
 android:layout_height="0dp"
 android:layout_weight="1"
 android:weightSum="2"
 android:orientation="horizontal">
 </LinearLayout>

 <LinearLayout
 android:layout_width(기존 언더바 두 개)="match_parent"
 android:layout(기존 t 두 개)_height="0dp"
 android:layout_weight="1"
 android:background="#F44336"
 android:gravity="center">

 <TextView
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:text="A1" android:textSize="70dp"
 android:textColor="#0000000"/>
 </LinearLayout>
</LinearLayout>
```

예제 코드를 위와 같이 보면서, 같이 해석 해나가도록 하겠습니다.
교재가 문제가 있었던 부분 사과드리면서, 문제가 있었던 부분을 그대로 칠 경우 오류가 발생하니, 수정해드린 위의 코드를 참고하여 XML을 작성하시면 되겠습니다.
이제, 저 코드들이 무엇을 의미하는지 알려드리도록 하겠습니다

< Layout 1 : 기본 설정 >

첫 번째 줄부터 보겠습니다
<?xml version="1.0" encoding="utf-8"?>
xml의 버전과, 인코딩 방식을 utf-8방식으로 정한 것을 알 수 있습니다

<LinearLayout xmins:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
레이아웃은 Linear Layout(배치 방법을 어떻게 작성하느냐에 따라 달라지는 레이아웃)을 시작 태그로 시작하였고, app과 툴과 관련된 내용을 받아온 것을 알 수 있습니다.

android:layout_width="match_parent"
android:layout_height="match_parent"
레이아웃의 너비와 높이를 정해준 것을 알 수 있는데, 여기서 match_parent를 사용했으므로, 자신이 가진 화면을 꽉 채워준다는 내용으로 선언한 것을 알 수 있습니다.

tools:context=".MainActivity"
tool이 어느 Activity를 만든 것인지 알아보는 것인데, 이것은 조금 있다 xml을 마치고 나서, Android Studio의 구조를 살펴보면서 조금 더 자세히 설명 드리겠습니다.

android:orientation="vertical"
orientation은 뷰가 쌓이는 방향을 설계해주는 부분이였죠? vertical은 세로로, Horizontal은 가
가로로 뷰가 쌓인다는 점을 이용해보면, 이번 vertical은 세로로 뷰가 쌓인다는 사실을 알 수 있습니다

android:weightSum="2“
레이아웃에서 비중을 정하는 것 중에서 weightSum에서는 레이아웃의 최대 비중을 설정해주는 코드입니다

>
종료 태그가 아닌, 시작 태그에서 꺽쇠 괄호를 닫기만 한 것을 알 수 있습니다

< Layout 2 : 레이아웃의 비중 정하기 >

이제, 다음 LinearLayout 코드를 보겠습니다. 이제 설명을 여러 번 들었으니, 여러분들이 의미를 맞춰볼 수도 있겠죠? *(각 코드마다 무슨 의미인지 물어보기 - 물어본 다음 내용 서술하기)*

<LinearLayout

전 코드와 마찬가지로, LinearLayout 예제이므로 시작태그가 LinearLayout임을 알 수 있습니다

android:layout_width="match_parent"
android:layout_height="0dp"

가로와 세로를 지정해주는 width와 height에서, width는 match_parent로 꽉 채운 것을 알 수 있고, height, 즉 세로는 0dp로 채워지지 않은 것을 알 수 있습니다
그렇다면 이런 질문을 할 수 있겠죠? 세로는 뭘로 채워주나요? 라는 질문 말이죠.
다음줄에서 같이 알아보도록 하겠습니다

android:layout_weight="1"
android:weightSum="2"

레이아웃의 비중을 맞춰주고 있는 것을 알 수 있습니다. weightSum은 2로, 최대 비중을 잡아주고 있고, 부모에 맞는 레이아웃의 weight값은 1로 지정하였음을 알 수 있습니다.

android:orientation="horizontal">

방금 전 LinearLayout의 코드에서는 orientation이 vertical이었는데, 이번에는 horizontal 이라는 것을 알 수 있습니다. 이 horizontal은 가로로 쌓아주는 것이였으므로 가로로 쌓는다는 의도를 가진 코드라고 볼 수 있습니다.

</LinearLayout>

전 코드와는 달리, 종료 태그가 있다는 것을 알 수 있는데, 꺽쇠 괄호 안에 / (슬래쉬)를 입력하고, 시작태그에 담아졌던 내용을 종료태그에 입력한 코드입니다.

< Layout 3 : 비중에 따라 용지 가져오기 >

이제, 다음 LinearLayout 코드를

<LinearLayout

LinearLayout으로 시작 태그를 새로 작성한 것을 알아볼 수 있습니다.

android:layout_width="match_parent"
android:layout_height="0dp"

가로는 화면을 꽉 채워주고, 세로같은 경우는 0dp로 설정하였습니다.

android:layout_weight="1"
android:background="#F44336"

android:gravity="center">
레이아웃에서 weightSum 값을 전에 정해주었죠? 거기서 얼마나 차지할 것인지 weight을 통해 나타내어 주엇씁니다. 그리고, 배경과 중앙으로 배열해주는 코드를 작성 해주었습니다.

<TextView
이번에는 Text를 입력 해보아야 하므로, TextView를 설정 해주었습니다.

android:layout_width="wrap_content"
android:layout_height="wrap_content"
width(가로)와 height(세로)를 필요한 길이만 가지도록 설정 해주었습니다.

android:text="A1" android:textSize="70dp"
android:textColor="#0000000"/>
텍스트를 A1으로, 텍스트의 크기를 70dp로, 텍스트의 색상을 검정색으로 설정해준 모습입니다

</LinearLayout>
LinearLayout을 종료하는 종료태그입니다. 그런데 종료태그가 두 개 있는 모습이 보이죠?

</LinearLayout>
이 종료태그는, 처음 시작 부분의 LinearLayout 종료태그입니다.
시작부분에서는 >, 꺽쇠괄호가 닫힌 부분만 있었고, 코드를 모두 작성하였으니 종료태그를 입력 해준 것이 되는 것입니다.

이제 각 부분이 어떤 의미를 가지는 코드들인지 알아 보았으니, 이 코드들을 바탕으로 하여 A4용지 예제를 만들 수 있을 것 같습니다. 교재에 첨부된 그림을 참고하여, A4용지 예제를 직접 만들어보세요! 모르는 부분은 질문하셔도 되며, 시간은 넉넉하게 드리겠습니다.

XML 파일 짜는 시간 20~30분 정도

PPT 26~28페이지 (깃허브 프로젝트 불러오기)
이제 앱 프로젝트를 받아오도록 하겠습니다. 주어진 깃허브 링크를 들어가서 다운로드 ZIP을 눌러, 프로젝트를 다운로드를 받아줍니다. 그 뒤, 안드로이드 스튜디오에서 프로젝트 Open을 통해 프로젝트를 열어주도록 하겠습니다.

프로젝트를 다 열었나요 *(확인 후 넘어가기)*

PPT 29페이지 (프로젝트 설명)
안드로이드 스튜디오를 이제 열어서 앱에 관련된 설명을 시작하도록 하겠습니다.
먼저, 앱의 구조부터 보도록 하죠. 첫 번째는 Menifest입니다. 여기서는 프로젝트 Activity가 무엇이 있는지 등의 기본 정보를 담고 있는 파일이라고 생각하시면 됩니다.

두 번째로는 Activity입니다. 이 Activity는 코틀린이라는 언어 혹은 자바라는 언어로 개발을 할 수 있는데, 자바로 코드를 짰기 때문에 확장자를 봤을 때 java라는 문구를 볼 수 있습니다

세 번째로는, XML 파일입니다. 화면 디자인을 해주는 역할이 XML이라고 했었는데, 예제를 풀면서 디자인에 관련한 내용 들이 모두 이 안에 들어가 있음을 알게 되었을 수 있습니다.

네 번째로는, gradle입니다. 프로젝트 전체의 버전 관리나 기능을 추가하여 쓸 것이 있을 때 입력하여 사용할 수 있는 부분입니다.

이렇게 네 가지를 크게 작성을 마쳤다면, 실행을 통해 테스트하거나 빌드를 해봐야겠죠?
테스트를 할 수 있는 곳은 오른쪽 위에 망치모양, 화살표 모양이 있을텐데요
여기서 초록색 망치 모양은 단순히 빌드만 하는 것이고, 기기를 설정 한 후 오른쪽에 있는 초록색 화살표를 통해 설정한 기기로 자신이 만든 앱을 테스트 해볼 수 있습니다.

PPT 30페이지 (날씨앱의 XML)
A4용지 예제를 통해 XML에 대해 자세히 알아봤었죠? 이제 실전으로 들어가서 실전 예제를 살펴보도록 하겠습니다. 처음 우리의 목표였던 날씨앱의 XML 파일의 빈칸들을 채워보는 과정을 해볼텐데요, 우리가 배운 내용 안에서 충분히 맞출 수 있는 예제이니, 직접 해보시기 바랍니다.

XML 파일을 확인해보면 [빈칸] 이라고 적혀져 있는 곳을, 직접 코드를 보면서 알맞게 채워주시면 될 것 같습니다. 힌트를 드리자면, 아래와 위 코드를 적절히 확인해가면서 코드를 입력하신다면, 쉽다고 느끼실 수 있을 겁니다!

XML 파일 짜는 시간 20~30분 정도 (여기까지 3~4교시면 좋음)

PPT 31~34 페이지 (날씨앱의 XML)
이제 코드를 다 짜셨나요? *(다 짠 것을 확인한 후 진행하기 바람)*
코드를 하나하나 살펴보면서 무엇이 있었는지 확인해보도록 하겠습니다.

날씨앱의 기능은 크게 4가지로 나눠볼 수 있습니다. 하나하나 살펴보자면,

< 빈칸 첫 번째 : 현재 위치(지역)을 표시해주는 텍스트 >
첫 번째로 현재 위치(지역)을 표시해주는 곳입니다.

<빈칸
빈칸이 처음부터 뚫어져 있어, 알기 어려운 부분이 있으므로 우리는 다음 줄부터 살펴보면서 이 빈칸에 무엇이 들어가야 할지 알아보도록 하겠습니다.

```
android:layout_width="wrap_content"
```

android:layout_height="wrap_content“
레이아웃의 width(가로)와 height(세로)를 지정해주는 부분입니다. wrap_content로 가로와 세로를 설정해준 것을 알 수 있습니다.

android:layout_gravity="center“
비율을 지정해주는 gravity를 center로 설정해 요소들이 균일하게 쌓이게 됩니다

다음 줄부터는, text를 지정해주는 코드라는 것을 알 수 있는 부분입니다.
android:text="서울“
첫 번째부터 알아보자면, text = 서울이라는 것은 서울이라는 텍스트를 지정해주었고,
android:textSize="30sp“
textSize, 즉 글씨체의 사이즈는 30sp로,
android:fontFamily="@font/notosanscjkkrmedium“
fontFamily는 글씨체 설정이라고 생각하시면 되고, notosans 글씨체를 이용했습니다.
android:includeFontPadding="false“
이 부분이 어려웠을 수도 있을 것 같은데, 이 부분 같은 경우에는 폰트에 여백을 주는 것을 설정하지 않았다고 생각하고 넘어가시면 될 것 같습니다.
android:textColor="#ffffff“
textColor 부분은, 색상 설정 부분인데 하얀색으로 설정한 것을 알 수 있습니다.
색상코드가 추가적으로 궁금하시다면 XML파일에서 색깔을 확인해볼 수도 있습니다
(코드 왼쪽 조그마한 사각형 모양)
/>

그렇다면 이번 코드에서 빈칸은 무엇이었을까요? 당연하게도, Text를 지정해주었으니 Text와 관련된 시작태그를 이용해 주었어야 할 것입니다. 아래 코드를 본다면 답이 나와있을 수 있다고 생각할 만큼 쉬웠는데요, 바로 TextView였습니다.
아래 코드에 비슷한 코드가 다시 나오는데, 다시 보면서 설명을 해보도록 하겠습니다.

<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center“
아까와 같은 부분입니다. 시작 태그는 TextView로 설정 되어 있는 것을 알 수 있고, width와 height 또한 전 코드와 동일하도록 wrap_content로 설정해주었으며 center로 layout의 위치를 설정 해준 것을 알 수 있습니다.

android:layout_marginTop="5dp“
아까와 다른 부분이 나왔는데요, 이 부분같은 경우에는 마진이라는 부분을 준 것입니다 쉽게 생각한다면, 이 5dp라는 크기만큼 여백을 주었다고 생각하시면 될 것 같습니다.
여백을 준 이유는, 레이아웃이 서울이라는 글씨 밑에 존재해야 하기에, 위에 여백을 5dp만큼 줌으로써, 아래로 내려가도록 해준 것입니다

android:text="대한민국"
android:textSize="20sp"
android:fontFamily="@font/notosanscjkkrmedium"
android:includeFontPadding="false"
android:textColor="#ffffff"
아래 부분 또한 위와 같은 코드입니다. 다시 한 번 살펴보자면,
text를 지정해주고, textSize를 크기와 수치를 입력해 설정하였으며
글씨체를 설정하였고, padding 설정을 하지 않도록 하였고, 색상을 지정하였습니다.
/>

여태까지 작성해준 것을 정리해보면, 서울이 위에, 대한민국이 아래에 중앙에 각각 배치가 되어 있으며, 색상은 흰색으로 설정 해주었다고 생각하시면 될 것 같습니다.

< 빈칸 두 번째 : 아이콘을 지정해주는 코드 >

이 부분이 그림이 나오지 않아서 조금 어려웠을 수도 있을 것 같습니다.
A4용지 예제에도 Image가 있지 않았기 때문이었죠. 코드를 함께 살펴보도록 하겠습니다.

<빈칸
android:id="@+id/weatherIcon"
이번에는 레이아웃에서 id를 지정해주었는데, weatherIcon이라는 ID를 지정해주어,
JAVA 코드 상에서 활용할 수 있도록 선언해준 부분입니다.

android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginTop="35dp"
android:layout_gravity="center" />
이 부분들은 전 코드에서 있었던 부분이었는데, 달라진 부분은 마진의 값이 변경된
점 이라는 것을 볼 수 있습니다. 이 마진은 값이 추가되었는데, 위에서 여백을 더 주어
아래로 내려가게 했음을 알 수 있는 코드입니다.

그렇다면, 빈칸에 관련된 코드는 무엇인지 알 수 있어야 하는데, 아래 부분에서 참고하여 작성할 수 있을 것 같아요! 아이콘 같은 경우에는, 텍스트(글자)도 아니기에 다른 view를 사용해 주어야 하는데요, 이미지를 선언해 준 것이기 때문에 ImageView를 빈칸에 입력해주어야 합니다

이제 다음 줄로 넘어가서, 어떤 코드가 작성되었는지 살펴보겠습니다.

<TextView
이번에는 텍스트를 지정해주는 TextView를 사용 하였습니다

android:id="@+id/tempMain“
여기에서도 id를 지정 해주었는데, 이는 JAVA 코드에서 활용할 수 있도록
선언하기 위해 지정한 것이라 생각하시면 되겠습니다.

android:layout_width="wrap_content"
android:layout_height="wrap_content“
width(가로)와 height(세로)를 wrap_content로 설정 해주었습니다.

android:layout_marginTop="20dp“
margin(마진)을 위로 주어서, 20dp를 여백으로 남기도록 하였습니다.
아까 전, 대한민국이라는 텍스트가 5dp였으니, 더 많은 여백을 준 것이죠.

android:layout_gravity="center"
android:text="--°"
android:textSize="70sp"
android:fontFamily="@font/notosanscjkkrbold"
android:includeFontPadding="false"
android:textColor="#ffffff"
/>
이 부분은 많이 중복되었으니, 여러분들도 무엇인지 알 수 있을 것 같습니다
한 줄 한 줄 무엇인지 여러분들이 해석 해볼까요? *(질문)*

답변 : 중앙으로 레이아웃을 지정했고, text는 --°로, text의 크기는 70sp이며
폰트와 폰트가 여백을 가질 것인지 설정해주었고, text의 색상을 설정 해주었습니다.

< 빈칸 세 번째 : 현재 위치(지역)을 표시해주는 텍스트 >
첫 번째로 현재 위치(지역)을 표시해주는 곳입니다.