App Inventor 2
앱 인벤터 2

David Wolber, Hal Abelson, Ellen Spertus, Liz Looney 지음
오일석, 이진선 옮김

O'REILLY
한빛아카데미 Hanbit Academy, Inc.

추천사

연예와 오락, 교육 등의 다양한 분야에서 현대인이 누리는 소비문화는 사람들에게 다양한 기회를 제공한다. 다만 이러한 소비 활동은 대체로 수동적이다. 그저 쉬고 즐기기만 해도 좋지만, 그것이 전부는 아니다. 사람들은 소비의 매력에 푹 빠지기도 하지만, 무엇인가를 생산해 낼 때 창조의 기쁨을 훨씬 많이 느낀다. 그림을 그리거나 모형 비행기를 만들거나 특별한 사람을 위해 빵을 구울 때 느끼는 즐거움과 자부심은 소중하다.

오늘날 정보를 소비할 때 주로 쓰는 스마트폰, 태블릿 컴퓨터, TV 등의 첨단 제품은 보통 사람들에게는 그저 블랙박스이다. 박스 안에서 벌어지는 일을 이해하기란 보통 사람들로서는 불가능에 가깝다. 화면 위에 손가락을 대고 그림을 그린다거나 비디오 촬영을 할 수는 있지만 그런 행위 자체를 창조적이라고 보기는 어렵다. 이 책의 관점에서 말하자면, 이들 장치에서 실행되는 앱을 스스로 제작할 수 있는 사람은 극소수이다.

이런 상황을 바꿀 수 있다면 어떨까? 이들 장치를 제어하는 프로그램을 스스로 만들어볼 수 있을까? 앱을 만드는 일이 그림을 그리거나 빵을 굽는 일처럼 쉽게 접근할 수 있다면 얼마나 신날까? 그렇게만 된다면 소비문화의 최전방에 있는 이들 장치가 창조적인 삶에 좀 더 가까이 다가갈 수 있을 것이다.

앱 인벤터 세상에서는 이들 첨단제품은 더 이상 그저 신비로운 장치가 아니다. 내부 동작원리를 전혀 알 수 없던 블랙박스가 만지작거리며 무엇인가를 만들어볼 수 있는 장치로 바뀐다. 그러면서 수동적인 소비자 입장에서 벗어나 창조적인 생산자로 거듭난다. 실제로 이런저런 앱을 만들어보면서, 보다 깊이가 있고 새로운 방식으로 이들 장치를 가지고 놀 수 있게 된다.

이 책의 저자 중 한 명인 할 아벨슨Hal Abelson이 앱 인벤터 아이디어를 처음 말했을 당시, 스마트폰이 교육에 미칠 수 있는 독특한 힘에 대해서도 이야기를 나누었다. 할은 이 힘을 활용하면 컴퓨터 과학의 중요 개념을 학생들에게 효과적으로 가르칠 수 있을 것이라고 생각했다. 또한 앱 인벤터 개발이 끝나고 이 책의 또 다른 저자인 데이빗 올버 교수가 자신의 강의에서 앱 인벤터를 가르쳐본 결과, 기대를 훨씬 뛰어넘는 강력한 힘이 있다는 사실을 깨달았다. 한 마디로 앱 인벤터는 학생들을 소비자에서 창조적 생산자로 바꾸어 놓았다. 대부분의 학생들은 자신의 폰에서 동작하는 앱을 직접 제작해 보는 데 재미를 붙였고, 창의력을 한껏 발휘했다. 수강생 중 한 명이 간단하지만 매우 유용한 〈운전 중 문자 금지〉 앱을 만들었는데, 전문 소프트웨어 개발자가 아닌 누구라도 앱을 만들 수 있다면 무슨 일이 벌어질지 보여주는 좋은 사례이다.

한편, 구글은 앱 인벤터를 보다 쉽고 재미있고 강력하게 만드는 작업을 꾸준히 해왔다. 2012년부터 MIT의 할 아벨슨과 그의 팀이 대신 맡아 개선 작업을 계속 해오고 있다. 이 책이 소개하는 새로운 버전인 앱 인벤터 2를

이용하면 프로그래밍 경험이 전혀 없는 사람도 웹 브라우저에서 몇 분 안에 앱을 만들 수 있다.

이 책의 저자들은 모두 진정한 세계 수준의 교육자이자 소프트웨어 공학자이다. 앱 인벤터를 꾸준히 개선해 가는 이들의 노력에 감사드린다. 또한 멋진 책에 진정으로 감사드린다.

이제 당신의 창의성을 발휘할 때다. 앱을 만들어보자!

마크 프리드먼 Mark Friedman (구글 안드로이드 프로젝트 앱 인벤터 부서 기술 책임자)

지은이의 말

동네 산책길에서 조깅을 하다가 기발한 앱 아이디어가 머리를 스쳤다. 집에 돌아오는 내내 아이디어를 어떻게 실현할지 그 생각으로 머릿속이 가득 찼다. 하지만 어쩐다지? 프로그래머가 아닌데, 프로그래밍을 공부해 구현하려면 몇 년이 걸릴 테고 시간은 돈인데, 음, 그 사이 누가 만들어버릴 수도 있고……. 이렇게 아이디어는 묻혀 버린다.

몇 년 동안 프로그래밍 경험을 쌓아야만 앱을 만들 수 있는 세상을 넘어, 예술가, 과학자, 시민 운동가, 의료 종사자, 변호사, 소방관, 마라톤 선수, 축구 코치 등 누구나 손쉽게 앱을 만들 수 있는 다른 세상을 그려보자. 프로그래머를 고용하지 않아도 아이디어를 프로토타입으로 구현할 수 있는 세상, 내 취향대로 필요한 기능만 딱 들어간 맞춤식 앱을 만들 수 있다고 상상해 보자.

앱 인벤터가 바로 그런 세상이다. 앱 인벤터에서는 비주얼 프로그래밍 방식으로 안드로이드 폰에서 동작하는 앱을 만든다. 레고 블록 조립하듯 코딩하므로 어린아이들도 앱을 만들 수 있으며, 앱 프로그래밍의 진입 장벽을 극적으로 낮춰준다. 내 모습 또는 내 친구 모습이 캐릭터로 등장하는 비디오 게임을 만들어볼까? 오후 세 시가 넘어 식료품 가게 근처를 지나가면 자동으로 "우유를 구입해야죠."라고 말해주는 앱을 만들어볼까? 넌지시 이성 친구에게 프로포즈하는 퀴즈 앱을 만들어볼까? "문제 4: 나랑 결혼할래요? 수락 버튼을 누르면 문자가 자동 전송됩니다."라는 퀴즈 앱을 만든 사람이 실제로 있었는데, 그의 여자 친구는 수락 버튼을 눌렀다.

앱 인벤터는 누구나 무료로 사용할 수 있다. 웹 브라우저로 접근하여 개발하며, 프로그램 파일은 클라우드 방식으로 구글 서버에 저장된다. 또한 안드로이드 폰이 없어도 사용할 수 있도록, 안드로이드 폰을 그대로 흉내 내는 에뮬레이터를 제공한다. 2014년 9월을 기준으로 195개국에서 190만 명이 사용하고 있으며, 이들이 개발한 앱은 총 500만 개에 달한다.

그렇다면 이들 앱 개발자는 누구인가? 앱 인벤터를 배우기 시작할 때 이미 프로그래밍 경험이 있었을까? 몇몇은 그렇지만 대부분은 그렇지 않다.

이 책의 저자 중 한 명인 데이빗 올버David Wolber 교수가 샌프란시스코 대학교에서 개설한 과목에서 벌어진 이야기는 유명하다. 이 대학에서는 주로 경영학부와 인문학부 학생을 대상으로 교양 컴퓨터 과목에서 앱 인벤터를 가르친다. 수학 과목으로 인정해주기 때문에 주로 수학을 싫어하는 학생들이 이 과목을 수강한다. 수강생 대부분이 컴퓨터 프로그래밍을 해보리라고는 꿈에도 생각하지 못했던 학생들이다.

프로그래밍 경험이 전혀 없음에도 불구하고, 학생들은 앱 인벤터를 아주 재미있게 배우며 훌륭한 앱을 만들

어 낸다. 영문학을 전공하는 학생은 〈운전 중 문자 금지〉 앱을, 신문 방송학을 전공하는 두 학생은 〈내 차를 찾아줘〉 앱을, 국제학을 전공하는 학생은 〈방송 허브〉 앱을 만들었다. 언젠가 근무 시간이 훨씬 지난 늦은 밤에 예술학과에 다니는 한 학생이 올버 교수의 연구실에 찾아와 while 문에 대해 물어보았는데, 이것을 본 올버 교수는 앱 인벤터가 컴퓨터 교육을 크게 변화시키고 있다는 사실을 깨달았다.

대중 매체도 앱 인벤터의 영향력을 직감하였다. 〈뉴욕 타임스〉는 앱 인벤터를 "DIY 앱 제작 소프트웨어"라 불렀고, 〈샌프란시스코 크로니클〉은 "구글이 대중에게 앱 제작 도구를 선물했다."라는 기사에서 샌프란시스코 대학교 학생들의 앱 인벤터 교육 사례에 대해 썼다. 〈와이어드〉 지는 〈운전 중 문자 금지〉 앱을 만든 대니얼 피네건을 특집으로 다루면서 "피네건의 이야기는 중요한 점을 말해준다: 지금 컴퓨터 프로그래밍이 민주화되고 있다."라고 언급했다.

앱 인벤터로 통하는 문은 이미 활짝 열렸다. 전 세계의 중학교와 고등학교에서 널리 쓰이고 있다. 몇 가지 예를 들자면, 테크노베이션 챌린지(http://www.technovationchallenge.org)에는 28개국에서 2,500여 명의 소녀가 참가했다. 고등학교 방과 후 교실을 비롯해 고등학교 AP[Advance Placement] 프로그램에 새로 개설된 "컴퓨터 과학 원리" 과목에서도 가르치며(http://mobile-csp.org), 여러 대학교의 초급 컴퓨터 과학 과목에서 다룬다. 지금 이 순간에도 수천 명의 애호가, 직장인, 청혼을 준비하는 사람 등이 앱 인벤터 사이트에 접속하여 코딩하고 있다. 당신도 함께 참여해보고 싶지 않은가? 프로그래밍 경험이 없다고? 걱정하지 말고 시작해 보라.

이제 당신의 창의성을 발휘할 때다. 앱을 만들어보자!

저자 일동

요즘 스마트폰의 순기능과 역기능에 대한 논쟁이 한창이다. 공해라는 역기능 때문에 마차를 타고 다닐 수 없듯이, 스마트폰은 거스르기 어려운 대세가 되었다. 이런 상황에서는 순기능을 강화하여 역기능을 자연스럽게 줄이려는 태도가 현명할 것이다. 스마트폰은 최신 정보 기술의 총체로서 창의성을 발휘하기에 그만한 것이 없으며, 학교 현장에서 교육을 혁신할 수 있는 잠재력을 지니고 있다.

역자들은 몇 년 전부터 대학교 교양 과목에서 앱 인벤터를 가르치고 있다. 인문대, 사회대, 경영대, 자연대, 농대, 공대 학생들이 골고루 섞여 수업을 듣는다. 수업이 거듭되면서, 다른 과목에 비해 확연하게 학생들의 집중도와 성취도가 높다는 사실을 깨달았다. 게다가 프로젝트나 과제를 내주면, 각자 재미있는 아이디어를 생각하고 발전시켜 앱으로 만들어 낸다. 창의성이 발휘되고 길러지는 것이다. 그러면서 그들의 스마트폰 사랑도 두 배로 커지고, 커진 사랑만큼 새로운 아이디어가 샘솟고 앱 인벤터를 이용하여 실제로 앱을 만들어 본다. 멋진 선순환이다! 역자들은 앞으로 우리나라 초중고에 적용하고 확산해 보려는 계획을 가지고 있으며, 대학에서와 비슷한 현상이 나타날 것이라 확신한다.

이 책의 영어 원서는 구글과 MIT에서 앱 인벤터를 직접 개발하고 여러 대학에서 앱 인벤터를 가르쳐온 선구적인 교수들이 집필하였는데, 앱 인벤터의 바이블로서 손색이 없다. 또한 역자들이 그간 강의에서 사용해온 책이기도 하다. 이 책은 크게 두 부분으로 나뉜다. 앞부분은 13개의 재미있는 프로젝트로 구성되고, 뒷부분은 여느 프로그래밍 언어 책과 같이 언어의 기능을 하나하나 설명한다. 프로젝트는 따라 하기 쉽게 구성되어 있으며 곳곳에 컴퓨터 공학의 중요 원리와 프로그래밍할 때 꼭 지켜야하는 중요한 교훈이 스며있다. 역자들은 이런 특성을 이 책의 강점으로 보았고, 번역하기로 마음먹은 중요한 동기가 되었다.

한편, 앱 인벤터는 진화하고 있다. 이전 버전의 앱 인벤터를 설명한 1판이 2011년 4월에 출간되었고, 앱 인벤터 2를 설명한 2판이 2014년 10월에 출간되었다. 이 책은 2판을 번역했는데, 2판이 나온 지 불과 1년이 지나지 않은 지금 앱 인벤터 2는 개선되어 2판에도 낡은 부분이 있다. 예를 들어, 자동으로 프로젝트 이름이 앱 이름으로 설정되었는데 얼마 전 App Name이라는 속성이 추가되어 앱 이름을 별도로 지정할 수 있게 되었다. 번역을 하면서 이러한 변경점들을 최대한 반영하려고 노력했다. 이런 상황이라고 "앞으로 또 바뀌면 어쩌지?"라고 걱정할 필요는 없다. 학생들을 가르치면서 알아낸 사실 중의 하나는, 학습이 어느 정도 깊어지면 새로운 기능을 스스로 알아서 즐길 수 있게 된다는 것이다.

현재 MIT의 앱 인벤터 개발팀은 앱 인벤터를 여러 나라 언어로 서비스하는 일을 진행하고 있다. 이미 중국어, 이탈리아어, 에스파냐어는 번역이 완료되어 서비스 중이다. 예를 들어 "if count > 2 then ..." 대신 "만

일 개수＞2라면 ...”처럼 코딩할 수 있는 것이다. 모든 컴포넌트 이름도 자국어로 서비스된다. 역자들이 한국어 번역을 시작하였는데 아직 완성 단계까지 가지는 못했다. 관심 있는 독자들의 참여를 바란다.

영어 원서의 저자들이 운영하는 앱 인벤터 사이트 두 곳은 앱 인벤터를 공부하는 데 크게 도움이 된다. 역자들이 운영하는 사이트에서도 도움을 받을 수 있다. 다음은 이들 사이트의 주소이다.

- 앱 인벤터 공식 사이트 http://appinventor.mit.edu
- 앱 인벤터 교육 사이트 http://appinventor.org
- 역자들이 운영하는 앱 인벤터 교육 사이트 http://appinventor.chonbuk.ac.kr

이 번역서가 우리나라 소프트웨어 교육과 우리 국민의 창의력 증진에 조금이나마 기여하기를 바라는 마음 간절하다. 더불어 앱 인벤터를 통해 사람들이 더욱 행복해지길 기대해 본다!

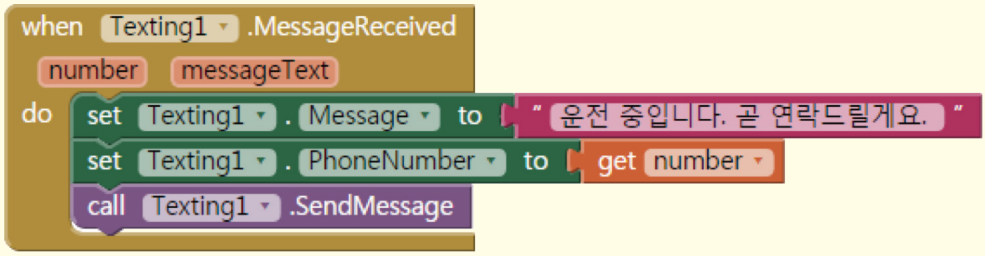
2015년 6월

역자 오일석, 이진선

안드로이드 폰을 위한 블록 언어, 앱 인벤터

앱 인벤터는 안드로이드 플랫폼에서 앱을 제작할 때 쓸 수 있는, 드래그-앤-드롭 방식의 비주얼 프로그래밍 언어이다. 웹 브라우저로 접속하여 GUI 방식으로 사용자 인터페이스(앱의 외양)를 설계하며, 블록을 끼워 맞춰가며 앱의 동작을 프로그래밍한다.

다음 그림은 프로그래밍을 전혀 해본 적이 없던 대니얼 피네건이라는 대학생이 만든 앱 프로그램의 초기 버전이다. 이 프로그램을 가만히 들여다본 다음, 무엇을 해주는 프로그램인지 말해볼 수 있겠는가?



이 앱은 문자 "응답 기계"라고 볼 수 있다. 예를 들어, 운전 중에 이 앱을 켜 놓으면 문자가 왔을 때 자동으로 응답 문자를 보내준다.

다른 프로그래밍 언어가 사용하는 문자 코드와 달리, 앱 인벤터는 블록 코드를 사용하기 때문에 훨씬 쉽게 현실 세계에서 필요한 일이 머릿속에 떠오른다. 예를 들어, 받은 문자를 크게 읽어주는 기능을 추가할 수 있을까? 응답 메시지를 각 상황에 맞게 맞춤식으로 입력할 수 있을까? 우리 학교 노래 경연대회에 쓸 수 있는 투표 앱을 만들 수 있을까? 이런 질문에 대한 답은 물론 "예"이다. 이 책은 이런 다양한 기능을 가진 앱을 어떻게 만드는지 알려줄 것이다.

앱 인벤터로 무엇을 할 수 있나?

앱 인벤터로는 많은 것들을 할 수 있다. 그중 주요한 몇 가지를 살펴보면 다음과 같다.

놀기

폰을 가지고 이런저런 앱을 만들면서 놀아보자. 그러다 보면 새로운 것을 탐구하고 발견하는 재미에 푹

빠지게 될 것이다. 웹 브라우저를 열고 앱 인벤터에 접속한 다음, 폰을 연결하고 위의 대니얼 피네건의 문자 응답 프로그래밍 같이 블록을 조립해보자. 즉시 폰에서 앱을 동작시켜볼 수 있다. 프로그래밍하고 있지만 일을 한다기보다는 폰을 가지고 노는 셈이다. 게다가 앱을 테스트해 보려면 친구에게 문자를 보내달라고 부탁해야 하므로, 놀이에 친구를 끌어들이는 셈이 된다. 이런 앱뿐만 아니다. LEGO NXT 로봇을 제어하는 앱을 만들며 놀 수도 있고, 위치 센서를 이용하는 앱을 만들면 폰을 들고 야외로 나가 앱을 테스트하면서 놀 수도 있다.

프로토타입 제작

새로운 앱을 만들 아이디어가 떠올랐다고? 머릿속에서 굴리다가 잊어버리거나 급한 김에 옆에 있는 냅킨에 끼적이지 말고, 앱 인벤터로 프로토타입을 만들어본다. 프로토타입이란 아이디어를 완벽하거나 세련되게 정돈된 수준은 아니지만 동작하는 정도까지는 구현한 모형이다. 아이디어를 문장으로 표현하는 일을 친구나 애인에게 긴 편지를 쓰는 것에 비유한다면, 앱 인벤터로 프로토타입을 제작하는 일은 벤처 투자자에게 보낼 시를 쓰는 것과 같다. 앱 인벤터는 전자 냅킨 구실을 해준다.

맞춤식 앱 만들기

사실 대부분의 사람이 주어진 앱에 갇혀있다고 말할 수 있다. 앱을 사용하다 보면, 쓸데없는 기능이 너무 많은데 정작 필요한 기능은 없어 불평하는 일이 다반사다. 나에게 딱 맞는 앱은 없을까? 다행이도 앱 인벤터로 나에게 맞춘 앱을 손쉽게 만들어 쓸 수 있다. 3장에서는 화면에 불쑥불쑥 튀어나오는 두더지를 손가락으로 때려 점수를 얻는 〈두더지 잡기〉 게임 앱을 만들어 보는데, 거기서 쓰인 두더지 캐릭터를 평소 미운 마음이 들었던 오빠나 언니 사진으로 바꾸고 실컷 때려줄 수 있다. 8장에서는 미국 대통령을 맞추는 퀴즈 앱을 만드는데, 여자 친구가 좋아하는 음악 장르나 우리 가족과 관련된 퀴즈로 쉽게 바꿀 수 있다.

인기 앱 제작

앱 인벤터는 프로토타입을 제작하거나 인터페이스를 설계하는 데 그치는 언어가 아니다. 좋은 아이디어만 있다면 완벽한 앱으로 제작한 다음, 플레이 스토어에 올려 인기 앱으로 올라설 수 있다. 앱 인벤터는 이런 앱을 제작하는 데 필요한 반복, 선택, 리스트, 프로시저 등과 같은 프로그래밍 언어의 기능을 블록 형태로 제공한다.

교육

앱 인벤터는 초등학교, 중학교, 고등학교, 대학교에 모두 적용할 수 있는 훌륭한 학습용 도구이다. 컴퓨터 공학 과목에서는 물론이고, 수학, 물리, 기업가 정신 등 많은 과목에서도 탁월한 도구로 활용할 수 있다. 직접 만들어 보면서 배운다는 점이 앱 인벤터를 활용한 교육의 핵심이다. 거리를 계산하는 공식을 외우는 고전적인 방식을 벗어나, 가장 가까운 쇼핑몰 또는 병원을 찾는 앱을 만들어 볼 수 있다. 혹

인 역사에 대해 에세이를 쓰는 대신, 마틴 루터 킹 목사와 말콤 엑스의 동영상과 연설 장면을 틀어주는 멀티미디어 퀴즈 앱을 만들어 보는 것도 좋다. 앱 인벤터와 앱 인벤터를 설명한 이 책은 다양한 과목에 걸쳐 선생님과 교수가 실제 강의실에서 활용할 수 있는 훌륭한 도구이다.

앱 인벤터는 왜 강력한가?

많은 사람이 앱 인벤터가 드래그-앤-드롭 방식의 비주얼 언어이기 때문에 프로그래밍하기 쉽다고 말한다. 그런데 이 말의 속뜻은 무엇일까? 앱 인벤터의 어떤 특성 때문에 그리 사용하기 쉬울까?

명령어를 기억하거나 입력할 필요가 없다.

초보 프로그래머가 처음으로 프로그램 코드를 입력하면 대부분의 경우 컴퓨터는 무슨 말인지 알아보기 힘든 오류 메시지를 쏟아낸다. 보통 이럴 때 많이 당황하고 좌절한다. 조금만 더 하다보면 흥미로운 프로그래밍 세계로 들어설 수 있음에도 불구하고 적지 않은 사람이 여기서 기가 꺾이고 포기한다. 앱 인벤터는 이런 고전적인 방식을 사용하지 않는다.

주어진 선택 사항에서 고른다.

앱 인벤터에서는 컴포넌트와 블록이 기능에 따라 구분되어 서랍에 들어있고, 언제든지 꺼내다 쓸 수 있다. 프로그래머는 필요한 기능을 생각한 다음, 그 기능을 해주는 블록을 찾아 끌어오면 된다. 명령어를 생각해내거나 생각나지 않을 때 매뉴얼을 뒤적거릴 필요가 없다.

끼울 수 있는 블록이 정해져 있다.

알아보기 힘든 오류 메시지를 쏟아내어 초보 프로그래머를 좌절시키는 고전적인 방식과 달리, 앱 인벤터는 처음부터 오류 가능성을 크게 줄여준다. 예를 들어, 어떤 프로시저 블록에 숫자가 입력되어야 한다면, 텍스트 블록은 들어맞지 않아 끼울 수 없다. 이런 방식이 오류 발생 가능성을 완벽히 없애진 못하지만, 크게 도움이 되는 것은 확실하다.

이벤트를 직접 처리한다.

대부분의 고전적인 언어는 명령어를 순서대로 처리하면서 프로그램을 실행하는 방식이 주류일 때, 즉 요리책에 적혀있는 명령어를 따라가면 요리가 완성되는 방식과 비슷할 때 개발되었다. 하지만 현대인이 사용하는 프로그램, 특히 모바일 앱 프로그램은 그래픽 인터페이스를 사용하기 때문에 사용자가 수시로 이벤트를 발생시킨다. 예를 들어, 문자를 받거나 전화가 오는 등의 이벤트가 수시로 발생한다. 이런 상황에서는 프로그램을 더 이상 요리책에 비유할 수 없고, 프로그램을 이벤트 처리기라는 관점으로 바라봐야 한다. 이벤트 처리기에서는 "이런 이벤트가 발생하면, 이런 작업을 수행하라"와 같이 프로그래밍 한다. 고전적인 언어 중 하나인 Java에서는, 단순한 이벤트를 처리할 때조차도 클래스와 객체, 리

스너라는 특별한 객체에 대해 이해하고 있어야만 프로그래밍이 가능하다. 하지만 앱 인벤터에서는 단순히 when이 붙어있는 블록을 끌어다 두면 "사용자가 버튼을 클릭하면..." 또는 "문자를 받으면..." 같은 이벤트를 처리할 수 있다.

어떤 앱을 만들 수 있나?

앱 인벤터로는 매우 다양한 종류의 앱을 만들 수 있다. 상상력을 펼쳐보자!

게임

보통은 3장의 〈두더지 잡기〉나 친구 얼굴에 낙서하는 2장의 〈페인트 통〉 같은 간단한 앱을 만들면서 앱 인벤터에 익숙해진다. 이후에는 자신의 스타일로 팩맨이나 스페이스 인베이더와 같은 복잡한 게임을 만들어 본다. 그런 다음, 5장의 〈무당벌레 추적〉처럼 폰의 기울음 정도를 감지하여 캐릭터를 이동시키는 게임으로 발전시켜볼 수 있다.

교육용 소프트웨어

앱 인벤터는 즐기기 위한 게임을 만드는 데 그치지 않고, 정보 제공용이나 교육용 앱을 만들 수 있다. 8장의 퀴즈 앱을 모방하여, 다음 주에 치를 국사 퀴즈에 대비하는 앱을 만들어 같은 반 학생들끼리 공유할 수 있다. 또는 10장처럼 사용자가 새로운 문제를 입력할 수 있는 퀴즈 앱을 만들 수도 있다.

위치 기반 앱

앱 인벤터는 GPS에서 위치를 읽어오는 블록을 제공하므로 현재 폰이 있는 곳에 따라 동작하는 앱을 만들 수 있다. 7장에서처럼 주차한 곳을 찾아가는 앱을 만들거나 넓은 쇼핑몰 또는 음악회장에서 친구나 가족을 찾는 데 쓰는 앱, 우리 학교의 캠퍼스를 소개하는 투어 앱을 만드는 것도 가능하다.

하이테크 앱

바코드 스캔을 비롯해 말하기, 듣기(음성 인식), 음악 연주, 음악 생성(9장 〈실로폰 앱〉), 비디오 재생, 폰의 기울음이나 가속도 알아내기, 사진 찍기, 전화 걸기 등의 고급 기능을 활용한 앱을 만들 수 있다. 스마트폰의 기술은 다용도의 대명사인 스위스아미 칼에 비유할 수 있는데, 앱 인벤터를 사용하면 이들 기술을 쉽게 다룰 수 있다.

SMS 문자 앱

4장의 〈운전 중 문자 금지〉 앱은 문자 기능을 활용하는 한 예에 불과하다. 사랑하는 사람에게 미리 설정해둔 시간만 되면 "보고 싶다."라는 문자를 자동으로 보내주는 앱을 만든다거나, 11장의 〈방송 허브〉 앱처럼 여러 사람의 모임을 주선하는 데 쓰는 앱도 만들 수 있다. "나는 가수다"와 같은 경연을 개최하

고, 최고 가수를 뽑기 위해 친구들이 투표하는 앱을 만들어볼까? 이러한 일도 앱 인벤터를 사용하면 가능하다.

로봇을 제어하는 앱

12장에서는 LEGO 로봇을 제어하는 앱을 만들어볼 것이다. 그 앱에서 스마트폰은 단순히 리모콘이 되는데, 스마트폰을 싣고 여기저기 돌아다니는 "두뇌"를 가진 로봇으로 변신할 수도 있다. 로봇과 스마트폰은 블루투스로 통신하며 다른 종류의 블루투스 장치를 다루는 앱도 얼마든지 만들 수 있다.

논리 흐름이 복잡한 앱

앱 인벤터는 프로그래밍 세상에 들어갈 때 넘어야하는 진입 장벽을 극적으로 낮춰준다. 앱 인벤터를 사용하면 섬광처럼 떠오른, 고급 기술을 써야하는 아이디어를 몇 시간 안에 만들어낼 수 있다. 하지만 앱 인벤터는 이런 종류의 앱을 만드는 데만 쓰이는 것은 아니다. 다른 프로그래밍 언어와 마찬가지로 복잡한 논리를 구현하는 데 필요한 반복, 선택, 논리 판단 기능을 제공한다. 앱 인벤터로 인공지능과 같은 복잡한 논리를 구현하다 보면 그 매력에 푹 빠질 것이다.

웹 기반 앱

웹과 대화하는 앱도 만들 수 있다. 트위터나 RSS 피드에 접속하여 데이터를 가져오거나, 아마존 온라인 서점에 접속한 다음 책에 찍혀있는 바코드를 스캔하여 최저 가격을 알아내는 앱을 만들 수도 있다.

Part 1

앱 인벤터 프로젝트

Chapter 1 안녕 야옹이

이제 여러분은 앱을 만드는 세상으로 들어왔다. 앱 인벤터의 중요한 구성 요소는 컴포넌트 디자이너Component Designer와 블록 에디터Blocks Editor이다. 먼저 두 요소를 중심으로 [그림 1-1]과 같은 첫 번째 앱 〈안녕 야옹이〉를 단계별로 만들어볼 것이다. 이 앱을 만들고 나면, 스스로 앱을 만들 수 있게 될 것이다.

그림 1-1 〈안녕 야옹이〉 앱

컴퓨터 언어를 배울 때에는 일반적으로 "Hello World"라는 텍스트를 화면에 출력하는 단순한 프로그램 작성부터 시작한다. 이 메시지가 성공적으로 나타나면 프로그램을 만들 수 있는 환경이 갖추어졌다고 본다. 이 전통은 벨 연구소에 근무하던 브라이언 커니건Brian Kernighan이 C 언어를 개발하던 1970년대로 거슬러 올라간다. 앱 인벤터에서는 가장 단순한 앱에서도 메시지 출력은 물론이고, 소리를 내거나 화면을 건드리면 반응하는 등 훨씬 재미있는 기능을 사용할 수 있다. [그림 1-1]은 이제부터 만들어볼 앱의 초기 화면으로, 고양이 얼굴에 손을 댄다거나 폰을 흔들면 '야옹~' 하는 울음소리가 나고 폰이 진동한다.

1. 무엇을 배우는가?

1장에서는 다음과 같은 주제를 다룬다.

- 필요한 컴포넌트를 선택하고, 컴포넌트가 할 일을 지정하여 앱을 만든다.
- 컴포넌트 디자이너에서 컴포넌트를 선택해 화면에 배치한다. 어떤 컴포넌트는 화면에 보이지만 어떤 것은 보이지 않는다.
- 컴퓨터에서 미디어(소리와 영상) 파일을 업로드하고 앱에 추가한다.
- 블록 에디터에서 블록을 조립한다. 블록은 컴포넌트가 할 일을 정의하는 요소이다.
- 앱 인벤터 라이브 테스팅을 통해 앱이 원하는 대로 실행되는지 확인한다. 제작하는 도중에도 수시로 동작을 확인할 수 있다.
- 앱을 폰에 다운로드하여 설치한다.

2. 앱 인벤터 개발 환경

이제부터 ai2.appinventor.mit.edu에 접속해 앱 인벤터 프로그래밍을 시작해 보자.[1] 그러면 2013년 12월부터 서비스하고 있는 새로운 버전의 앱 인벤터가 열린다. 이 새로운 버전을 앱 인벤터 2라 부르기도 하는데, 정식 명칭은 '앱 인벤터'이고 예전 버전을 '앱 인벤터 클래식'으로 구분하여 부른다. 이 책은 새로운 버전을 기반으로 한다.

앱 인벤터 프로그래밍 환경은 다음 세 가지 주요 요소로 구성된다.

- 컴포넌트 디자이너 : 앱을 제작할 때 쓸 컴포넌트를 가져오고 컴포넌트의 속성을 설정한다. 실행 화면은 [그림 1-2]와 같다.
- 블록 에디터 : 컴포넌트가 수행할 동작을 지정한다. 예를 들어 사용자가 버튼을 눌렀을 때 실행할 일을 코딩한다. 실행 화면은 [그림 1-3]과 같다.
- 안드로이드 폰 : 앱을 개발하는 도중에 프로그램이 제대로 작동하는지 검사하는 데 사용할 안드로이드 폰이 필요하다. 폰이 없는 경우에는 시스템이 제공하는 에뮬레이터를 쓴다.

1 현재 지원되는 브라우저와 버전은 Google Chrome 29, Safari 5, FireFox 23 버전 이상이다. 또한 앱 인벤터에 접속하려면 구글 계정이 필요하다.

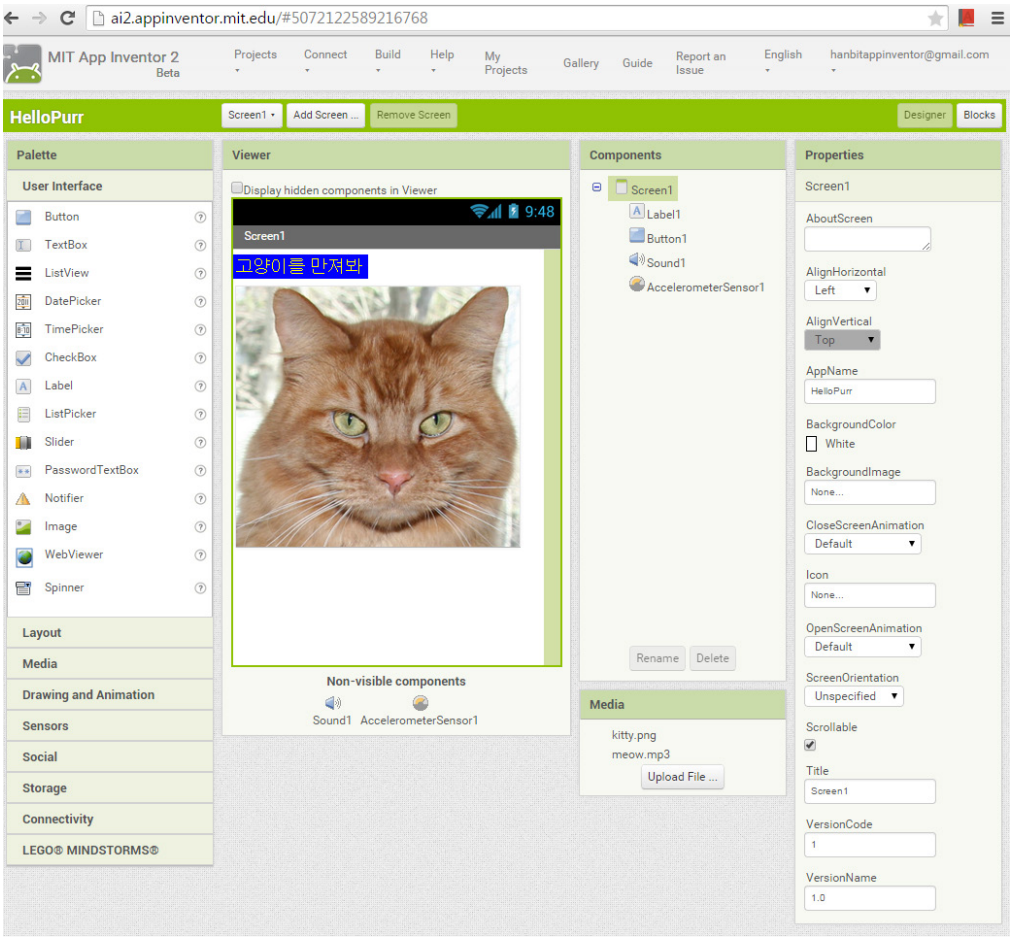


그림 1-2 앱 화면을 설계하는 데 사용하는 컴포넌트 디자이너

ai2.appinventor.mit.edu에 접속하면 Projects 화면이 나타난다. 프로젝트를 만든 적이 없으므로 비어 있을 것이다. 화면 왼쪽 상단의 [Projects]–[Start new project]를 클릭하여 새 프로젝트를 생성해 보자. 프로젝트 이름은 중간에 공백 없는 한 단어 형태로 지정해야 하는데, 여기서는 "HelloPurr"라 입력하고 [OK]를 클릭하자. 이렇게 하면 컴포넌트 디자이너가 나타나고, 화면의 오른쪽 상단에는 블록 에디터 화면으로 이동하는 [Blocks]가 보인다.

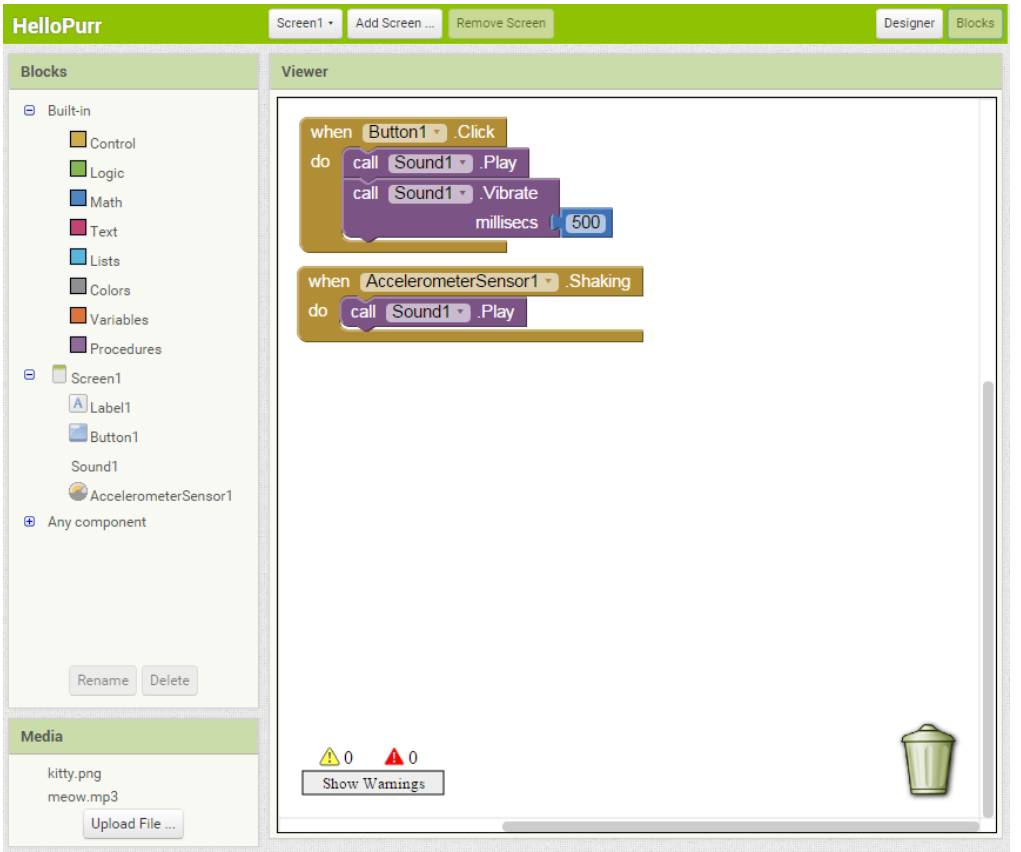


그림 1-3 앱이 수행할 동작을 지정하는 데 사용하는 블록 에디터

앱 인벤터는 클라우드 컴퓨팅 방식을 사용하는데, 이 방식은 작성된 프로그램을 사용자 컴퓨터가 아니라 온라인 서버에 저장한다. 따라서 앱 인벤터를 끝내고 나중에 다른 곳에 있는 컴퓨터에서 접속하더라도 기존에 했던 작업이 그대로 남아 있다. 워드프로세서를 쓸 때는 프로그램을 끝내기 전에 문서를 컴퓨터 디스크에 저장해야 하지만, 클라우드 방식에서는 그럴 필요가 없다.

3. 컴포넌트 설계

처음 사용할 도구는 컴포넌트 디자이너로, 줄여서 디자이너라고도 부른다. 컴포넌트는 앱을 제작하는 데 필요한 요소로, 서로 결합하여 앱을 만든다. 이는 요리책에 적힌 재료에 비유할 수 있다.

어떤 컴포넌트는 아주 단순하다. 예를 들어, Label 컴포넌트는 화면에 문자열을 표시해 주고,

Button 컴포넌트는 사용자가 컴포넌트를 누르면 지정된 동작을 실행한다. 반면 영상이나 애니메이션을 넣을 수 있는 Canvas 같은 복잡한 컴포넌트도 있다. 걷거나 흔들었을 때 움직임을 감지하는 AccelerometerSensor를 비롯하여 상대방에게 문자를 보내거나, 음악이나 비디오를 실행하거나, 웹사이트에서 정보를 가져오는 컴포넌트도 복잡한 축에 속한다.

디자이너를 열면, 컴포넌트를 배치하고 설정할 수 있는 [그림 1-4]와 같은 화면이 나타난다. 디자이너는 다음과 같은 영역으로 구성된다.

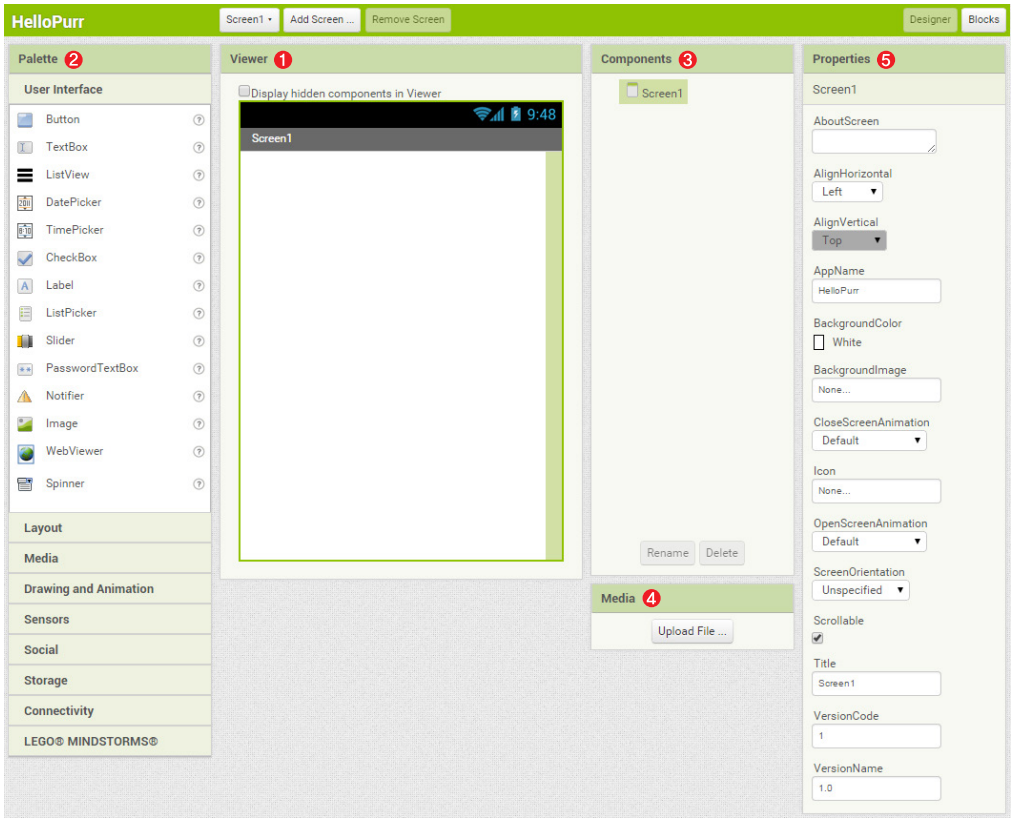


그림 1-4 앱 인벤터의 컴포넌트 디자이너

❶ **Viewer** : 이곳에 컴포넌트를 배치하여 앱의 화면을 설계한다. Viewer는 앱 화면을 개략적으로 보여준다. 예를 들어, 실제 앱에 표시된 문자열의 줄 바꿈 위치가 Viewer에서 본 것과 다르게 나타날 수 있다. 실제 화면을 보려면 폰이나 에뮬레이터를 써야 하는데, 이 과정은 잠시 후 배울 것이다.

❷ **Palette** : Viewer 왼쪽에 위치하며, Viewer에 배치할 수 있는 컴포넌트 목록을 보여준다.

Palette는 여러 구역으로 나뉘는데, [그림 1-4]는 User Interface 구역을 보여준다. 그 밑에 있는 Layout, Media 등의 구역을 보려면 그곳을 클릭한다.

❸ **Components** : Viewer 오른쪽에 위치하며, 프로젝트에서 현재 사용하고 있는 컴포넌트를 보여준다. [그림 1-4]에는 화면 전체를 뜻하는 Screen1이라는 컴포넌트만 있는데, 새로운 컴포넌트를 Viewer에 끌어다 두면 Components 영역에 이름이 나타난다.

❹ **Media** : Components 영역 아래에 위치하며, 프로젝트에 포함된 그림이나 소리와 같은 미디어 파일을 보여준다. 아직은 아무 미디어도 없지만, 곧 추가해 넣을 것이다.

❺ **Properties** : 화면의 맨 오른쪽에 위치하며, 컴포넌트의 속성을 보여준다. Viewer 영역에서 컴포넌트를 하나 클릭하면, 그 컴포넌트와 관련된 속성이 표시된다. 속성은 필요에 따라 값을 바꿀 수 있는 컴포넌트의 세부 내용이다. 예를 들어, Label 컴포넌트를 클릭하면 색상, 텍스트, 글꼴 등 여러 속성이 나타난다. [그림 1-4]에서는 Screen1의 속성으로 AlignHorizontal(수평 맞춤), AlignVertical(수직 맞춤), BackgroundColor(배경 색상), BackgroundImage(배경 영상), Title(제목) 등이 표시되어 있다.

이 장에서 만들 〈안녕 야옹이〉 앱은 두 개의 보이는 컴포넌트와 두 개의 보이지 않는 컴포넌트가 필요하다. "고양이를 만져봐"라고 표시할 Label 컴포넌트와 고양이 사진 모양의 Button 컴포넌트는 보이는 컴포넌트이고, "야옹"이라는 울음소리를 내줄 Sound 컴포넌트와 폰을 흔들었을 때를 감지해 줄 AccelerometerSensor 컴포넌트는 보이지 않는 컴포넌트이다. 새로 등장한 이들 컴포넌트를 모른다고 걱정할 필요는 전혀 없다. 지금 바로 배우게 될 테니까.

3.1. 레이블 만들기

제일 먼저 Label 컴포넌트를 추가해 보자.

1] Palette의 Label 컴포넌트를 클릭하여 Viewer로 끌어온다. 만약 User Interface 서랍이 닫혀 있다면 클릭하여 연다. Text for Label1이라고 적힌 직사각형 모양이 Viewer에 나타날 것이다.

2] Properties 영역으로 눈을 돌려보자. Label1의 속성이 나열되어 있는데 아래로 내려오다 보면 중간쯤에 Text 속성이 있다. Text for Label1이라 되어 있는 Text 속성을 "고양이를 만져봐"로 바꾸고 Enter를 누른다. 이제 Viewer 영역으로 눈을 돌려, 레이블이 "고양이를 만져봐"로 바뀌었는지 확인해 보자.

3] Label1의 BackgroundColor 속성의 값을 기본값인 None에서 Blue로 바꾸어 본다. 또한

TextColor 속성을 Yellow로, FontSize를 20으로 변경한다.

이제 화면이 [그림 1-5]와 같이 바뀌었을 것이다.

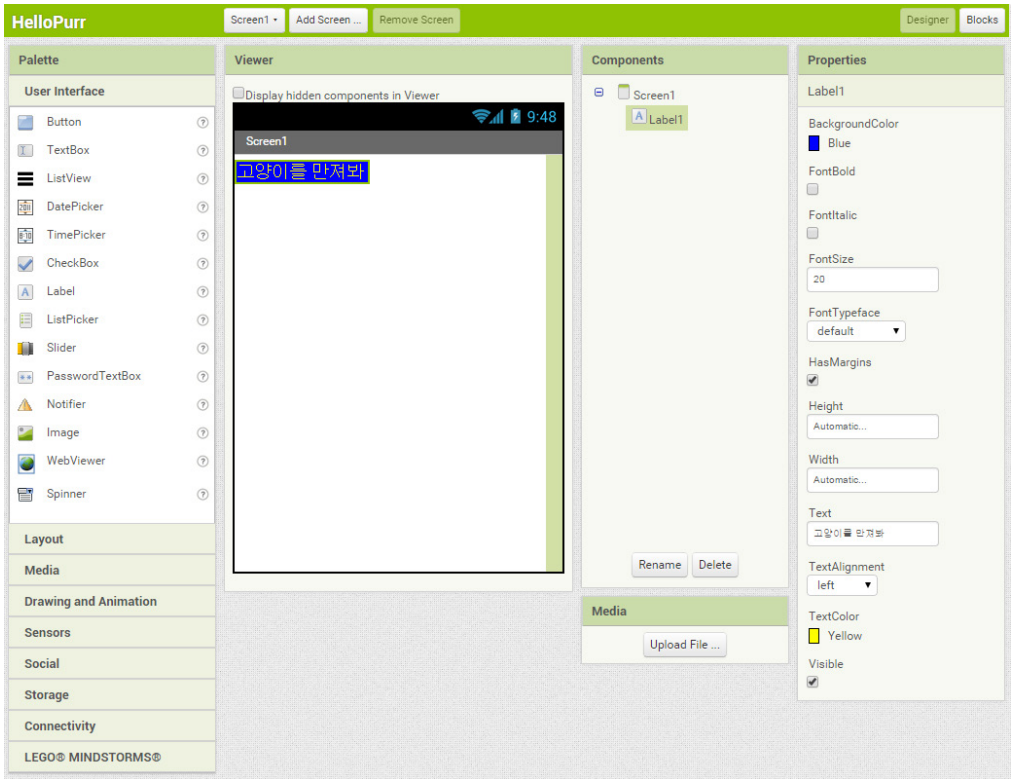


그림 1-5 Label 컴포넌트 추가

3.2 버튼 추가하기

〈안녕 야옹이〉는 화면에 있는 버튼을 누르면 고양이 울음소리가 나는 앱이다. 버튼은 Button 컴포넌트로 구현하고 버튼 배경에 고양이 사진을 넣어 재미를 더해 보자. Palette에서 Button 컴포넌트를 끌어다 Viewer 영역에 둔다. 이를 먼저 끌어다 둔 Label1 밑에 두어, "고양이를 만져봐"라는 글자 밑에 버튼이 나타나도록 하자.

나중에 이 버튼을 누르면 고양이 울음소리가 나도록 프로그래밍할 예정인데, 지금은 먼저 고양이 사진부터 넣어 보자. 다음 단계까지 진행한 화면은 [그림 1-6]과 같다.

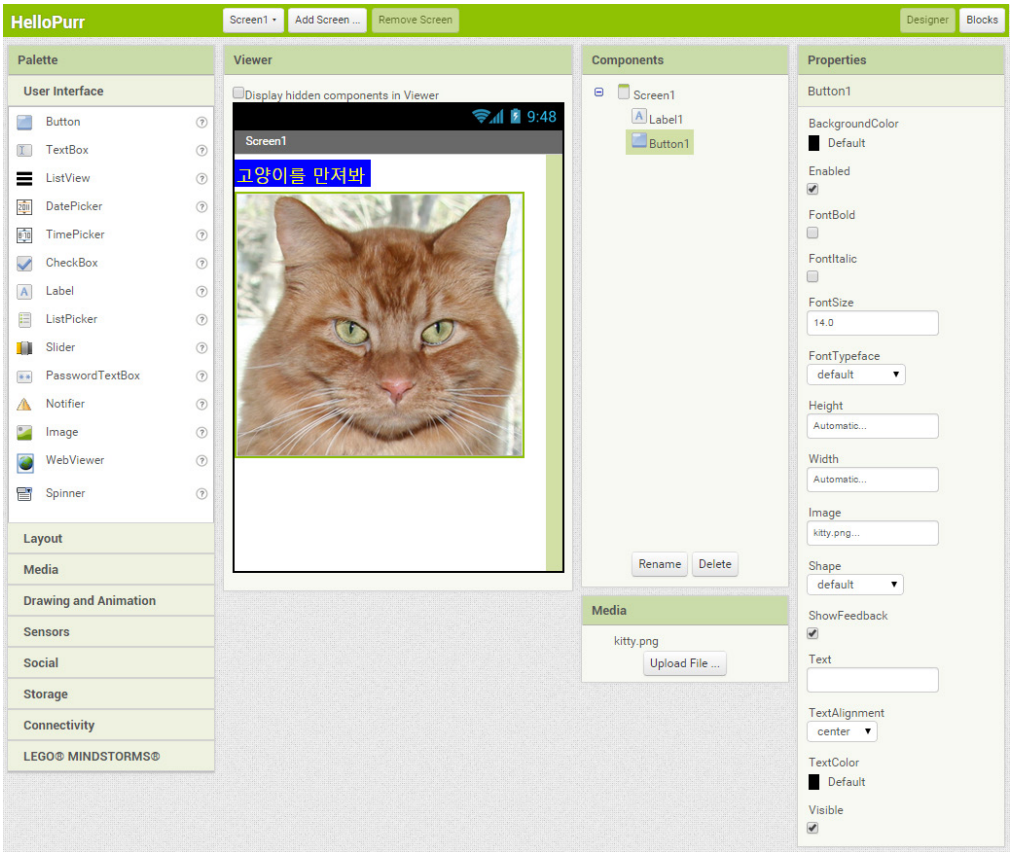


그림 1-6 Button 컴포넌트 추가

1] 먼저 고양이 사진을 준비해야 한다. 만약 마땅한 사진이 없다면, http://appinventor.org/bookFiles/HelloPurr/kitty.png에 접속하여 파일을 내려받는다. png는 jpg나 gif와 같이 영상 파일 확장자 중 하나이다. 소리 파일의 확장자로는 mpg나 mp3 등이 있다. 앱 인벤터는 이런 확장자를 가진 표준 파일을 모두 지원한다. 이참에 고양이 울음소리가 저장된 소리 파일도 http://appinventor.org/bookFiles/HelloPurr/meow.mp3에 접속하여 받아 두자.

Tip http://appinventor.org/bookFiles에 접속하면 1~13장에서 사용할 이미지와 소리 파일을 내려받을 수 있다.

2] Properties 영역에 Button1의 속성이 나타나 있을 것이다. 그렇지 않다면 Viewer에서 Button1을 클릭하여 선택한다. Image 속성을 찾아 클릭하면 기본값인 None으로 설정되어 있을 것이다.

3] [Upload File ...]을 클릭하여 [파일 선택]을 누른다. 1단계에서 내려받은 kitty.png를 찾아 선

택한 후 [열기] 버튼을 누른다.

4] 파일 업로드가 끝나면 kitty.png가 나타나는데, [OK]를 누르면 고양이 사진 넣기가 완료된다. Media 영역에 kitty.png가 있는지, 버튼에 고양이가 나타나는지를 확인한다.

5] 버튼에 여전히 "Text for Button1"이라는 텍스트가 덧씌워져 있는데, Button1의 Text 속성을 공백으로 바꾸면 없어진다.

3.3 고양이 울음소리 추가하기

사용자가 버튼을 클릭하면 고양이 울음소리가 나도록 만들어야 한다. 그러려면 앱 안에 소리를 추가하고, 사용자가 버튼을 클릭할 때 실행할 동작을 프로그래밍해야 한다.

1] 아직 소리 파일이 준비되지 않았다면, http://appinventor.org/bookFiles/HelloPurr/meow.mp3에 접속하여 파일을 내려받는다.

2] Palette에서 Media를 클릭하고 Sound 컴포넌트를 찾아 Viewer에 끌어온다. 어느 위치에 두던 Viewer 영역 바로 밑에 있는 Non-visible components 영역에 나타날 것이다. 이 영역에 나타나는 컴포넌트를 보이지 않는 컴포넌트라 부르는데, 이들은 화면에 나타나지 않은 채 자신이 맡은 역할을 수행한다.

3] Sound1 컴포넌트를 클릭하고 Properties 영역에서 Source 속성을 선택한다. 버튼의 Image 속성을 지정할 때와 같은 방법으로 Sound1 컴포넌트의 Source 속성을 미리 받아 놓은 meow.mp3 파일로 설정한다. 그러면 Media 영역에 kitty.png와 meow.mp3라는 두 개의 파일이 나타날 것이다.

[표 1-1]은 지금까지 〈안녕 야옹이〉 앱에서 사용된 컴포넌트를 정리한 것이다.

표 1-1 〈안녕 야옹이〉 앱에 추가한 컴포넌트

컴포넌트 유형	Palette 그룹	컴포넌트 이름	기능
Label	User Interface	Label1	"고양이를 만져봐"라는 텍스트를 보여준다.
Button	User Interface	Button1	누르면 고양이 울음소리가 난다.
Sound	Media	Sound1	고양이 울음소리를 낸다.

확장해 보기

지금까지 재미있는 앱을 하나 만들어 보았다. 그런데 앱을 다 만들고 나면 종종 앱을 개선하는 데 쓸 수 있는 아이디어가 여럿 떠오르곤 한다. 이 책은 장의 마무리 부분에 몇 가지 개선 아이디어가 제공된다. 이런 아이디어를 구현하다 보면 필요한 컴포넌트와 블록을 스스로 찾아 익히는 습관이 길러지고 결국 스스로 앱을 만들 수 있는 능력이 생길 것이다.

다음에 제시된 아이디어를 바탕으로 〈안녕 야옹이〉 앱을 개선해 보자.

- 폰을 흔들면 고양이 울음소리가 메아리처럼 이상하게 들릴 것이다. 폰이 올라갈 때와 내려갈 때 가속도 센서의 Shaking 이벤트가 발생하는데, 폰이 너무 빨리 움직여 이벤트가 초당 여러 번 발생해 울음소리가 중복되기 때문이다. 이럴 때는 Sound 컴포넌트의 MinimumInterval 속성을 사용하여 소리를 발생시키는 최소 간격(다음 소리를 발생시킬 때까지 최소 시간)을 조정해 준다. MinimumInterval 속성의 기본값은 500이다. 고양이 울음소리가 500밀리 초 이상이면 소리가 끝나기 전에 새로운 울음소리가 시작되므로, 앞뒤 소리가 중복되어 메아리가 들리게 된다. MinimumInterval 속성을 조절하여 메아리 현상을 없애 보자.
- 앱을 실행시킨 상태로 폰을 주머니에 넣고 빨리 걷다 보면, 주머니 속에서 고양이 울음소리가 나서 당황할 수도 있다. 일반적으로 안드로이드 앱은 사용자가 화면을 보면서 조작하지 않더라도 계속 동작하도록 설계되어 있다. 따라서 주머니 속에 있더라도 가속도 센서를 계속 감시하다가 폰이 흔들렸다고 판단되면 이벤트를 발생시킨다. 폰의 메뉴 버튼을 눌러 앱 실행을 완전히 중지시켜 보자.

요약

이번 장에서 배운 내용은 다음과 같다.

- 디자이너에서 컴포넌트를 끌어오고, 블록 에디터에서 언제, 무슨 일을 수행할지 지정한다.
- 컴포넌트에는 보이는 컴포넌트와 보이지 않는 컴포넌트가 있다. 보이는 컴포넌트는 앱의 사용자 인터페이스 화면에 나타난다. 보이지 않는 컴포넌트는 나타나지 않은 채 소리를 내는 것과 같은 효과를 제공한다.
- 블록 에디터에서 블록을 조립하여 컴포넌트가 할 일을 지정한다. 예를 들어 Button1.Click 이벤트 처리기 블록에 Sound.Play 명령어 블록을 끼운다. 사용자가 버튼을 클릭하면 Button1.Click에 꽂혀 있는 블록이 순서대로 실행된다.

- 어떤 명령어 블록은 추가적인 정보가 필요하다. 예를 들어 Vibrate 블록은 진동 시간을 지정해 주어야 한다. 이러한 값을 매개변수라 부른다.
- 숫자는 숫자 블록으로 만든다. 매개변수로 숫자가 필요한 명령어 블록에 숫자 블록을 끼워 넣는다.
- 앱 인벤터는 센서 컴포넌트를 가지고 있다. 예를 들어 AccelerometerSensor는 폰의 흔들림 또는 이동을 감지한다.
- 개발 완료한 앱을 폰에 다운로드하면 앱 인벤터와 연결되지 않아도 스스로 작동한다.