Engineering Tomorrow

Software Engineering

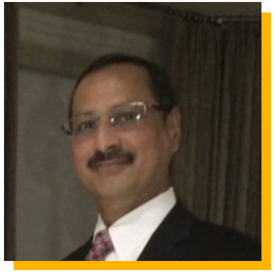
Building an App Using MIT App Inventor

GOAL

Students learn to build an app using MIT App Inventor.

2024-01-01_v2.0

Breakout Development Team



VIVEK BHARGAVA

College: Indian Institute of Tech, BS
Vanderbilt University, PhD
Major: Materials Science & Engineering
Industry Experience: General Electric, Regal Beloit Corporation

EMMA O'SHEA

College: Bucknell University
Major: Biomedical Engineering

LAUREN SAWYER

College: Penn State University
Major: Aerospace Engineering

GIGI BRUSCO

College: University of Notre Dame
Major: Computer Science & Engineering

Breakout Re-Development Team

GERALD DAWES

College: NYU Tandon School of Engineering
Major: Electrical Engineering
Industry Experience: 30+years of Leadership in Energy Systems, Business Ethics, Construction and Electrical Engineering

MARY JARRATT

College: Virginia Tech PhD Student
Major: Robotics and Engineering
Industry Experience: Computer Programer, Systems Engineer, Robotics Education

LINDSAY RONEY

College: University of Notre Dame
Major: Computer Science & Engineering

MARY BRUSCO

College: University of Notre Dame
Major: Computer Science & Engineering

PART 1

Background

What is an App?

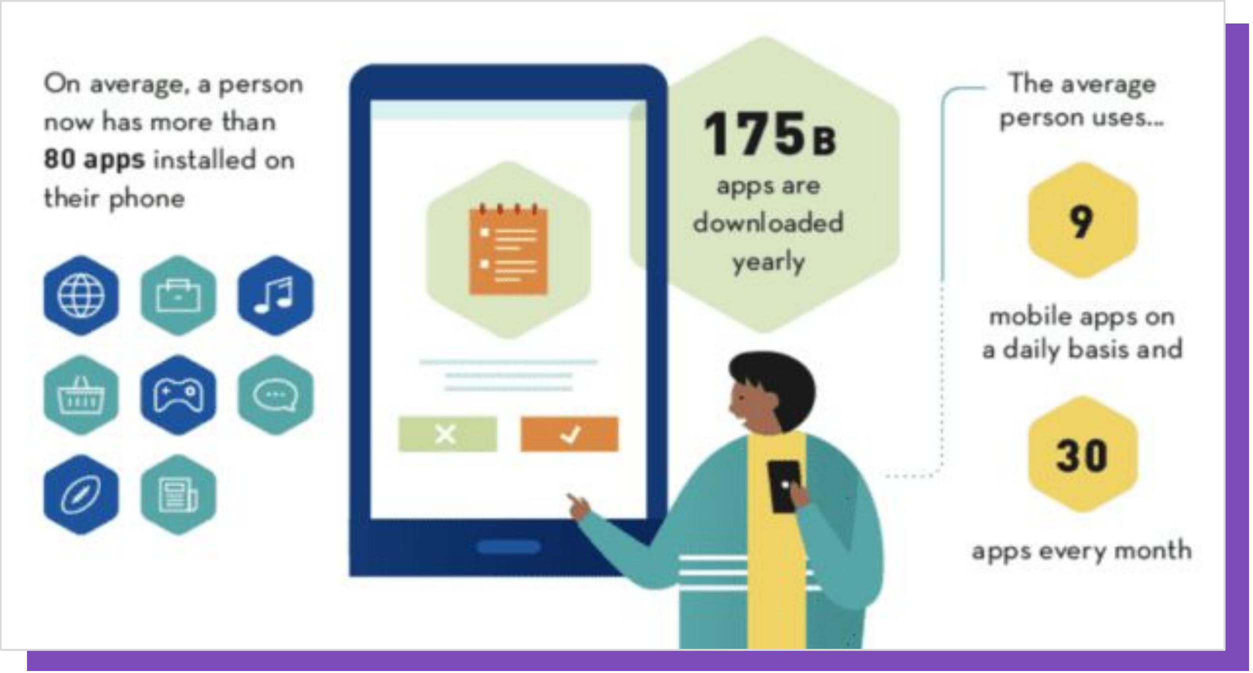
- An app is an application software designed to run on a mobile device, such as a smartphone, tablet, or computer.
- Apps place the power of software tools right at your fingertips but they may not have all of the same features.
- Apps are task-specific and can assist with performing personal, educational or business functions.

WHAT WERE THE TOP DOWNLOADED APPS IN 2022?

1. TikTok
2. Instagram
3. Facebook
4. WhatsApp
5. Shopee
6. Telegram
7. Snapchat
8. Messenger
9. CapCut
10. Spotify

The Statistics of App Downloads in 2022

HOW MANY APPS DO WE USE AND DOWNLOAD?



On average, a person now has more than **80 apps** installed on their phone

175B apps are downloaded yearly

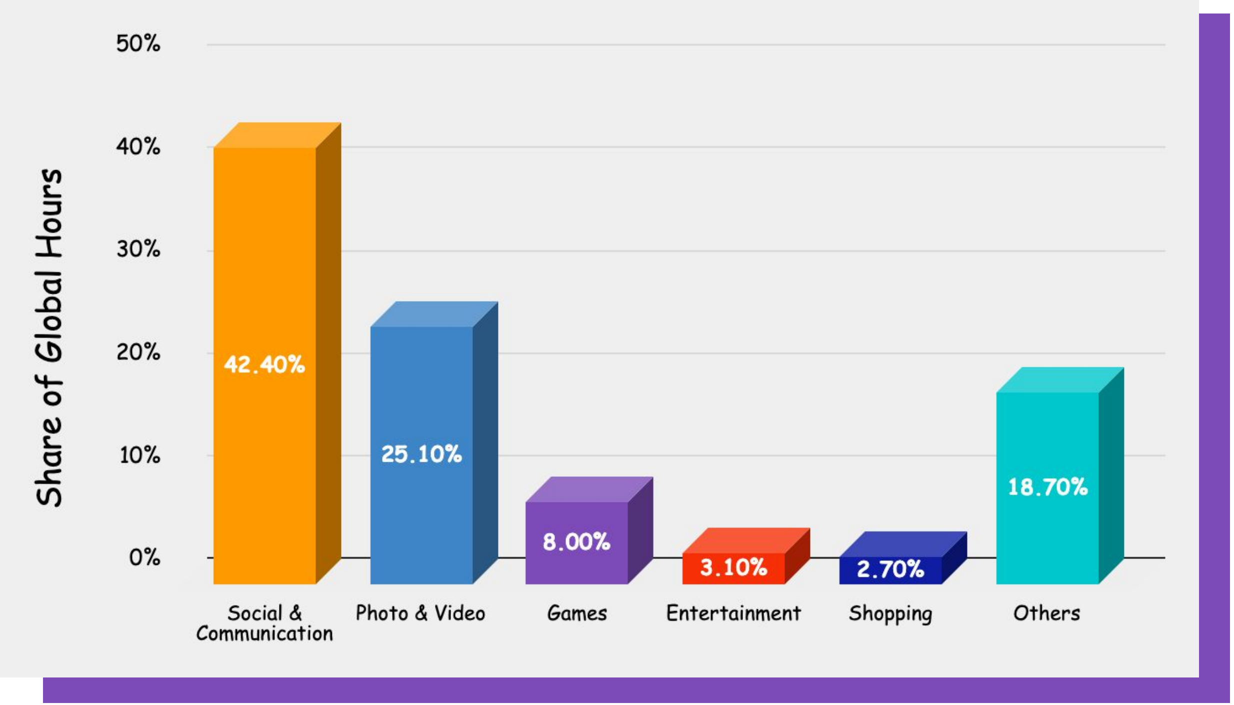
The average person uses...

9 mobile apps on a daily basis and

30 apps every month

Categories of App Usage in 2022

HOW DO WE USE APPS IN OUR DAILY LIVES?



Brief History of Apps

- **1994**
  IBM Smartphone preloaded with apps like Address Book and Calculator
- **1997**
  Nokia mobile phone includes arcade game “Snake”
- **2001**
  First iPod came with built-in games including Solitaire and Brick
- **2007**
  First iPhone released
- **2008**
  App store was launched
- **2010**
  App voted “Word of the Year”
- **2011**
  Uber launched
- **2012**
  Google releases all of its content under one platform: Google Play Store
- **2014**
  Applications are now downloaded to alternative devices like cars
- **2017**
  Mobile commerce has become a $30M industry and is continuing to grow

THIS VIDEO WILL TEACH YOU ABOUT THE HISTORY OF APPS.

PART 2

An Introduction to App Development

You are a Software Engineer

What do you need to do to develop an app? How would you start?

PROBLEM
People who can’t communicate because of distance.

LEARN
Research existing solutions and technical knowledge.

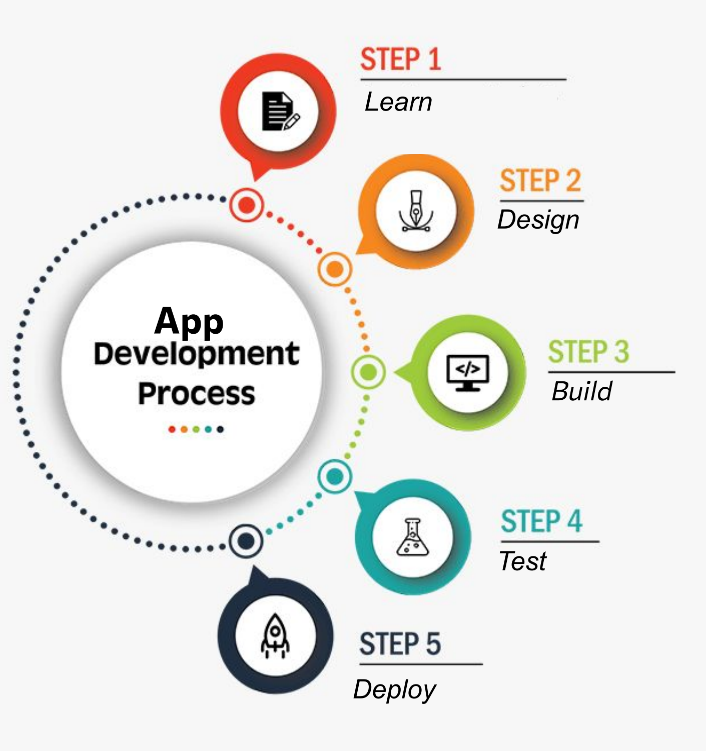
BRAINSTORM
Think which solutions are optimal?

BUILD + TEST SOLUTION
Code an app solution. For example, GMail.

The Engineering Design Process

Let's look at some specific technical components that make up this process as a software engineer.

- **Learn** → Research existing solutions and technical knowledge.
- **Design** → Utilize research and technical knowledge to create a prototype.
- **Build** → Code an application.
- **Test** → Have different users test the application. There will be several iterations of this step to find the most effective version.
- **Deploy** → The application can be released to the app store once user suggestions are integrated.



Click HERE for more information on the Engineering Design Process!

Front-end Design

THERE ARE TWO COMPONENTS TO APP DEVELOPMENT: FRONT AND BACK-END DESIGN.

- **Front-end Design** entails the graphic design and user interface components of an application.
- Everything the user sees and interacts with when they open an app is part of the **front-end Design**.
- Most of the technical work involved in the user interface is created using coding languages like HTML, CSS, and JavaScript.

Back-end Design

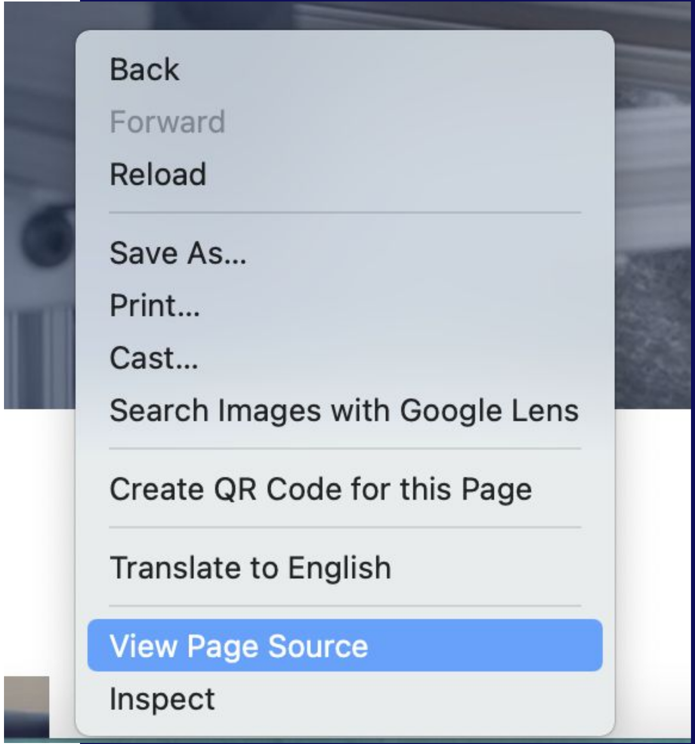
- **Back-end Design** includes the components of the app indirectly accessed by users through the front-end.

- For example, when the user clicks a button, the way that the application reacts in response to the user is all part of the **back-end design**.

- Think of the **back-end** as the invisible structure that helps an app function properly.

Front and Back-end Design Activity

- Click this link to get to Engineering Tomorrow's website in a **Chrome browser** (every browser has different inspection tools).
  - Here, we will explore components of the website's **front-end design**.
- Click (or right-click for PC) **View Page Source**. This shows the HTML or XML of the website you're viewing.
  - Browse through the code and find elements of the Engineering Tomorrow website within the code.



Front and Back-end Design Activity

- For example, this line: `">Get involved</a>"` is in the code and also a tab on Engineering Tomorrow's website.

- Next, click **Inspect**.
    - Here, you can view and *even* edit the source code. If you choose to edit the code, the changes will be reflected on your computer only, not on the website as a whole.
    - Notice the **Event Listeners** tab and based on observations, **what do you think an Event Listener is**?

App Terminology

- VISIBLE COMPONENTS
  are text boxes, buttons, labels, etc; all which can be seen by the user on the screen and are apart of the User Interface. (**front-end design**)

- NON VISIBLE COMPONENTS
  are Sound, Accelerometer, Orientation Sensor; all components which are not seen but provide built-in access to functions of the device. (**back-end design**)

- PROPERTY
  each **component** is characterized by various **properties** and are the state of each component (visible or non visible). For example, the state of a label could be "Hello World!" in purple text.

- VARIABLE
  is a storage container for information. This information could be a quantity, letter, word. Variables can also be global (accessible everywhere) or local (limited access).

App Terminology (continued)

- **BOOLEAN**
  is a type of variable that has two possible values: true or false. True is considered **1** and False is **0**.

- **EVENT**
  is an action that takes place when a user interacts with a program

- **EVENT LISTENER/HANDLER**
  dictates the action that follows the event. A programmer writes code for an action to take place when an **event** occurs.

- **PROCEDURE**
  a set of instructions

- **LIST**
  used to store information. When items are added to a list, they are placed in a certain position in the list. The position of an item in a list is often called its index.

PART 3

An Overview of Software Development

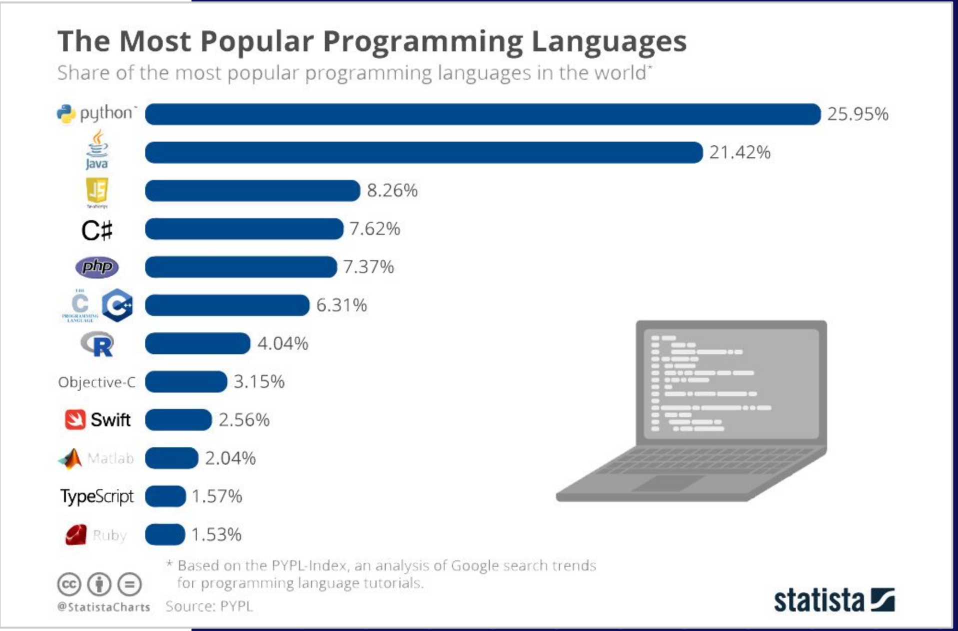
What is Software Development?

- **Software development** is the process of designing, creating, testing, and maintaining different software applications:
    - the development is achieved by writing computer code, also known as **coding**, which is a computer oriented process that uses reason to accomplish a goal
    - like reading, writing, and speaking, there are different languages that coding can be programmed in, Python being the most popular today

Programming Languages

Did you know there are over **700** different programming languages? Here are some of the most popular ones for coding apps:

Python is often favored because of its simplicity. It is known as a High-Level Programming Language.



The Most Popular Programming Languages

Share of the most popular programming languages in the world*

* Based on the PYPL-Index, an analysis of Google search trends for programming language tutorials.

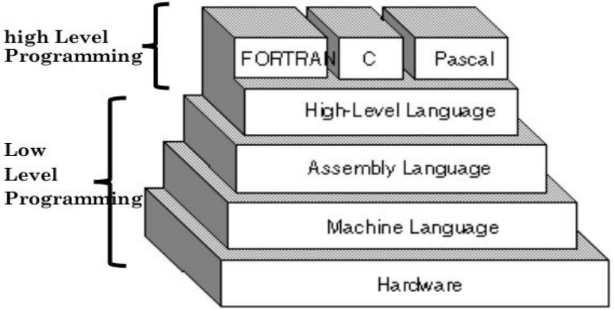
Source: PYPL

High vs. Low-Level Programming Languages

- **HIGH-LEVEL**
  A High-Level programming language has strong abstraction from the details of the computer.

  This means that many of the details you would have to account for in a Low-Level programming language are already built into the code.

  Generally, High-Level programming languages are more user friendly, such as block code as you will see in the next slide



- **LOW-LEVEL**
  Low-Level programming languages are languages written directly for the computer (i.e. Machine Code and Assembly Language).

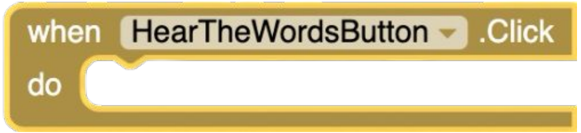
  Understanding Low-Level programming languages is important because you learn programming from the ground up. Knowing the most complicated base-level languages, teaches you what goes on under the hood.

  Then, when you're coding in high-level programming languages, you will have a better understanding on what is actually happening.

Different types of statements we will use

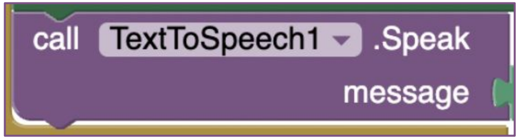
You will be working in block code (a High-Level Programming Language). This code is actually written in Python, but the block code allows for ease of learning.

Yellow → Control Blocks



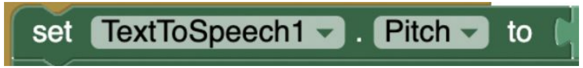
when HearTheWordsButton .Click
do

Purple → Procedure Blocks



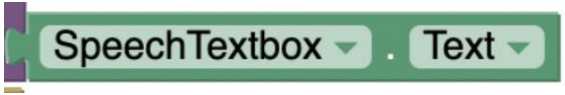
call TextToSpeech1 .Speak
message

Dark Green → Action Blocks



set TextToSpeech1 . Pitch to

Light Green → Description Blocks



SpeechTextbox . Text

Conditional Statements

What are conditional statements?

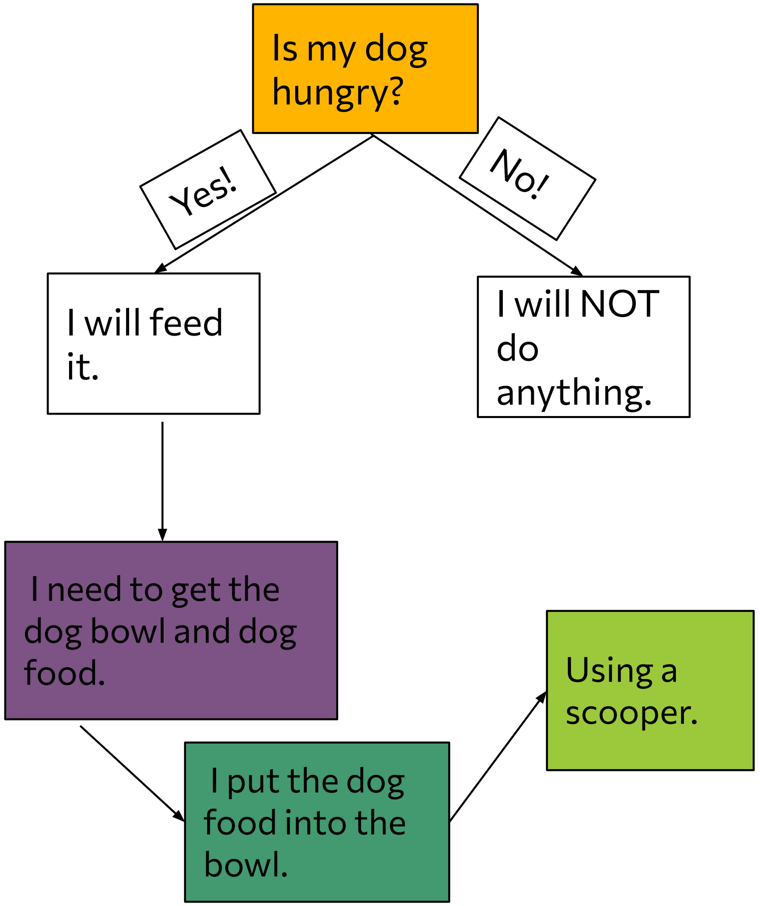
Think in terms of different instructions you should follow. Which direction you follow is based upon which conditions are met. If a condition is met, then you will complete a specified task.

For example (Written out Code):

Is my dog hungry?
When my dog is hungry, I will feed it I need to get the dog bowl and dog food. I put the dog food into the bowl using a scooper.
When my dog is NOT hungry, I will NOT do anything.

The yellow block is **controlling** the system by asking a question or condition. If the condition is met the system will think about the **procedure** of the purple block and output the **action** of the dark green block using the **description** of the light green block. Since there is no action when the dog is NOT hungry, usually there is no code written with it and its default is to not do anything unless the condition is met.

For example (Visual Code):



PART 4

Learning how to Build Your App

Modeling Real World Applications

TODAY'S CHALLENGE

Code a text-to-speech app in MIT App Inventor, essentially creating your very own **Deep Fake** app. We will further explore the topic of **Deep Fakes** later throughout the interactive workbook. Let's get started!

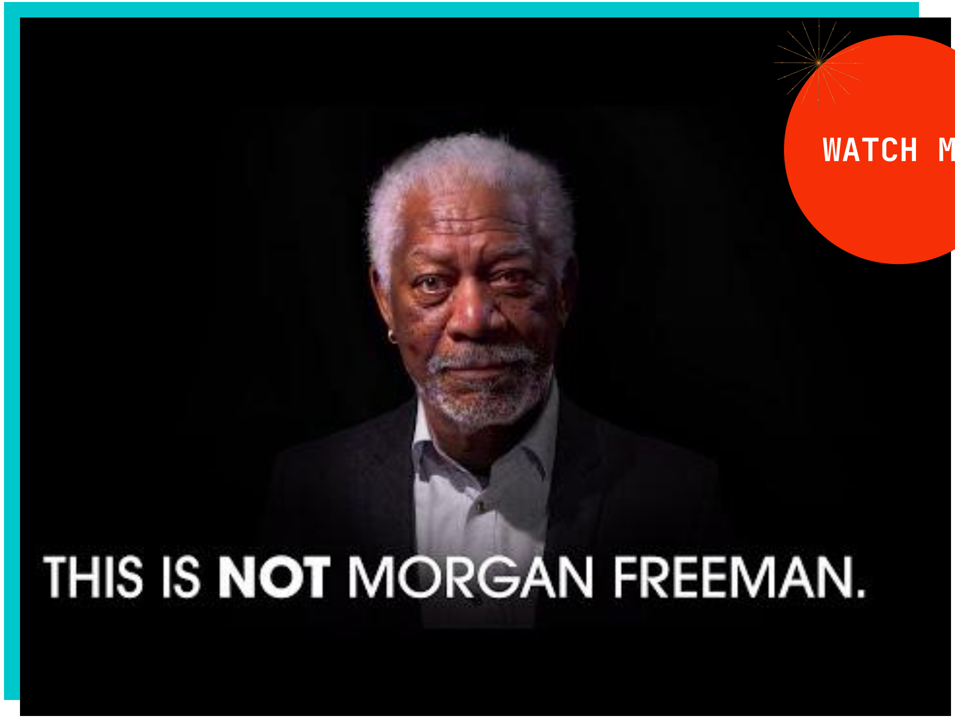
Wait, what is a Deep Fake?

- `DEEP FAKE`
  are video and audio files made using artificial intelligence

  - Powerful computers make **Deep Fakes** by learning from existing video and audio recordings of humans
  - The term, **Deep Fake**, comes from “deep learning”
  - Deep Fakes, when thought to be real, spread misinformation and use other’s fame to promote messages (these messages can be funny, political, serious, etc.)

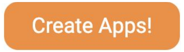
Wait, what is a Deep Fake?

Here is an example of a Deep Fake:



WATCH ME

Step 1: Getting Started with App Inventor

1. Go to https://appinventor.mit.edu/.
2. Click the "Create Apps!" button 

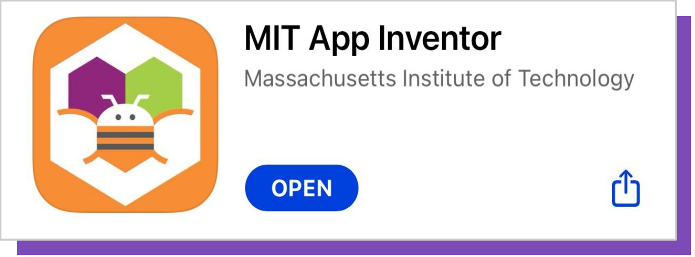
3. Sign in using your email account. When finished, you land on the projects page. A pop-up saying, "Get Started With Some Tutorials" appears. Click, "Close."

Reminder! Before you Build your App!

The cool thing about MIT App Inventor is that you can see your creations working like a real app on your phone. To do so, download MIT App Inventor on your iPhone or Android device. We will use this later in the tutorial.

NOTE

make sure your computer and phone are connected to the **same** wifi network. You will not be able to use your app on your phone if they are not both connected to the **same** network. For additional information, click this link. Additionally, if you do not have a phone available for use, complete the app tutorial independently and then, share a phone with a friend. You can both view your creations on one phone if you take turns.



Continue to Explore

▶ IF YOU LIKED TODAY’S BREAKOUT,
YOU MAY BE INTERESTED IN THESE TOPICS:

- Programming
- Digital Design
- Artificial Intelligence
- App Developer for Mac
- App Developer for Android
- Human-Centered Design

▶ TYPES OF ENGINEERING RELEVANT TO
TODAY’S SOFTWARE ENGINEERING BREAKOUT:

- Computer Science
- Software Engineering
- Computer Engineering
- Electrical Engineering

Thank you!

Engineering
Tomorrow