Exploring creativity through **mobile app development**

An introduction to learning to code, building games and finding a career

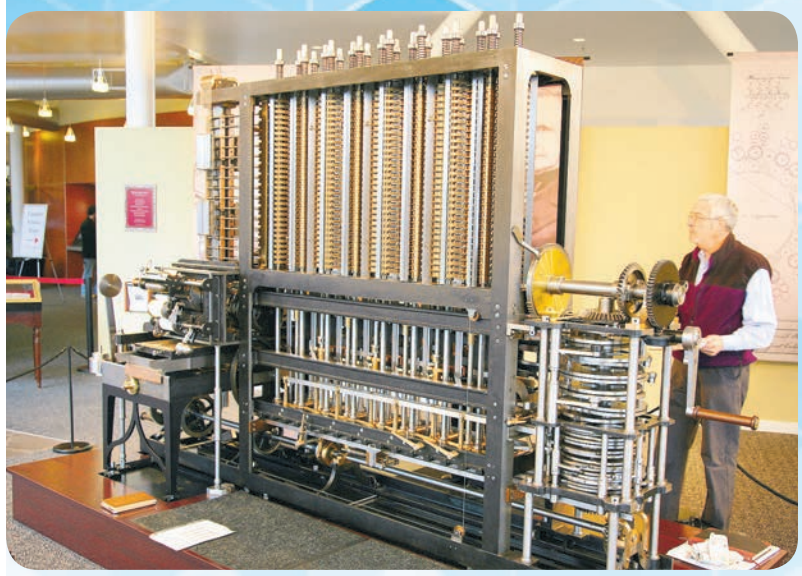
DevKit:
learn to code & make games

VYBE SOFTWARE

Tampa Bay Times NIE
newspaper in education
tampabay.com/nie

Download on the App Store

The computing revolution



Charles Babbage's Difference Engine calculating machine, invented in 1822

> Computing radically changes how humans solve problems, and even the kinds of problems we can imagine solving. Computing has changed the world more than any other invention of the past hundred years, and has come to pervade nearly all human endeavors. Yet, we are just at the beginning of the computing revolution."
>
> **— Dr. David Evans, *Introduction to Computing: Explorations in Language, Logic, and Machines***

In 1822, English mathematician Charles Babbage demonstrated a prototype of his Difference Engine calculating machine, and in 1833, he began work on his Analytical Engine, one of the first computational machines.

The 19th and early 20th centuries saw rapid advances in the field of computing. The world's first electronic computer was built in 1943; the world's first commercially available computer in 1951; and in 1969, the first message was sent over ARPANET, an experimental computer network that was the forerunner of the Internet.

In the 21st century, technology has come to influence almost every aspect of daily life and work, and the demand for skilled workers in science, technology, engineering and math (STEM) fields continues to grow.

Technology growth = jobs

The field of information technology is expected to grow by 12 percent between 2014 and 2024, adding almost half a million new jobs to our economy. Many of those jobs are right here in Florida!

According to the Alliance for Science & Technology Research in America, Florida will add more than 100,000 new jobs in STEM fields by 2024, with an average wage of almost $30 per hour.

According to the Florida Department of Economic Opportunity, six of the 10 occupations with the highest entry-level wages and seven of the 10 fastest-growing occupations in Florida are in STEM fields.

The 2014 average annual wage for workers in the information technology industry was $78,139, exceeding the total average annual wage for all industries by 74.4 percent. Software developers are projected to gain the most jobs.

Mobile apps

One of the most exciting, dynamic and fast-growing careers in information technology today is in mobile app development. We surround ourselves with this technology, and the more it grows, the more app developers are needed to keep up with the demand.

Not too long ago, creating software applications for any system required expensive machinery, lots of punch cards and an exceptional amount of patience. Today, people of all ages and skill levels can create their very own mobile apps in the palms of their hands!

What is computer science?

Computer science can be defined as the study of computers and their uses, or, more technically, as a branch of science that deals with the theory of computation or the design of computers.

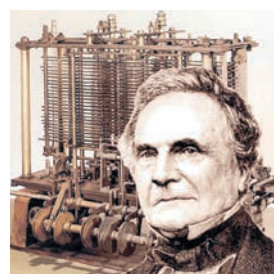
Computer hardware and software

Hardware is all the physical components that make up a computer or device. A desktop computer tower — and everything inside it — keyboard, mouse and monitor are all hardware. So is your laptop, tablet or cellphone.

Software is the generic term for the programs that tell the physical components what to do. Web browsers such as Chrome, word processors such as Microsoft Word, social media apps such as Facebook and games such as Minecraft are all software.

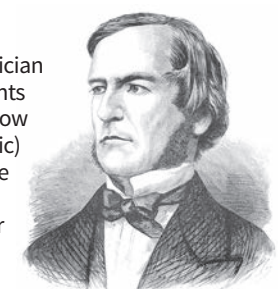
1801
French inventor Joseph Jacquard builds an automated loom that is controlled with punch cards. Early computers would use similar punch cards.

1822
English mathematician Charles Babbage demonstrates a prototype of his Difference Engine calculating machine.



1833
Charles Babbage begins work on his Analytical Engine, one of the first computational machines.

1847
English mathematician George Boole invents a symbolic logic (now called Boolean logic) that would become basic to the design of digital computer circuits.



Operating systems

An **operating system**, or OS, is a special type of software that manages the hardware and software on a computer or mobile device. You can think of the OS as the master set of instructions that runs behind the scenes.

For example, your desktop computer or laptop most likely runs on Microsoft Windows or Apple MacOS, while your cellphone or tablet probably runs on Apple iOS or Android.

Computer programming languages

A **natural language** is a language spoken by humans to communicate with each other, such as English or Spanish. Natural languages are very complex, and because of this, they do not work well for communication between humans and computers. Programming languages are designed for the specific purpose of creating instructions that can be executed by computers.

The lowest level of computer programming language is **machine language**. Machine language consists of instructions written in binary code – 0s and 1s – that a computer can execute directly. Machine language is difficult for humans to read and write, since it does not resemble human language.

Assembly language is one level above machine language. It allows a programmer to use simple commands instead of binary digits. Both machine and assembly language are considered **low-level programming languages**.

Today's programmers use **high-level programming languages**. High-level programming languages more closely resemble natural languages. Similarly to natural languages, high-level languages have a defined set of words that carry certain meanings and require that these words follow standard rules for placement and organization. This is known as **syntax**.

Programs written in higher-level languages must be **compiled**, or translated, into machine language before they can be executed by a computer.

Customized languages

Each of the major mobile device platforms has its own programming language that is used to specify instructions. Apple's current iOS platform relies on one of two languages called Swift and Objective-C, while the Android platform relies on the Java programming language. These languages are referred to as **native** because they were specifically customized to work directly with the operating system of the mobile device.

Apple and Android devices also are able to understand and process apps written in a language called Javascript. Javascript was originally created to allow websites to be more dynamic. While Javascript was originally developed as a web programming language, it can be used to create fully functional apps that can run on different mobile operating systems.

Vocabulary

- Assembly language
- Compile
- Hardware
- Low-level language
- High-level language
- Machine language
- Native language
- Natural language
- Operating System (OS)
- Software
- Syntax

Going beyond the text

Exploring language

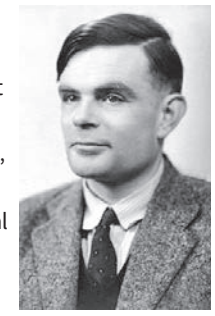
According to the *Cambridge Dictionary*, a natural language is a language developed as a method of communicating between people. It is spoken by humans to communicate with each other. One aspect of natural language is jargon. *Merriam-Webster Dictionary* defines jargon as technical terminology of a special activity or group. Businesses, hobbies and special interests all have jargon: computers, sports, music and literature are all examples. As you read this Newspaper in Education publication, write down all of the words you encounter that you would define as jargon. Write down the definition of the words as they pertain to computer science. Next, look in the *Tampa Bay Times* to find more words that you would define as jargon. Write these words down in your notebook and note what category they relate to.

> Our future depends on reaffirming America's role as the world's engine of scientific discovery and technological innovation. And that leadership tomorrow depends on how we educate our students today, especially in math, science, technology and engineering.
>
> **— former President Barack Obama**

Sources: BBC; *Encyclopaedia Britannica*; How Stuff Works; *DevKit: Build Games and Learn to Code* by Mitch Marchand and Dr. Christopher Maurer; *Introduction to Computing: Explorations in Language, Logic, and Machines* by Dr. David Evans

1890
Herman Hollerith designs a punch card system to calculate the 1880 census and establishes a company that would become IBM.

1936
British mathematician and computer scientist Alan Turing completes his seminal paper "On Computable Numbers" and introduces the concept of a theoretical computer called the Turing Machine.



1942
American physicists John Vincent Atanasoff and Clifford Berry design the Atanasoff Berry Computer (ABC), the first electronic digital computer to perform numerical calculations digitally and store information on its main memory. It can solve 29 equations simultaneously.

1943
Colossus, the world's first large-scale programmable electronic digital computer, is built in Britain by a team of mathematicians, electrical engineers and intelligence agents to crack Nazi codes.

Core concepts

What is an algorithm?

An **algorithm** is a series of instructions that take in some type of input and produce some type of output.

Simple algorithms are often compared to recipes, which take in a set of ingredients and instructions that produce an output of cake or chili.

In computer programming, an algorithm is a set of steps that direct a computer in one or many directions to produce a solution to some type of input. The steps in the algorithm describe how to use the input to accomplish a task.

Going beyond the text

Creating an app

Look through the BayLink, Local, Sports or Taste sections of the *Tampa Bay Times* for an idea for a simple mobile app or game. Once you decide on an idea, complete steps 2, 3 and 4 on Page 6: Define the goal of your app, identify its key features and draw out the main user flow(s). Share the details of your game with your class, including what inspired you to create the game.

Creating an algorithm

Let's say you are a grading computer system, and a student has input a series of five scores between 0 and 100 that he or she received on tests. The task that you need to accomplish is determining the student's letter grade based upon the average test score.

To calculate the user's letter grade, the computer needs to calculate the average of the scores and then assign a letter grade to the average. To accomplish this, the programmer can use an algorithm, as shown below:

Step 1: Add the five scores to get the total
total = score1 + score2 + score3 + score4 + score5

Step 2: Calculate the average score
average score = total score / 5

Step 3: Output the letter grade based upon the average score
if average score is less than 60,
    set letter grade equal to "F"
otherwise if average score is less than 70
    set letter grade equal to "D"
otherwise if average score is less than 80
    set letter grade equal to "C"
otherwise if average score is less than 90
    set letter grade equal to "B"
otherwise
    set letter grade equal to "A"
then end if statement

Going beyond the text

Creating algorithms

An algorithm is a series of instructions that takes in some type of input and produces some type of output. In computer programming, an algorithm is a set of steps that directs a computer in one or many directions to produce a solution to some type of input. Using the Sunday through Friday editions of the *Tampa Bay Times* as your sources, create algorithms to determine the percentage of news articles, advertisements and photographs for each of the following sections:

• Nation & World • Local • Sports

Note: You must first determine the total number of news articles, advertisements and photographs in the newspaper. Share your results with your classmates.

Extra credit

Using the "Creating an algorithm" example on Page 4 as an example, create a grading system for this week's baseball, hockey or football statistics.

Programming environments

The term **programming** or **development environment** refers to the tools a programmer uses when he or she actually creates computer code.

The simplest programming environment is a text editor. The developer simply types in the lines of code and saves the program as a text file. A program editor is a type of text editor that is specifically designed for writing computer code.

A **visual development environment** (VDE) provides programmers with point-and-click tools to create applications without typing code. An integrated development environment (IDE) typically includes a comprehensive set of tools that allow the user to create, test and debug their code.

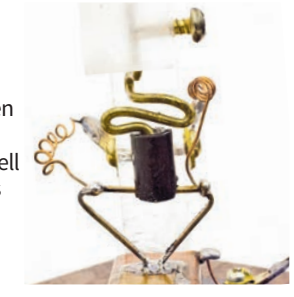
Sources: BBC; *Encyclopaedia Britannica*; How Stuff Works; *DevKit: Build Games and Learn to Code* by Mitch Marchand and Dr. Christopher Maurer; *Introduction to Computing: Explorations in Language, Logic, and Machines* by Dr. David Evans

1945
American physicist John Mauchly and American engineer J. Presper Eckert create ENIAC (Electronic Numerical Integrator And Computer), the first American electronic digital computer, to run ballistics calculations for the United States Army. It filled a 20-foot-by-40-foot room and had 18,000 vacuum tubes.

1946
Whirlwind, the first general-purpose digital computer able to operate in real time, solves its first problem at MIT.

1946
AT&T Bell Labs introduces the first mobile telephone. It is car-based and weighs 80 pounds.

1947
William Shockley, John Bardeen and Walter Brattain of Bell Laboratories invent the transistor.



1950
Alan Turing proposes a test to determine whether or not a machine has gained the power to think for itself. It becomes known as the Turing Test.

Designing a mobile app

No matter what type of program or app you want to create, the basic steps are the same.

Step 1: Think of an idea. The best apps solve a problem that people encounter in their daily lives. If you don't already have an idea, think about problems you have faced and how an app might help solve them.

Step 2: Define your goal. What is the purpose of your app? What will it do? What problem will it solve or make better?

Step 3: Identify key features. What are the key features and functions that you want to include in your app?

Step 4: Document user flows. How will the user navigate through your app to perform these functions? Draw this out on paper before going any further.

Step 5: Build your app. This is where you will actually code and document your program.

Step 6: Test and troubleshoot your app. Review the look and feel of your app and go through each of the user flows that you defined above to verify that the user is able to complete it as designed. Because users often perform unexpected actions, you should also test for other actions not specified in your user flows.

Step 7: Release your app.

Documenting user flows

For each **user flow**, it is best to create an outline that begins with an entry point and ends with an end point. The entry and end points of mobile applications will commonly be home or main menu screens. In between the entry and end points, you should identify intermediary screens and options that will be available to the user.

The goal of this outline is to lay out the screens in a logical order and to identify which elements will appear on different screens to allow the user to perform all necessary functions.

When considering intermediary steps in the user flow, consider how much information or functionality you want to provide on each individual screen. Sometimes it may make sense to split the process into multiple screens to minimize the user scrolling through long lists of items.

To illustrate a typical user flow, consider an app that requires the user to create a user account before accessing any functionality. There are many possible ways to accomplish this common task, but one possible outline could look like this:

Example

Documenting user flows

1. Entry points
   a. User opens the app for the first time.
   b. User clicks on the "Create New Account" option.
2. Display "Account Credentials" screen.
3. Collect the following information
   a. First and last name
   b. Username
   c. Password
   d. Repeat password
   e. Email
   f. Phone number
4. User clicks "Continue."
5. Display "Preferences & Locale" screen.
6. Collect the following information
   a. Country of residence
   b. Time zone
   c. Preferred language
7. User clicks "Create My Account."
8. Display name and username to the user and a message stating that a confirmation email was sent to the email address provided.
9. User clicks "OK" to proceed.
10. End points
    a. Display the home screen of the app
    b. Ask the user to sign in

Activity

Documenting user flows

You have decided to develop a simple mobile game called Grab the Crab, similar to Whack-a-Mole.

The game screen will display a sandy beach with several holes in it. Crabs will randomly appear from and disappear into these holes. The goal of the game is to "grab" each crab by touching it when it appears from a hole before it goes back inside. In addition to the "Play Game" screen where users try to grab crabs, users also should be provided with a "High Score" screen with the ability to save high scores, a "Game Over" screen, and some instructions on how to play.

The goal of this activity is to begin thinking like an application developer and envision what steps would be required to implement this functionality. Using the above outline as an example, create an outline of each of the following user flows:

1. Playing the game
2. Loading the "Game Over" screen
3. Viewing top scores

User interfaces

One of the most important aspects of any computer system is its **user interface**, or how users interact with it.

Graphical user interfaces, or GUIs, allow users to provide input and receive feedback from the computer system through visual images such as menus, status bars and icons.

The best user interfaces anticipate what actions users will need to perform and provide elements that are easy to understand and use to perform those actions.

In a mobile environment, designing a beautiful and easy-to-use user interface is extremely important because of the limited size of mobile device screens and the small number of gestures permitted in mobile operating systems.

The best mobile apps have user interfaces that are intuitive, functional and easy to use. Design elements such as color schemes, navigation and screen layouts are all critical to the usability and appeal of an app.

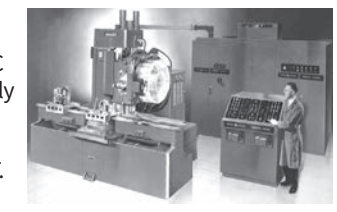
Sources: Apple Inc.; Entrepreneur.com; *DevKit: Build Games and Learn to Code* by Mitch Marchand and Dr. Christopher Maurer

Vocabulary

Algorithm
Graphical user interface (GUI)
Programming environment
User flow
User Interface (UI)
Visual development environment

1951
UNIVAC (Universal Automatic Computer), the world's first commercially available computer, is designed by ENIAC creators Eckert and Mauchly. The first UNIVAC came online for the U.S. government's Census Bureau.

1952
The first NC (numerically controlled) machine is built at MIT.



1953
American mathematician and computer programmer Grace Hopper develops the first compiler, leading to the development of the first high-level computer language, COBOL.

1956
American researchers Allen Newell, Herbert Simon and John Shaw create the Logic Theorist, the first artificial intelligence program.

DevKit:

A different way to touch and learn

Traditionally, beginning programmers are taught the logic and concepts of software development by learning a particular programming language. Many aspiring coders get discouraged at the steep learning curve.

However, all programming languages rely on a set of similar underlying concepts and principles. If these basic concepts are taught independent of a programming language, newcomers to software development can apply the same principles and create applications without writing any code. Those who are interested in studying further are well prepared to learn the programming syntax for a multitude of languages.

DevKit was created by developers for aspiring developers. It teaches all of the same logic, concepts, procedures and terminology used every day by professional software developers. DevKit's goal is to lower the barriers to entry for software development and make it easier to begin learning how to build apps.

The DevKit app and approach for teaching development principles have four key benefits over traditional methods for learning how to develop software:

1. No coding languages are required.
2. DevKit runs in a user-friendly, non-threatening environment.
3. Apps can be tested and executed in real time, with no need for multiple devices or complicated emulators.
4. Learners can build high-powered apps right away.

DevKit embodies the application development process as a system of moving parts. It breaks down the key aspects of coding and places them into an intuitive structure that makes it easy for students to create, edit and test apps right away. These concepts can be extrapolated and applied to learn any programming language.

Connection to Apple

Since the introduction of the iMac in 1998, Apple has dominated the technological education space by providing schools around the country with easy access to their user-friendly systems. Over the years, Apple has come to be known for their ability to connect people of all ages to the latest and greatest technology. DevKit is rapidly becoming aligned with that exact same identity.

Fast forward to the present day, and Apple's 1:1 iPad program in classrooms is one of the fastest growing advances in technological education. Students who grow up using their mobile devices are now just as comfortable using that technology in the classroom. With tools like DevKit, students are finding new ways to explore their creativity – and they're doing it in the palms of their hands.

DevKit is an Apple-only program because the company sees the importance of aligning the perception of DevKit's application development process with what Apple has done for students during the past 20 years. DevKit provides an interface for creative students to learn and grow in brand new ways, which is exactly what is needed to grow the next generation of app developers.

Real-world applications

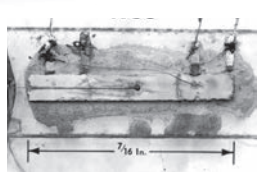
With the DevKit platform, students are not just playing around with a learning tool, but actually building real-world applications that can be deployed to the iOS App Store.

On most learning platforms, users are typically limited to creating apps that can only run within the system that was used to build it. But with DevKit, students develop apps that can run on any iOS device worldwide.

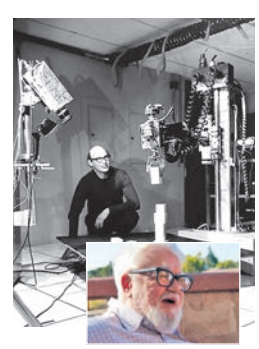
The DevKit eBook contains complete instructions on how students can enroll in the Apple Developer Program and submit their apps created in DevKit for release through the App Store.

Although Apple charges a yearly subscription fee of $99 to submit apps to the App Store, students who participate in a DevKit Hackathon (see Pages 12-13) will have the chance to have their apps published in the App Store for free!

1958
Engineer Jack Kilby and physicist Robert Noyce invent the integrated circuit. Kilby is awarded the Nobel Prize in Physics in 2000 for his work.



1959
Researchers John McCarthy and Marvin Minsky start the Artificial Intelligence Laboratory at MIT.



1964
Engineer Douglas Engelbart demonstrates a prototype of the modern computer with a mouse and a graphical user interface (GUI).

1964
Artificial intelligence research laboratories are opened at MIT, the Stanford Research Institute (SRI), Stanford University and the University of Edinburgh.

1965
Stanford University researchers Edward Feigenbaum and Joshua Lederberg create DENDRAL, the first expert system designed to execute the accumulated knowledge of subject experts.

Going beyond the text

Education and technology

There is no question that technology has changed the world of education. From computers, SmartBoards and iPads in the classroom to online education and YouTube lectures, technology has changed the way people approach learning. It also has changed the way we prepare for current and future careers. Look through the *Tampa Bay Times* and write down all the different types of technology you find in the newspaper. Highlight or underline the technologies that are used in your school. Select one of these technologies to research. Find out as much information as you can about the history of the technology and how it is used for business, education and/or entertainment. Write a fully developed essay based on this research. Be sure to use reliable sources and to document those sources. Share what you have learned with your classmates.

Information for educators

The DevKit Classroom Package

The DevKit Classroom Package is a complete bundle of all the tools and resources needed in order to implement a DevKit course at an educational institution. It's been engineered to provide a complete solution for your classrooms, from device management to classroom activities to in-app assessments for the students and much more. When you order the Classroom Package, you'll choose the number of student licenses that you'd like to purchase. Each student license grants unlimited access to all the features of the DevKit application. The cost of each student license is $100 per year.

The Package also contains a free instructor license, which includes access to all the features of the student license as well as access to the instructor portal to view student assessment submissions. The entire list of features included in the Classroom Package includes:

- A set of student licenses with a corresponding instructor license
- The DevKit eBook, containing 13 chapters, 37 sections and 29 in-class activities
- 27 step-by-step video tutorials to walk students through the in-class activities
- 20 in-app assessments to gauge students' understanding
- Unlimited access to the DevKit app

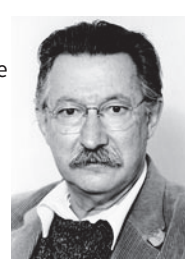
To learn more about the DevKit Classroom Package for educators, visit **DevKitApp.com.** For details on free trial opportunities, contact Mitch Marchand at **marchand@vybesoftware.com.**

Assessments

DevKit offers a series of in-app assessments to check students' understanding of the material as you move through the eBook. When a student completes an assessment, he or she submits it through the app to the instructor, giving him or her access to a fully functional version of the student's app and its DevKit components.

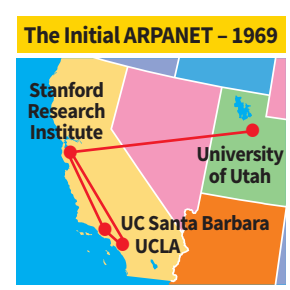
Each assessment consists of a partially completed DevKit app that is missing a component that was covered in the section to which the assessment pertains. For example, after reading Section 9.1 - The If Statement and completing the practice activities, students can navigate to the Assessments section of DevKit and add Assessment 9.1, which is a partially completed application, to their apps. Since Section 9.1 covered use of the if statement, students will need to add an if statement to the application to complete it. Once they've done so, they can navigate to the Build section of DevKit and submit their completed app to the device of their instructor.

1966
An artificial intelligence program named ELIZA is created at MIT by Joseph Weizenbaum. ELIZA functions as a computer "psychologist" that manipulates its users' statements to form questions.



1967
MIT researcher Richard Greenblatt writes MacHack, the first chess program to win against a person in a chess tournament.

1969
The first message is sent over ARPANET (Advanced Research Projects Agency Network), an experimental computer network that was the forerunner of the Internet. Its initial purpose was to link computers at Pentagon-funded research institutions.

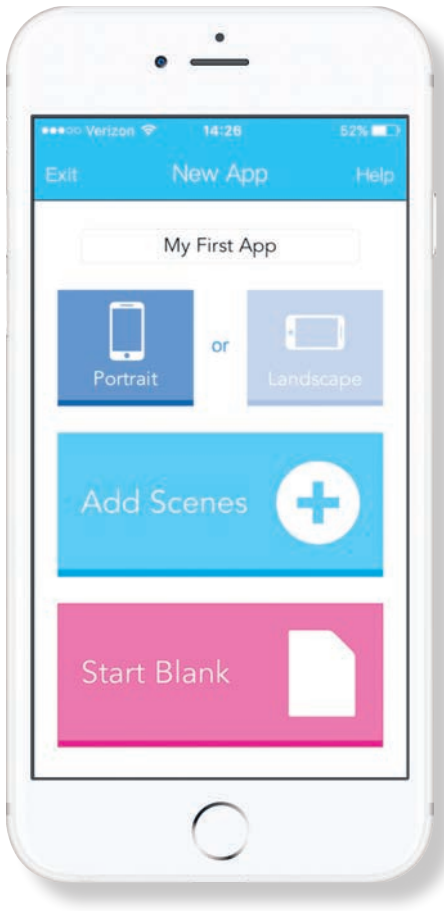


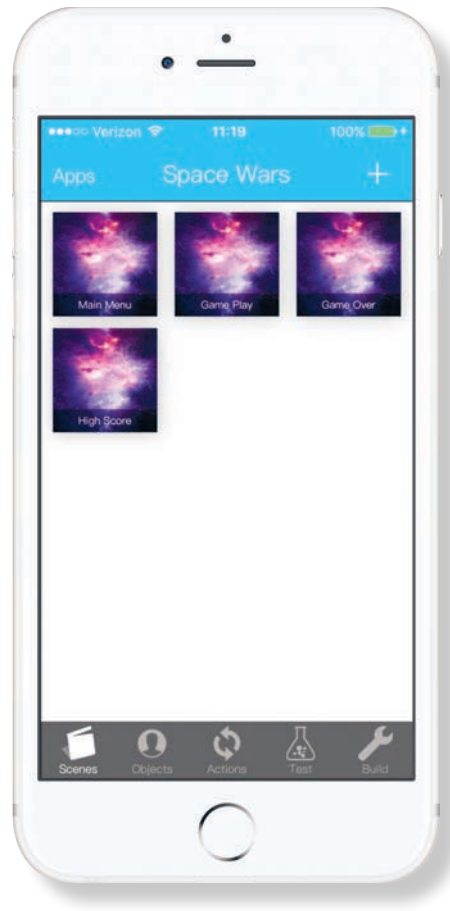
1970
Intel releases the Intel 1103, the first Dynamic Random Access Memory (DRAM) chip.

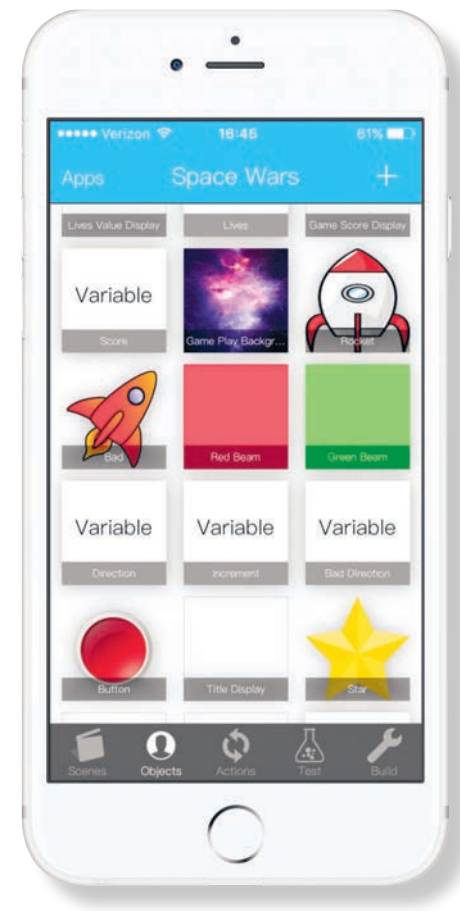
1971
IBM engineers invent the floppy disk, allowing data to be shared among computers.

Using DevKit to design an app:

Scenes, objects and actions



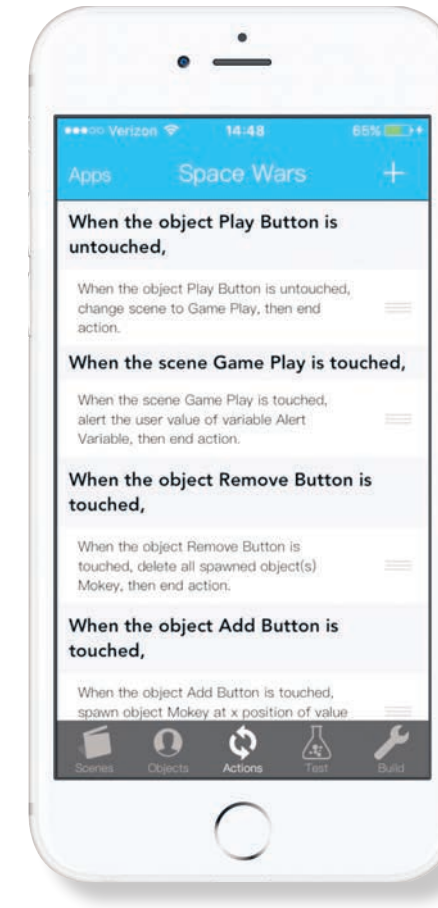




- Variable objects allow you to store values that can be used throughout the app, such as names, scores, titles or any text that the user can see.
- Group objects are collections of objects that are related to each other or used in unison with one another.

Instances

Objects must be created before they can be added to your scenes. When you create



Scenes

The basic building block of app design within DevKit is the **scene.** Scenes are used to organize content on a single view that users see when they use your app. Scenes can be simple and straightforward, such as a home scene that is used to display a few buttons, or they can be complex and allow for detailed animations and advanced user interaction.

DevKit provides customizable template scenes for layouts that are common in mobile apps. Or, you can choose to create a blank scene and design it from the ground up.

You can control certain features, or **properties**, of your scenes. Scene properties include name, orientation (vertical or horizontal) and background image or color.

Your user flow will determine how many and what kind of scenes you will need in your DevKit app.

Objects

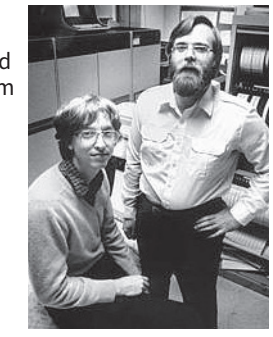
The second component of a DevKit app is the **object**. Objects make up everything the user sees in your scenes. There are four main types of objects in DevKit:

- Image objects are pictures that you can add to your scenes for the user to see.
- Color objects are simple colored shapes that you can add to your scenes.

1971
Texas Instruments introduces the first electronic handheld calculator, the Pocketronic. It weighs more than 2 pounds and can only perform the four main arithmetical functions.

1971
American computer engineer Ray Tomlinson writes the first email program. He chooses the symbol @ to separate the name of the recipient from the destination address because it was not used in names or the programming languages then in use.

1975
Paul Allen and Bill Gates form Microsoft.



1975
The MITS Altair 8800 is introduced as the first "personal" computer.

a new object in your DevKit app, you are specifying the basic blueprint of that object. You can use this blueprint to add multiple copies, or **instances**, of the same object to your app. Each instance is one particular occurrence of that object in one of your scenes.

Say you create a color object that consists of a black square named "box." You can add multiple copies of the "box" object to your scenes. Each copy is one instance of the "box" object. Creating a new instance of an object is sometimes called **spawning.**

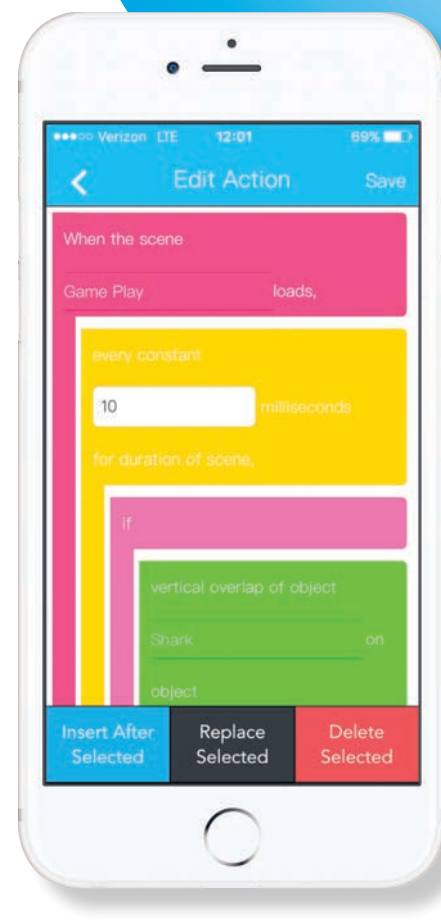
Like scenes, objects have properties. Some object properties are shared by all instances of an object. Others can be modified for each instance of an object. Object properties that you can manipulate include name, color, size and position in a scene.

Actions

The third and final component of a DevKit app is the **action**. Actions tell the app what functions to perform as a result of certain **events**. They define how the user will interact with your app.

The very first thing you must do when creating a new action is add a new **event listener**. An event listener waits for a specific event, such as a finger tap or swipe, to occur. When the event being listened for is triggered, the app will perform some action in response.

Event listeners in DevKit apps can be assigned to individual objects or entire scenes. They listen for events only within the boundaries that you set. This is why you can touch different areas on the screen and trigger different actions.



Each object in a scene can be assigned a different event listener. Or, one object can be assigned multiple event listeners, so that different actions will be performed when different events are triggered. For example, a certain button may perform different actions if the user simply touches it or if the user touches and holds his or her finger on the button for a certain length of time.

In DevKit, there are three primary events that your actions can listen for: touch, untouch and load.

- A **touch event** is triggered whenever the user touches a particular point on the screen. Touch event listeners can be assigned to individual objects or to an entire scene.
- An **untouch event** is triggered only after a touch event has ended, which happens when a user lifts his or her finger off the screen. Untouch event listeners can be assigned to individual objects or to an entire scene.
- A **load event** occurs when a new scene is presented to a user. This type of event listener will trigger an action immediately upon displaying the new scene.

Commands and parameters

As you build an action, DevKit works behind the scenes to write the computer code that will perform the functions you want. Each action is actually a series of **commands** that provide specific instructions for performing a task.

Parameters fill in the gaps in your commands and specify the values used to complete them. For example, in the action "Change scene to Second Scene," "Change scene to ____" is the command and "Second Scene" is the parameter used to complete the command.

Going beyond the text

Deconstructing the whole

DevKit uses three components to build an app: scenes, objects and actions. You can use the same concepts to analyze the comics in the *Tampa Bay Times*. Choose one comic from the *Times* and break down each panel into its component scene, objects and actions. Describe your deconstructed comic in a blog post. Share your thoughts with your class.

What is a variable?

Variables in computer programming are containers, or placeholders, for information that can change.

Computer programs must be able to remember certain things, such as the name of a user, the last level that was completed in a game or the highest score previously achieved.

Programs use variables to keep track of these values. Each variable has two main properties: an identifier and a **value**.

The identifier is a unique name that is assigned to the variable. The identifier does not change as the application runs.

The value is the actual data that is stored for the variable. Values often change as the application runs.

As an example, imagine a game that records the high score. The first time a user opens the game, the variable with the identifier "High Score" will have a value of 0. After a few games, "High Score" may have a value of 220. If another user plays the game and records a score of 240, then the value of the "High Score" variable will change to 240.

Vocabulary

Action
Command
Event
Event listener
Instance
Object
Parameter
Property
Scene
Spawn
User interface
Value
Variable

Sources: Apple Inc.; Code.org; *DevKit: Build Games and Learn to Code* by Mitch Marchand and Dr. Christopher Maurer; usability.gov

1976
Steve Jobs and Steve Wozniak form Apple Computers and release the Apple I. It does not come with a case, power supply, keyboard or display, all of which have to be supplied by the user.

1977
Radio Shack introduces the TRS-80 personal computer.

1977
Jobs and Wozniak introduce the Apple II, which features 16K of RAM, color graphics and an audio cassette drive for storage.

1977
AT&T Bell Labs introduces the first public cellphone network, AMPS (Advanced Mobile Phone System). It is North America's first 1G analog service.

1978
The first computerized spreadsheet program, VisiCalc, is released.

Building your first DevKit app

This activity will walk you through designing and building an app called "Leaper" using DevKit.

Part 1: Planning your app

"Leaper" will be a game that consists of an animal that jumps up to try to catch an item. To plan your game, you must decide what kind of animal you'd like to use, what type of item the animal will try to catch and what background your game should have. Your game can take place on a beach, in the woods, in a city, in the sky — anywhere you can imagine!

Part 2: Creating objects

Objects make up everything that the user sees in a DevKit app. In this section, you will create three objects for the "Leaper" app.

1. Open the DevKit app. From the "My Apps" view, tap the plus sign in the upper right corner. Name the app "Leaper," then select "Add Scenes."
2. From the next view, tap the circle in the "Game Play" row.
3. Press "Done" to create the app.
4. Tap on the "Leaper" app that you just created, then select the "Objects" tab at the bottom of the view.
5. To create your animal, tap the plus sign in the upper right corner of the "Objects" tab. To name your new object, type "Animal" in the input box at the top of the screen.
6. Use the search bar to search for the type of animal you'd like to use. Once you've found one, tap on it, then tap "Add" in the top right corner.
7. Repeat steps 4 and 5 to create the object for the animal to catch. Name this object "Item."
8. From the "Objects" tab, scroll down until you see the "Game Play Background" object. Tap on it, then choose "Options" in the bottom left corner.
9. Use the search bar to search for a background image for your app. Once you find one that you like, tap on it and then press "Save."

Part 3: Creating scenes

Scenes are used to organize content in views for the user. In this section, you will set up the layout of the "Game Play" scene using a customizable template and the objects you created in Part 2.

1. Navigate to the "Scenes" tab at the bottom of the view. Tap the "Game Play" scene, then tap the "Edit Scene" button. This is what the "Game Play" scene template looks like.
2. The "Lives" and "Time" objects are not needed for the "Leaper" app. To remove them from the scene, touch and hold the text that says "Lives," then let go. You should see a black bar appear at the bottom of the screen.
3. Next, tap on "Menu" in the upper left corner and choose "Remove Selected". A confirmation message will appear. Select "Remove" to delete the "Lives" object from the scene.
4. Repeat steps 2, 3 and 4 to remove the "0" text next to where the "Lives" object was. Then repeat them again to remove the "Time" object and to remove the "0" next to where the "Time" object was.
5. Tap on "Menu" again and choose "Add Object". If an alert appears, press "Save and Add."
6. Scroll to the bottom of the list and tap the circle in the row for the "Animal" object that you created, then tap the circle in the row for the "Item" object. Then press "Add" to add these two objects to the scene.
7. Drag the "Animal" object into the bottom center of the scene, then let go. Use the circle that appears at the bottom right corner of the "Animal" object to adjust its size and make it a little smaller.
8. Drag the "Item" object into the top center of the scene. Use the circle to adjust its size as well. Make it so the width of the "Item" object is about 1/3 the width of the scene.
9. Once you're done, press "Menu," then press "Save," then press "Exit."
10. Once you're back at the Edit Scene view, tap the back arrow in the top left.

Part 4: Creating actions

Actions tell a DevKit app what functions to perform as the user interacts with the app. In this section, you will create two actions: one to make the "Animal" object jump and one to monitor whether or not the "Animal" object caught the "Item" object.

1979
The first computer word processing program, WordStar, is released.

1981
The first IBM personal computer is introduced. It uses Microsoft's MS-DOS operating system and features an Intel chip and two floppy disk drives.

1983
Apple's Lisa, the first personal computer with a GUI, is released.

1983
The first flip top "laptop" computer, the Gavilan SC Mobile Computer, is introduced.

1. Navigate to the “Actions” tab at the bottom of the view and tap the plus sign in the upper right corner to begin creating an action.
2. Select “Add Event Listener” and choose “When the scene _____ is touched” from the list.

Event listeners wait for the user to perform a specific event in order to trigger an action. In this case, the event listener will wait for the user to tap on the “Game Play” scene. When he or she does so, the event listener will trigger the “Animal” object to jump.

3. Tap “Insert After Selected.” Scroll down the list until you find the command “make _____ spawned object(s) _____ jump by a height of _____ at a speed of.” When you find it, tap on it. Tip: commands are ordered alphabetically!
4. Swipe up on the text in the command that says “Score Colon Display” until you find the “Animal” object.
5. The value that you enter in the input box defines how high the “Animal” object will jump. If you are using an iPhone, enter 400. If you are using an iPad, enter 600.
6. To define how fast the “Animal” object will jump, tap on “Insert After Selected” again and scroll down the list until you find the “value of _____” command.
7. Objects in DevKit can move at speeds between 1 and 100, where 1 is very slow and 100 is very fast. Type 80 into the “value of _____” input box.
8. Finally, press “Insert After Selected” again and choose “, then end action.” to tell the app that the action is over.
9. Starting with the event listener at the top of this section, read the words of the action back to yourself to understand how they will cause the “Animal” object to jump. When you are done, press “Build” in the upper right corner.

The second action that you will build uses some advanced commands. Do your best to understand the commands as you add them, but don’t worry if you find them difficult to understand at this point.

This action will constantly monitor the “Game Play” scene to see whether the “Animal” object comes into contact with the “Item” object. If it does, it deletes the “Item” object and increases the “Score” by 1. Then, it waits a half-second and spawns a new “Item” object at the top of the scene for the “Animal” object to try to catch.

1. Just as you did for the first action, start by tapping the plus sign in the upper right corner, then tapping “Add Event Listener.” Select “When the scene _____ loads” from the list.
2. Tap “Insert After Selected” and choose the “every _____ milliseconds for duration of scene” command from the list. Disregard the yellow box that appears and press “Done” to add the command to the action.
3. Add each lettered command to the action in the order shown below.

Each command can be added by tapping “Insert After Selected” from within the action builder and choosing the command from the list.

The italicized words go in the blank spaces of the command, either by swiping up and down on a list of objects or by typing a value into an input box.

a. if
b. object Animal is touching object Item
c. , then
d. delete first spawned object(s) Item
e. , and do
f. increment variable Score by
g. value of “1”
h. , and do
i. after “1000” milliseconds (Again, disregard the yellow box and tap “Done” to add this command.)

j. spawn object Item at x position of
k. value of “100” if using an iPhone or “250” if using an iPad
l. and y position of
m. value of “50” if using an iPhone or “100” if using an iPad
n. , then end if statement
o. , then end action

4. When you are done, press “Build.”

Part 5: Testing your app

Navigate to the “Test” tab at the bottom of the view and tap the button in the middle of the screen. Test whether or not the game works as designed by tapping the screen to make the “Animal” object jump in the air toward the “Item” object.

If the “Animal” object is not jumping high enough to catch the “Item” object or is jumping too high and touching it twice, you have two options to fix the app.

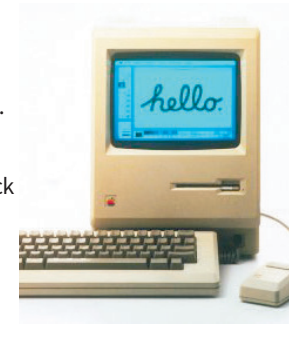
1. To make the “Animal” object start from a higher or lower position, open the “Game Play” scene and drag the “Animal” object to a different position in the scene.

2. To make the “Animal” object jump higher or lower, navigate to the “Actions” tab, tap on the action in the “When the scene Game Play is touched” section, and locate the “make all spawned object(s) Animal jump by a height of _____ at a speed of” command. In the text box that has a value of 600 (if using an iPad) or 400 (if using an iPhone), change the value to a larger or smaller number.

Congratulations – you have just completed your very first DevKit app! Modify the objects, scenes and actions that you created to customize your app more or add additional features. Happy coding!

1983
Motorola Corp. researcher Martin Cooper invents the first cellphone approved for commercial use, the DynaTAC. It weighs 2 pounds.

1984
The Apple Macintosh is released. It features a built-in, 9-inch black and white screen.



1985
Microsoft releases the Windows 1.0 operating system.

1985
The first domain name, symbolics.com, is registered on the Internet by the Symbolics Computer Co.

1990
Researcher Tim Berners-Lee develops HyperText Markup Language (HTML).

Hold a DevKit **Hackathon** at your school!

What is a Hackathon?

Despite what it sounds like, a Hackathon is not about hacking! Instead, it is a fun team learning event where developers come together to collaborate and create a solution to a problem using technology.

A DevKit Hackathon allows students to compete against other students across the state of Florida to build the coolest mobile apps and games that they can think of with DevKit.

A DevKit Hackathon is a perfect opportunity for your school to showcase the creative talents of your students to the world, and to let everyone know that you have some of the most creative young app developers out there.

DevKit experts will come to your school, teach your students how to use DevKit and get them started on building their ideas. At the end of the Hackathon, the best apps will be published in the iOS App Store by Vybe Software for free.

The best part? DevKit Hackathons are 100 percent free for schools!

Hosting a DevKit Hackathon at your school

- DevKit Hackathons are 100 percent free for schools. All that is needed is a classroom and access to iPads or iOS devices.
- A typical Hackathon involves several two-hour sessions spread over three to four days. The more time that your students have to practice using the app, the better their creations will be.
- Your Hackathon can involve one group of students or multiple sessions with different student groups. Sessions can be organized back-to-back or simultaneously.
- DevKit is currently an Apple-only development program. Students will need access to an iPad or another iOS device in order to participate.

Get started

For more information on how to host a free DevKit Hackathon at your school, visit **DevKitApp.com/Hackathon.**

Activity

Hacking

According to many popular movies and television shows, hackers are criminals who gain unauthorized access to computer systems for malicious reasons.

However, many computer professionals and enthusiasts object to the use of the word "hacker" in this context. In their view, a hacker is simply someone who enjoys figuring out new and innovative uses for technology. The bad guys are known as *black hat* hackers.

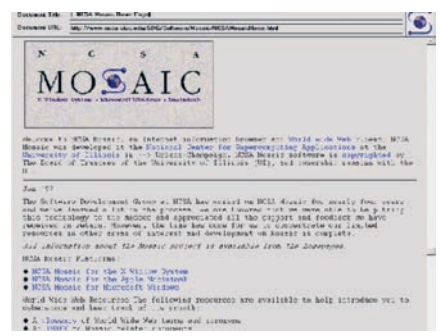
Ethical hackers, also known as *white hat* hackers, often are employed as cybersecurity professionals who try to identify and patch any security weaknesses in computer systems before unauthorized users are able to gain access.

Watch the short Nova Cybersecurity Lab video "The Secret Lives of Hackers" at mass.pbslearningmedia.org/resource/nvcy-sci-slhackers/the-secret-lives-of-hackers. In small groups, discuss the following questions:

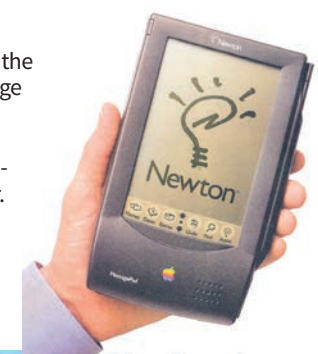
- What is hacking?
- Name some examples of hacks outside of computing.
- What are some reasons that people hack computers?
- How can hackers protect people from exposing personal information?

1992
The first text-based web browser is released.

1993
The first point-and-click web browser, Mosaic, is released.



1993
Apple releases the Newton Message Pad, the first commercially available hand-held computer. It weighs more than a pound and features handwriting recognition.



1995
Microsoft releases the Internet Explorer web browser.

Going beyond the text

Information technology careers

Did you know that information technology specialists work in almost every professional career field — from NASCAR to fashion design? From creating custom business software to identifying the reason a car engine is misfiring to designing and printing a daily newspaper, information technology is indispensable to today's businesses. Look in the *Tampa Bay Times* for jobs and career choices that involve IT. Make a list of all of the jobs you find. What skills are needed for these positions? What type of education is required? Discuss what you have found with your class.

Reading closely

On Pages 14 and 15, three professionals working in the IT field share insights about what knowledge and skills students interested in an IT career should work toward. Read each interview carefully. Choose one of the interviews and rewrite it in the form of a newspaper article. Use the articles in the *Tampa Bay Times* as a model. Your article needs to have an introduction and conclusion. Your article should begin with a *lead*, which is similar to a *hook* in essay writing: a statement to grab the reader's attention. You also need a headline for the article.

Careers in information technology

Information technology (IT) professionals implement and manage computer hardware and systems; design, implement and manage network systems; design and develop computer software; and design and develop multimedia and web products and services. IT professionals work in every sector of the economy.

IT positions typically require a combination of education and training, work experience and specific technical skills. Although some entry-level jobs may require only an associate's degree, many positions will require at least a bachelor's degree, and some may require additional training or industry certifications.

On Pages 14 and 15, three professionals working in the IT field share insights about what knowledge and skills students interested in an IT career should work toward.

Sources: Minnesota State CAREERwise Education, O*NET OnLine, United States Department of Labor Bureau of Labor Statistics

Florida occupations with the highest entry-level wages

Occupational Title	Hourly Wage Mean	Hourly Wage Entry
General and Operations Managers	63.16	32.07
Administrative Services Managers	52.64	31.94
Software Developers, Applications	42.11	26.69
Computer Systems Analysts	40.24	26.13
Network and Computer Systems Administrators	39.49	25.98
Information Security Analysts	41.03	25.69
Database Administrators	39.77	25.55
Construction Managers	44.50	25.38
Social and Community Service Managers	37.53	24.69
Registered Nurses	31.34	24.56

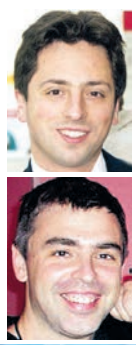
Source: Florida Department of Economic Opportunity

Fastest-growing occupations in Florida

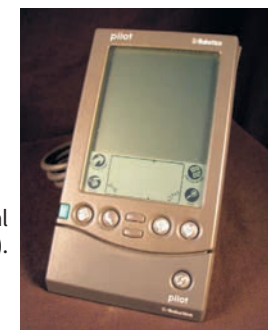
Occupational Title	Annual Percent Growth	Annual Openings
Physical Therapist Assistants	3.72	302
Diagnostic Medical Sonographers	3.33	262
Brickmasons and Blockmasons	3.22	180
Web Developers	3.14	414
Medical Assistants	2.99	2,451
Computer Systems Analysts	2.85	869
Cardiovascular Technologists and Technicians	2.76	206
Information Security Analysts	2.74	163
Cement Masons and Concrete Finishers	2.70	497
Opticians, Dispensing	2.68	300

Source: Florida Department of Economic Opportunity

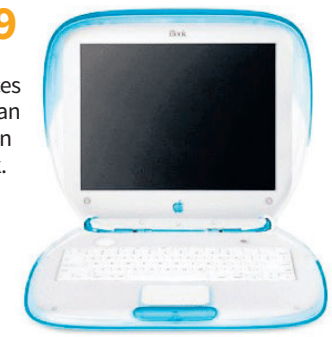
1996
Sergey Brin and Larry Page develop the Google search engine.



1996
US Robotics introduces the Palm Pilot 1000, the first commercially successful personal digital assistant (PDA). It features a stylus input.



1999
Apple introduces Wi-Fi as an option on its iBook.



1999
Research in Motion releases the BlackBerry 850 mobile device, which supports email and web browsing.

An Entrepreneurial Vybe

Going beyond the text

Entrepreneurship

Vybe Software founder Mitch Marchand started his company when he was a sophomore in college. Being an entrepreneur can be incredibly rewarding, but there is a lot that goes into starting a business. One of the key components of a business plan is figuring out how much funding you will need to start your business.

Let's say you'd like to start a website that sells one particular product of your choice. Research the cost of starting such a business. Some of the costs you might need to estimate include: website development and maintenance, website hosting, cybersecurity, wholesale cost of your product and marketing of your product, including search engine optimization (SEO).

Next, estimate what you can expect to make from this business each year for the first three years. How many products will you sell, and what profit can you expect for each? Use this figure to determine a valuation for your business.

Finally, imagine that you are approaching a potential investor for startup funding. How much funding will you request, and what percentage of your company are you willing to give up in exchange?

Mitch Marchand

Founder, president and lead developer

What is your role at Vybe?

"My role at Vybe consists of developing and maintaining our products and ensuring that they continue to grow with our user base. I like to be as hands on as possible with every aspect of the development process. After that, I focus on developing marketing materials, exploring new ideas, communicating with clients and researching for the future."

What is your educational and professional background?

"I graduated from the University of Tampa with a degree in mathematical programming, and started Vybe Software during my sophomore year."

What skills are required for your position?

"Knowledge of programming and application development are essential, of course. Aside from that, I would say that the most important piece to the puzzle is pure ambition – a desire to innovate and share your innovations with the world."

What parts do you find most enjoyable?

"There is nothing like seeing a student's face light up when they've created something with DevKit. To see them enjoying the process and learning through their own experience really affirms the importance of our mission, and it's quite rewarding."

What advice do you have for students considering a career similar to yours?

"From an entrepreneurial perspective, it's all about drive. Believe in yourself and in your ideas, and never stop trying to perfect them. If you're confident in yourself, and you have the ambition to show it, you can be a great entrepreneur.

"From a developer's perspective, understand that technology will continue to grow, and that your skill set will need to grow with it. There will always be something new and exciting to learn, and you should always be prepared for that and be looking forward to what's next."

What skills should students work on to prepare for a position similar to yours?

"Problem-solving skills are essential. Whether aspiring to be a developer, an entrepreneur or both, mastering the art of problem-solving will be crucial to your success."

What classes should students in middle and high school take to prepare for a career in IT?

"Taking a course in DevKit and learning coding concepts and logic can make it much easier to learn to code. Beginning the process early can be very helpful, and I strongly believe that it should begin at the middle school level. JavaScript is an excellent place to start when (one is) ready to begin tackling a language."

2001
Apple releases the Mac OS X operating system.

2001
Microsoft releases the Windows XP operating system.

2002
Research in Motion releases the BlackBerry 5810, the first BlackBerry mobile device with call-making capability.

2003
Research in Motion releases the BlackBerry Quark, the first mobile device with integrated cell phone and email.

Dr. Christopher Maurer

Head of curriculum development

What is your role at Vybe?

"My primary responsibility is to ensure that educators have the tools and resources they need to integrate DevKit into their classrooms to provide an engaging and positive learning experience. This includes writing and updating textbook materials, identifying projects and assignments for students to complete using the DevKit app, and evaluating new features to be included in future versions of DevKit."

What is your educational and professional background?

"I have a bachelor's degree in business administration with a major in management information systems (MIS) and a Ph.D. in MIS from the University of Georgia. I am an assistant professor at the University of Virginia."

What parts of your job do you find most challenging?

"IT is always changing, so it can be very difficult to keep up to date with the constantly evolving technology. I spend a great deal of time reading various sources and speaking with IT professionals to learn about innovative new technologies and how they may be integrated into courses I teach in college and into the DevKit curriculum."

What parts do you find most enjoyable?

"There is nothing better than seeing a student have an 'aha' moment when everything that I have been teaching them starts to make sense. It is rewarding to know that I helped them achieve that goal."

What advice do you have for students considering a career similar to yours?

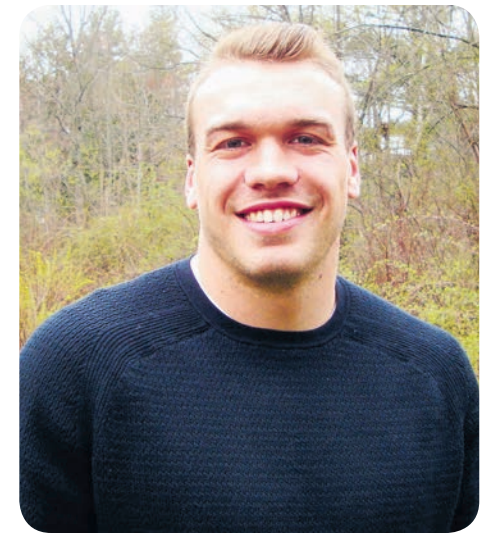
"Never stop being curious and inquisitive. In IT, the most curious people and those who are willing to research new ideas and concepts typically have the greatest success. Whether you are developing new software or trying to protect information systems against cybersecurity threats, there will be a need for continuous learning."

What skills should students work on to prepare for a position similar to yours?

"The ability to break down a complex problem into smaller pieces and then work through those smaller problems until the overall problem is solved."

What classes should students in middle and high school take to prepare for a career in IT?

"Obviously, courses in computer programming and computer networking can be of great benefit to students. Many math courses can greatly help students develop problem-solving skills. English classes are also important, as technology professionals typically need to write documentation, and the ability to write clearly and concisely is necessary."



Erik Martel

Vice president of sales and marketing

What is your role at Vybe?

"I am responsible for researching potential clients, customer acquisition and organization, and implementation of our DevKit summer camps. I am also responsible for managing our social media outlets and other marketing material and channels."

What is your educational and professional background?

"I am a graduate of the Peter T. Paul School of Business and Economics at the University of New Hampshire. I graduated with a bachelor of science degree in business administration with an option in entrepreneurship, along with a minor in economics."

What skills are required for your position?

"I think the most important skills for my position are excellent communication skills; organizational skills; time management skills; adaptability; strategic planning skills; and ambition and drive."

What part of your job do you find most enjoyable?

"The most enjoyable part about working at Vybe is seeing children's excitement during our DevKit Hackathons. Seeing their eyes light up after creating and using their own unique app provides validation for our vision and mission. Furthermore, there is the excitement that comes with building something yourself."

What skills should students work on to prepare for a position similar to yours?

"Students should work on honing their communication and interpersonal skills. In sales, you need to be able to connect and build a relationship with many different types of people with different backgrounds, interests and outlooks. Students should also practice professional communication skills, such as email etiquette and business writing."

What classes should students in middle and high school take to prepare for a career in IT?

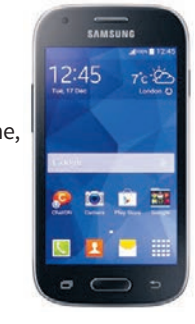
"Students interested in a career in IT should start taking as many computer science classes as their school offers. The more exposure to different types of systems, concepts and technology mediums that students have, the better their overall understanding of IT will be. And, of course, sign up for a DevKit elective (if your school offers it yet) or one of our DevKit Day Camps to get involved in mobile application creation!"

Sources for Timeline: BBC, NASA, the History of Computing Project, history-computer.com, LiveScience, the Tech Museum of Innovation, *World History Encyclopedia*

2007
Apple releases the iPhone smartphone.

2008
Google launches the Chrome Web browser.

2008
The first Android-based smartphone, the G1, is released.



2010
Apple launches the iPad tablet computer.

About Vybe Software

Vybe Software is a web and mobile development company with a focus on innovation in the education space. Since its conception in the summer of 2015, every push made by Vybe has been done in an effort to connect people with technology and information. In the fall of 2016, DevKit was born from that effort. Since then, Vybe has focused all of its attention on the DevKit platform, developing curriculum content and management systems that will get DevKit into the classroom and help inspire young students to pursue careers in computer science.

Vybe will always continue to innovate and grow its impact in the classroom. Its focus is on a desire to connect the young people of today with the tools of tomorrow. For more information about DevKit and all that it can do for your school, visit DevKitApp.com and check out the Classroom Package for educators.

Educators

Share 100 words about how you used this resource in your classroom for a chance to win a $15 gift card! Visit tampabay.com/nie for details and to enter.

About NIE

Tampa Bay Times NIE newspaper in education tampabay.com/nie

The Tampa Bay Times Newspaper in Education program (NIE) is a cooperative effort between schools and the Times Publishing Co. to encourage the use of newspapers in print and electronic form as educational resources – a "living textbook." Our educational resources fall into the category of informational text, a type of nonfiction text. The primary purpose of informational text is to convey information about the natural or social world.

NIE serves educators, students and families by providing schools with class sets of the daily newspaper plus award-winning original educational publications, teacher guides, lesson plans, educator workshops and many more resources – all at no cost to schools, teachers or families. In 2015-2016, NIE provided more than 1.8 million print copies and 10 million digital editions of the *Times* to area classrooms free of charge thanks to our generous subscribers and individual, corporate and foundation sponsors. NIE teaching materials cover a variety of subjects and are aligned to the Florida Standards.

For more information about NIE, visit tampabay.com/nie, call 727-893-8138 or email ordernie@tampabay.com. Follow us on Twitter at Twitter.com/TBTimesNIE.

Florida Standards

Reading this publication and completing its activities incorporate the following Florida Standards for middle and high school students.

Science: SC.68.CS-CC.1.1; SC.68.CS-CC.1.2; SC.68.CS-CC.1.3; SC.68.SC.68.CS-CP.1.2; SC.68.CS-CP.2.1; SC.68.CS-CP.2.2; SC.68.CS-CS.2.2; SC.68.CS-CS.2.5; SC.68.CS-CS.2.6; SC.68.CS-CS.2.8; CS-CS.2.10; SC.68.CS-CS.4.2; SC.68.CS-CS.6.6; SC.68.CS-PC.2.6; SC.68.CS-PC.2.7; SC.68.CS-PC.2.8; SC.68.CS-PC.3.1; SC.912.CS-CC.1.1; SC.912.CS-CC.1.2; SC.912.CS-CC.1.3; SC.912.CS-CP.1.2; SC.912.CS-CP.2.1; SC.912.CS-CP.2.2; SC.912.CS-CS.2.2; SC.912.CS-CS.2.5; SC.912.CS-CS.2.6; SC.912.CS-CS.2.8; SC.912.CS-CS.2.10; SC.912.CS-CS.4.2; SC.912.CS-CS.6.6; SC.912.CS-PC.2.6; SC.912.CS-PC.2.7; SC.912.CS-PC.2.8; SC.912.CS-PC.3.1

Language Arts: LAFS.68.L.1.1; LAFS.68.L.1.2; LAFS.68.L.2.3; LAFS.68.L.3.4; LAFS.68.L.3.5; LAFS.68.L.3.6; LAFS.68.RI.1.1; LAFS.68.RI.1.2; LAFS.68.RI.1.3; LAFS.68.RI.2.4; LAFS.68.RI.2.5; LAFS.68.RI.2.6; LAFS.68.RI.3.7; LAFS.68.RI.3.8; LAFS.68.SL.1.1; LAFS.68.SL.1.2; LAFS.68.SL.1.3; LAFS.68.SL.2.4; LAFS.68.SL.2.5; LAFS.68.SL.2.6; LAFS.68.W.1.1; LAFS.68.W.1.2; LAFS.68.W.1.3; LAFS.68.W.2.4; LAFS.68.W.2.5a; LAFS.68.W.2.6; LAFS.68.W.3.7; LAFS.68.W.3.8; LAFS.68.W.3.9; LAFS.68.W.4.10; LAFS.910.L.1.1; LAFS.910.L.1.2; LAFS.910.L.2.3; LAFS.910.L.3.4; LAFS.910.L.3.5; LAFS.910.L.3.6; LAFS.910.RI.1.1; LAFS.910.RI.1.2; LAFS.910.RI.1.3; LAFS.910.RI.2.4; LAFS.910.RI.2.5; LAFS.910.RI.2.6; LAFS.910.RI.3.7; LAFS.910.RI.3.8; LAFS.910.SL.1.1; LAFS.910.SL.1.2; LAFS.910.SL.1.3; LAFS.910.SL.2.4; LAFS.910.SL.2.5; LAFS.910.SL.2.6; LAFS.910.W.1.1; LAFS.910.W.1.2; LAFS.910.W.1.3; LAFS.910.W.2.4; LAFS.910.W.2.5; LAFS.910.W.2.6; LAFS.910.W.3.7; LAFS.910.W.3.8; LAFS.910.W.3.9; LAFS.910.W.4.10; LAFS.1112.L.1.1; LAFS.1112.L.1.2; LAFS.1112.L.2.3;

LAFS.1112.L.3.4; LAFS.1112.L.3.5; LAFS.1112.L.3.6; LAFS.1112.RI.1.1; LAFS.1112.RI.1.2; LAFS.1112.RI.1.3; LAFS.1112.RI.2.4; LAFS.1112.RI.2.5; LAFS.1112.RI.2.6; LAFS.1112.RI.3.7; LAFS.1112.RI.3.8; LAFS.1112.SL.1.1; LAFS.1112.SL.1.2; LAFS.1112.SL.1.3; LAFS.1112.SL.2.4; LAFS.1112.SL.2.5; LAFS.1112.SL.2.6; LAFS.1112.W.1.1; LAFS.1112.W.1.2; LAFS.1112.W.1.3; LAFS.1112.W.2.4; LAFS.1112.W.2.5; LAFS.1112.W.2.6; LAFS.1112.W.3.7; LAFS.1112.W.3.8; LAFS.1112.W.3.9; LAFS.1112.W.4.10

Newspaper in Education Staff

Jodi Pushkin, manager, jpushkin@tampabay.com
Sue Bedry, development specialist, sbedry@tampabay.com
Noelle Sansom, coordinator, nsansom@tampabay.com

Credits

Written by Sue Bedry, *Times* staff
Curriculum activities by Jodi Pushkin, *Times* staff
Designed by Lisa Schillinger, *Times* staff