사용자 인터페이스 정보

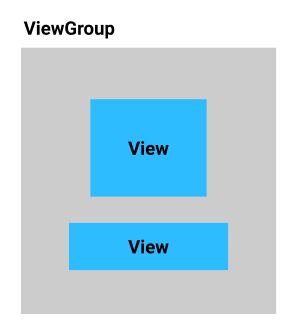
앱의 사용자 인터페이스(UI)는 화면에 표시되는 텍스트, 이미지, 버튼, 기타 여러 유형의 요소입니다. UI를 통해 앱은 사용자에게 콘텐츠를 표시하고 사용자는 앱과 상호작용합니다.

이러한 각 요소를 View라고 합니다. 앱 화면에 표시되는 대부분의 내용은 View입니다. Views는 클릭 가능한 버튼이나 수정 가능한 입력란처럼 상호작용할 수 있습니다.

이 Codelab에서는 텍스트를 표시하는 View 유형인 TextView를 사용해 봅니다.

Android 앱의 Views는 단순히 화면에 자체적으로 플로팅되지 않습니다. Views는 서로 관계가 있습니다. 예를 들어 이미지는 텍스트 옆에 있을 수 있고 버튼은 행을 형성할 수 있습니다. Views를 구성하려면 컨테이너에 배치합니다. ViewGroup은 View 객체가 있을 수 있는 컨테이너로, 내부에 있는 Views를 정렬하는 역할을 합니다. 정렬 또는 *레이아웃*은 앱이 실행되는 Android 기기 화면의 크기와 가로세로 비율에 따라 변경될 수 있고 레이아웃은 기기가 세로 모드인지 가로 모드인지에 따라 조정될 수 있습니다.

ViewGroup의 한 종류인 ConstraintLayout을 통해 내부 Views를 유연한 방식으로 정렬할 수 있습니다.



Layout Editor 정보

Views와 ViewGroups를 정렬하여 사용자 인터페이스를 만드는 것은 Android 앱을 만드는 주요 부분입니다. Android 스튜디오에서는 이 작업을 실행하는 데 도움이 되는 **Layout Editor**라는 도구를 제공합니다. **Layout Editor**를 사용하여 'Hello World!'를 'Happy Birthday!'로 변경하고 나중에 텍스트의 스타일을 지정해 봅니다.

**Layout Editor**를 열면 많은 부분으로 구성되어 있습니다. 이 Codelab에서 대부분의 사용법을 알아봅니다. 아래의 주석이 달린 스크린샷을 통해 **Layout Editor**의 창을 알아보세요. 앱을 변경하면서 각 부분에 관해 자세히 알아봅니다.

* 왼쪽 (1)은 이전에 봤던 **Project** 창입니다. 프로젝트를 구성하는 파일을 모두 나열합니다.
* 가운데에는 앱의 화면 레이아웃을 나타내는 2개의 그림 (4)와 (5)가 있습니다. 왼쪽 그림 (4)는 앱이 실행될 때 표시될 화면의 모습과 가깝습니다. 이를 **Design** 뷰라고 합니다.
* 오른쪽 그림 (5)는 **Blueprint** 뷰이고 특정 작업에 유용할 수 있습니다.
* **Palette** (2)에는 앱에 추가할 수 있는 다양한 유형의 Views 목록이 포함됩니다.
* **Component Tree** (3)은 화면 뷰의 다른 표현입니다. 화면의 모든 뷰가 나열됩니다.
* 맨 오른쪽 (6)은 **Attributes**입니다. View의 속성을 표시하고 변경할 수 있도록 합니다.

**Layout Editor**와 그 구성 방법에 관한 자세한 내용은 [개발자 참조 가이드](https://developer.android.com/studio/write/layout-editor)를 참고하세요.

주석이 달린 전체 **Layout Editor**의 스크린샷

텍스트, 스크린샷, 검은색이(가) 표시된 사진

자동 생성된 설명

* **Layout Editor**를 사용하면 Android 앱용 UI를 만들 수 있습니다.
* 앱 화면에 표시되는 대부분의 내용은 View입니다.
* TextView는 앱에서 텍스트를 표시하는 UI 요소입니다.
* ConstraintLayout은 다른 UI 요소의 컨테이너입니다.
* Views는 ConstraintLayout 내에서 가로와 세로로 제한되어야 합니다.
* View를 배치하는 한 가지 방법은 여백을 사용하는 것입니다.
* 여백을 통해 View가 컨테이너의 가장자리에서 떨어진 정도를 설정할 수 있습니다.
* TextView에 글꼴, 텍스트 크기, 색상과 같은 속성을 설정할 수 있습니다.