Kids, Code, and Computer Science

beanz

2020 Tillywig Award Winner

2020 Academics' Choice Award Winner

Blast Off!

The Heir to Hubble is Launched

June 2022 $6.00

Finding Your Way in the World

In the Middle: All Kinds of Games

What is Permacomputing?

beanz magazine

June 2022: Volume 9 Issue 6
Issue 60 (online) & 43 (print)
ISSN: 2573-3966 (online)
ISSN: 2573-3958 (print)
beanz Magazine© is published bi-monthly, six times a year, online at http://beanzmag.com and in print.
A print + online magazine subscription includes online access to all articles, with links to let you explore topics in more detail.

SUBSCRIBE: visit http://beanzmag.com/subscribe or email us for checks/invoices. We're also available through EBSCO, Discount Magazine, WT Cox, Magazine PTP, and many other subscription services.

ONLINE MAGAZINE ACCESS: send your email address to hello@beanzmag.com and we'll set you up.

Published by Owl Hill Media, LLC
378 Eastwood Rd, Woodmere, NY 11598
Email: hello@beanzmag.com
Phone: (646) 553-3390

POSTMASTER: Send address changes to Owl Hill Media, LLC, 378 Eastwood Rd, Woodmere, NY 11598.
Periodicals postage paid at Woodmere, NY and other mailing offices

Publisher/Editor: Tim Slavin
Staff Writers: Ethan Pate, Amy S. Hansen, Bonnie Roskes, Simon Batt, Clarissa Littler, Jennifer Newell, Tim McGuigan, Bianca Rivera, Jo Hinchcliffe, Tim Slavin
Contributors: Patricia Foster, Erin Winnick, Jay Silver, Jeremy Kubica, Colleen Graves, Jean-Francois Nguyen, Paul Seal, Madeleine Slavin, Joe Strout
Copy Editor: Eileen Seiler
Art Director: Kelley Lanuto
Webmaster: Ethan Pate
Editorial Assistant: Tom Slavin
COVER IMAGE: NASA/CHRIS WEBB, FLICKR

Publisher's Note

Welcome to *beanz*!

It's summer...in the northern hemisphere at least. Time to relax and explore things that might interest you about technology. This issue has a bunch of stories to discover and things to do.

I loved reading the article about the James Webb telescope, which was launched around the time we started to put this issue together. There are also interesting stories about color, permacomputing, bandwidth, Minecraft mods, OS INT and more.

As for things to do, this issue has articles about responsive web design, a Mini Micro project, playing the Geoguesser game, how to use a multimeter, and links to summer projects for SketchUp, 3D printing, electronics, Python, and Scratch.

And if you play video games, the "In the Middle" story this issue is about all the different types of video games. Did you know the first video game was written for an oscilloscope, a scientific instrument? That's a long way from a *Nintendo Switch*. It was fun to write the article with my son, who loves to play games online.

Did you know we have a new website design? Check it out! It features at least one of our readers, Wyatt, in Michigan. We also have started to publish a monthly email newsletter to feature content from back issues. If we don't have your email address, email us at **hello@beanzmag.com** and we'll add you to the list.

Thanks as always for reading!

Tim Slavin, Publisher
beanz Magazine

Our Mission

***beanz* magazine is a bi-monthly online and print magazine about learning to code, computer science, and how we use technology in our daily lives. The magazine includes hard-to-find information, for example, a list of 40+ programming languages for education, coding schools, summer tech camps, and more.**

While the magazine is written to help kids ages 8 and older learn about programming and computer science, many readers and subscribers are parents, teachers, and librarians who use the articles to learn alongside their young kids, students, or library patrons. The magazine strives to provide easy to understand how-to information, with a bit of quirky fun. Subscribers support the magazine. There is no advertising to distract readers.

The magazine explores these topics: Basics of programming and where to learn more, Problem solving and collaboration, Mathematical foundations of computing and computer science, Computational thinking, Recognizing and selecting computer devices, and the Community, global, and ethical impacts of technology.

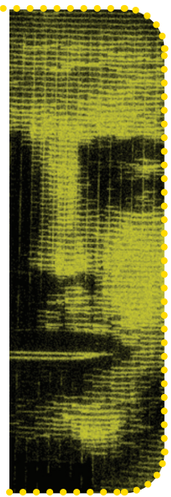
no coding experience
some coding experience
experienced coder

Here is the key for our color coding.

beanz

https://beanzmag.com/subscribe

June 2022



Me and My beanz!

Hi! My name is Jack and I live in the UK. I have been fascinated by coding, computers and technology since I was tiny and finding *beanz* magazine was the best discovery ever! We have nothing like it here in the UK. One day I really hope to work with coding and robotics so I particularly enjoy reading the articles on programming. I learn so much from reading *beanz* and it really inspires me to experiment with new ideas using my Microbit and Raspberry Pi. Thanks *beanz*—you're awesome :)

How much do you love ***beanz***? We'd love to see! Send a photo of you and your ***beanz*** to us at **hello@beanzmag.com**. We may print it right here and make you famous. Bragging rights!

BY CLARISSA LITTLER

Color My World

We don't often think about how color shows up on our computer script. I mean, the computer says "put this color in this pixel" and then it appears and that's the end of it, right?

Actually, no. It's not that simple at all.

We'll get there, but first, let's look at some important things: what are color and light; how do we make color on a screen; and how do programmers represent color in order to give the screen instructions.

Waves of color

Light is a lot like sound: a wave that your body can perceive. Sound is a wave of pressure that shakes the inside of your ear, causing you to hear. How fast it shakes is the frequency of the wave. Low frequencies sound low to our ears and high frequencies sound, well, high.

Light is slightly more complicated. It's still a wave, and the frequency of the wave is what your eyes detect as color: low frequencies are red and high frequencies are blue. When an object has a particular color, it's because the light being emitted from the object is mostly the frequency of that color.

Now, your eye has three kinds of "sensors" in it for color: one for low-frequency light, (reddish and orange colors); one for medium-frequency (yellows and grassy greens); and one for high frequency light (blues and purples). This bit of biological weirdness is going to lead to something really interesting.

The Purple Effect

You may already know that each pixel in any computer or phone screen has red, green, and blue emitters in it. These screens display different colors by changing the proportions of red, green, and blue light emitted in the pixel. The catch is, that if you add red light and blue light together, you just get red and blue light. It's like how playing a chord on an instrument doesn't give you a note in-between the notes of the chord, it gives you multiple notes.

So, why is it that we can mix light from blue and red LEDs to make purple? It goes back to the fact that our eyes have sensors for low, medium, and high. The red light from the LED stimulates the low-sensor, the blue LED stimulates the high sensor, and your brain interprets the proportion between those two as purple. Pretty cool, right?

Now, each different type of screen and each way of making LEDs is going to lead to differences in how you have to mix light together to emulate different colors. And part of making screens is encoding them with information on how to take the request for a particular color and turn it into a mixture of red, green, and blue to give the desired effect.

Notorious RGB & HSV

So, how do programmers even describe colors? There's a bunch of different ways, but the two most notable in coding are RGB and HSV. RGB stands for what you might guess: red, green, and blue. You generally describe each of those components with a number between 0 and 255. That gives a total of 256*256*256 possible colors, which is almost 17 million colors. We can't even distinguish more than 10 million colors.

The big problem with RGB is, if I tell you (200,164,190), can you tell me what color that is? I mean you might be able to make guesses if you're experienced but it's not easy to just look at a color as a triple of numbers and know what your eye will see.

HSV (hue, saturation, value), on the other hand, is a bit more intuitive once you get the hang of it. The kind of color is the hue, expressed as 0-359 degrees, like a circle. A hue of 0 is very pure red, a hue of 120 is a light green, a huge of 240 is a deep blue, and through the purples back to red at 360 which is just 0 again. Saturation is how intense the color is, with 0% being gray and 100% being the most intense and rich version of that color. Value, sometimes called brightness, is how light it is: like if you were shining a bright or dim white light on it. So if I gave you (300,80,50) as an HSV color you could pretty reasonably guess it's a rich but dark reddish-purple.

This is just a first taste of how color works; but there's actually far more to explore in the world of how color spaces are designed, how the human eye responds to color, even in how you have to change the ways you describe colors in RGB to reflect the fact that the human eye has a really complicated response to different intensities of color. It's a fascinatingly complex topic. b

MOURIZAL ZATIVA, UNSPLASH

BY AMY S. HANSEN

Practice Makes Perfect

On December 31, 2021 the James Webb Space Telescope (JWST) started unfurling its five layers of sunshield. The sunshield, about the size of a tennis court, is held out by booms and pulled tight. No one was completely sure the project would work.

"This is the first time we've done this deployment in zero gravity. Every other time we've done it in the clean room, we've had to take gravity into account," Julie Van Campen said on NASA TV. Van Campen is deputy commissioning manager for JWST. There were serious questions about how the sunshield would react. In the end, it opened just as planned and there was cheering from the mission crew. One of the many critical steps was done.

On January 5, 2022 JWST accepted commands from Earth to move its secondary mirror into place.

"That was what I was most nervous about," said Jonathan Gardner, deputy senior project scientist for JWST. "Without that mirror, the observatory would have collected light in the big mirror, but not been able to focus it."

The mirror, like the sunshield, went into place without a hitch.

Hubble Heir

JWST launched on December 25, 2021, riding on an *Ariane 5* rocket supplied by the European Space Agency. JWST is designed to look at ancient galaxies, and is the successor to the *Hubble Space Telescope*, which began gathering images of space in 1990.

For launch, the whole telescope was collapsed down. Getting it to pull its transformer act—changing from a bulky domino-shaped box into its almost flower-shaped finale—has been a nail-biting experience for the mission team and the audience alike. But almost everything went according to plan.

"It's very exciting and kind of amazing," Gardner said, having just watched the latches close on the big mirror, the last of the big pieces of equipment to snap into place on January 8. "We tested it before we launched to make sure it worked, and it did."

Gardner has worked on JWST for almost 20 years and is one of 100 or so members of the mission team. The team worked around the clock for the first 30 days from their headquarters at the Space Telescope Institute in Baltimore, Maryland. The launch and deployment is a process they know well; they've been practicing for the last two years.

Yes, MOM

To do those practices, the JWST team used a giant computer program that simulated everything from launch to full deployment of the telescope. Everyone involved had to learn launch protocol. For instance, the order of deployment is organized by the timeline coordinators. And everyone listens to their *MOM* (Mission Operation Manager). MOM is the one that says when engineers can send commands to the telescope.

And a lot of time is spent sitting around. "Some of the processes go very, very slowly," Gardner said, explaining that heating up motors, for example, can take as much as 18 hours. "And we're always looking for things that could go wrong."

That's where the *RAT* comes in. RAT stands for Rehearsal Anomaly Team. The job of these engineers is to meet secretly and come up with anomalies, or problems that might happen. The RATs then spring these anomalies on the team during rehearsals.

So during a rehearsal with the sunshield one of the motors didn't turn on. The motor was designed for super-cold space, but sometimes motors object to being that cold.

This RAT-created problem "gave the team practice in how to respond when something goes wrong," Gardner said. And the answer for the sunshield would be to switch to the secondary motor electronics.

The other part of the rehearsal is the coordination of the ground crew and their equipment. JWST commands are sent to one of three sites in the world: California, Spain or Australia, where 30-meter antennas can communicate with the deep-space telescope. As the Earth turns, a different antenna rotates into position to point in the right direction. The ground team had to practice the hand-off from one antenna to another.

The team also realized they needed to know what to do should the power go out in Baltimore. In an early rehearsal, the electricity did go out, Gardner said, and the emergency power came on. However, the air conditioning for the computer room wasn't hooked into the emergency power, so computers overheated and that practice came to a stop. From then on, the air conditioning was put on the emergency power system as well.

During the real deployment, however, the team hasn't needed too much of their RAT training, Gardner said. So far, the only anomaly has been in the solar array. "Our solar array was not putting out as much power as we needed," he said. "So we went through the process to tune it." The tuning, he explained, was supposed to get done on Day 30, but was moved up in the schedule so that the sunshield motors had power.

If everything stays on schedule, Gardner says JWST will be set up soon. "We should have the first science images in early July," he said. JWST will start looking at stars and galaxies that haven't even been imagined. b

ALL PHOTOS NASA/CHRIS GUNN

Entering the cleanroom airlock

In cleanroom at launch site in Guiana Space center, French Guiana

Encapsulated in its rocket fairing (protective cone) Ariane 5

BY CLARISSA LITTLER

A Family Heirloom... *Computer?*

WIKIMEDIA COMMONS

How long could a computer last...for a decade, two decades, maybe even a century like an old typewriter or a fountain pen? It's a funny question because almost everything pertaining to how computers are made and sold is about making sure we don't keep them for a decade or two.

I mean, just think of how much games have changed in the last decade: they've gone from a few megabytes (MB) to dozens of gigabytes (GB) for big AAA titles, running on computers that are 10-100 times faster. On one hand, that's cool. Technology advances. Things get shinier. You've got live ray-traced lighting in Minecraft. Awesome. But on the other hand, is it strictly necessary?

Look at a *Chromebook*. It's basically a little laptop that's running a web-browser and, yet, it's kinda sluggish. Websites these days send huge amounts of data back and forth. Not only can you not fit a single game into a few MBs, but you can't even fit the JavaScript code for a lot of websites into that space. We constantly send images that are higher resolution than our screens can even display and they are downloaded in fractions of a second.

In other words, as our computers are getting faster and bigger, are we really gaining anything or is it growth for growth's sake? This "disposability" of computers has real-life consequences for all of us.

Which bring us to the idea of permacomputing, ironically, a rather new field. Permacomputing has more questions than answers for us at this point; it's more of an attitude than a solid set of ideas. But, basically, the concept is to make new computers that can last and make the old ones last as long as possible.

This means designing computers that are more repairable and maintainable, which already eliminates products like iPads that are deliberately made to be hard to crack open. It also means making computers customizable. You should be able to optimise your device for what you need to do and reconfigure it later as your needs evolve. Your computer should grow with you rather than you—or the software you use—outgrowing it.

Imagine a computer that's an heirloom, that you can pass down. Imagine being 50 and seeing a vintage computer and knowing that you could just do a little repair on it and run it just fine. That's the kind of future we're talking about.

It will involve not just changing how we make computer hardware but also changing how we write software. We'll need to be more thoughtful about the resources we're using, more creative in the ways we use what we have. We'll need to get good at doing more with less.

There are people already experimenting with this. Companies like *viznut*, that are involved in the size-coding/demoscene world where people write fully functioning graphics and music programs in only a few hundred characters of code... not lines, characters. Also, 100 rabbits, the duo who made ORCA, (see the February 2022 issue of *beanz*), has started making all their art projects on a virtual computer called UXN that's so small and efficient it should be able to run on almost any hardware.

All the people who intentionally restore old computers or figure out how to live on nothing more than a Raspberry Pi are all learning how to work with software in new ways. While none of these things, per se, is a way of achieving permacomputing, they are all a part of the experiment as we collectively feel out the path forward.

BY SIMON BATT

May I Borrow a Cup of Bandwidth?

We all love our convenient internet-connected devices, except when the connection goes down and these devices become little more than expensive paperweights. The key, then, is to ensure you always have a connection, even if it's not your own.

One way to keep connected is to install a 4G or 5G chip into your device, but it is very expensive for both the manufacturer and the user to maintain an emergency cellular data plan.

There is another way. When your internet is down, there's a good chance that your neighbor's internet is not. Perhaps they use another service or their connection is more stable than yours. So why not use their network instead to keep your IoT devices up and running?

This is the goal of *Amazon Sidewalk*, a special program that allows you to connect your Amazon IoT devices to others in your neighborhood, and theirs to yours. This creates what's called a "mesh network" that many devices can piggyback off of to get internet access.

Usually, mesh networks are restricted to the devices in your home. However, Amazon Sidewalk wants to make it so neighbors can connect to neighbors, who can then connect to their neighbor's neighbors, and so on. The idea is that entire neighborhoods can connect to one another and help each other out when one or more connections go down. And while the bandwidth isn't enough to bring people's laptops, computers, and phones online, it's enough for Amazon's IoT devices.

Unfortunately, this opens up some privacy issues. Some people won't like the idea of other people connecting to their router and probably will turn off the Sidewalk feature. And for every person who turns off Sidewalk, the mesh network gets weaker.

That being said, Amazon Sidewalk is an interesting insight into how IoT will progress and share data in the future. Will people allow other IoT devices to connect to their network? Or will privacy concerns prove too much for this plan to take off?

It's a tough dilemma and IoT developers are still trying to find that perfect middle-ground that will satisfy the majority. Until then, community-created mesh networks will always be hit-or-miss among the public. b

INDIANA UNIVERSITY ARCHIVES/INDIANA UNIVERSITY BOARD OF TRUSTEES

BY SIMON BATT

Back to the Stone Age

WIKIMEDIA COMMONS

Minecraft mods tend to cover only one aspect of technology each, such as electricity, or space rockets for interstellar travel. But imagine a modpack that starts from the beginning of technology. You work your way up to the modern era...and beyond.

Introducing *SevTech Ages*, a module that contains six ages in total, ranging from the Stone Age all the way to a Futuristic Age.

The Stone Age acts as the tutorial for your adventures, and once completed, allows you to progress to the Bronze Age. To advance through an age, you have to craft a specific item or achieve a specific feat that increases your technical skills to the next era of science.

For example, you'll learn very quickly that you can't mine for iron or diamonds in the Bronze Age because, much like the real-world bronze age, your character's scientific prowess isn't developed enough to understand what iron and diamonds are, and how to use them. So you make do with the ores and metals you CAN use in that age, using them to build toward the final goal of that time period. Once completed, you advance to the next age and unlock certain ores and resources as a reward.

But don't think you're the only one advancing. As you get stronger and more knowledgeable about the world around you, so do the monsters that inhabit that world.

During the stone age, your main threats are skeletons and zombies wielding sharp bone weapons and wearing leather armor. Not too much of a threat, but remember; you have no iron at your disposal, so you'll be on a level playing field with them. By the time you hit the futuristic age, you'll be facing down charged-up creepers, boss monsters, and enemies equipped with diamond gear. Put that new tech to good use.

SevTech has it all for the technologically curious. Machines, space rockets, and even planetary exploration; and if you really want to, you can push the limits of the futuristic age and unlock a bonus tier that lets you play with creative mode-level tools.

SevTech Ages is available on *Curseforge* so be sure to give it a try if this piques your interest. b

BY ETHAN PATE

Where in the World Am I?

Are you a geography nut? I know I am, and so when I learned about *GeoGuessr,* I was obsessed pretty quickly. GeoGuessr (**geoguessr.com**) is a geography-based guessing game that tests your critical thinking and world geography skills all at the same time.

In case you haven't played before, it works by using images from *Google Street Views* to place you on a random street or pathway somewhere in the world. You then get to look around the location to try and gather clues and guess where you are. It is sort of like looking at old photographs from when you were younger and trying to place where they were taken...but on a global level.

It can be pretty fun, but it is also super challenging. Lucky for you, I have a few tips to help improve your GeoGuessr score.

Which way do the shadows go?

Did you know that if the image was taken during the summer months you can figure out which hemisphere you are in? To use this trick, look at the compass icon at the top of your screen and orientate yourself so you're facing north.

Now look around and see if the shadows in the picture are pointing north as well. If they are, that means the sun is shining from the south and that you most likely are somewhere in the northern hemisphere. And if the shadows are pointing south, you are most likely in the southern hemisphere.

Check out the Satellite dishes

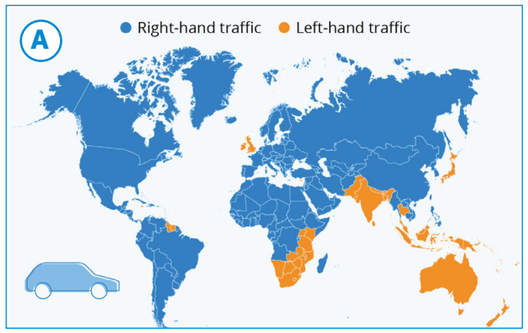
But what if it's a cloudy day and there aren't any shadows? If this is the case, I like to use any visible satellite dishes to my advantage. Because almost all dishes point towards satellites orbiting the equator, these dishes can be pretty useful as stand-ins for the shadows.

If the satellite dish is pointing south, then you are most likely in the northern hemisphere, and if it is pointing north then you are most likely in the southern hemisphere. Even easier is if a satellite dish is pointing straight up. This almost always means you are very close to the equator.

What do the buildings look like?

Now that we have an idea of which half of the world we are in, we can get a little more detailed. Looking at the types of houses is a great next step. This is because the design of houses can vary greatly depending on the region and climate.

For example, houses in very warm places are often painted white to reflect heat. While houses in very wet places usually have very sloped roofs because of the large amounts of rain.



What Vehicles Do You See?

It is always a good idea to check what side of the road people are driving on as this can be a great clue to the country you are in. **A**

Another thing to keep in mind is the type of vehicles you see. If you see just a few scooters and motorbikes there is a good chance you are somewhere in western or central Europe. But if you see a ton of scooters and other small engine bikes you are most likely somewhere in Africa, Southeast Asia, or South America.

Know the Limits of the Game

The last tip I have for you is to familiarize yourself with the limited possible locations you can be dropped into. Because the game uses Google Street View, you can only be put in locations that exist on Google Street View. If you look at the map below, you will see that Google Street View does not have global coverage yet. **B**

So that once you have answered all the questions above, ask yourself: "Is my guess actually possible?" If it is, then drop your pin. If not, see if there is a country that would fit the evidence you have gathered.

While there is a lot more to this game than just the tips I have provided, such as learning how to identify languages, different kinds of traffic signs, and even distinct types of vegetation, the tips above should help you start pumping up your scores. Happy geo-guessing. **b**

SUBHASH NUSETTI UNSPLASH

BY BIANCA RIVERA

Responsive Web Design Using Grid View

In the April 2022 article, *Responsive Web Design Using Trinket,* we learned how to make our websites adapt to various screen sizes. With just a few tweaks, website elements could be responsive to different devices by setting the viewport, using the **max-width** property for images, using the **vw** unit of measure for text, and adding a percentage value for padding-top when using Iframes.

We can go a step further and control where elements are on the page depending on screen width. This is called *Grid-View* and the idea is that each web page can be divided into 12 equal columns. A total web page is 100% of the view and if we divide that web page into 12 equal columns, each column is 8.33% of the total page size (100% / 12 columns = 8.33%).

With that in mind, we can place elements where we want by defining which columns they should be in. We'll do all of this in the CSS style sheet. To follow along with this project, visit the project page on Trinket (**https://trinket.io/html/4be0cef67e)**.

To start, we need to ensure that every element on the web page has the box-sizing property set to **border-box** (as seen on line 1 of the project page style.css). Next, we need to create a class that defines the width of each column (in this project, it's **class*="dcol-"**). Each column will be 1/12 larger than the next column. **A**

Next, we can use the class we made to style our HTML. To start using the grid-view, we need to create a class called **row** and, within that, insert the desktop columns to arrange our content. For example, the navigation menu in the project is set within two classes—row and **dcol-3** (**dcol-3** has a width of 25% of the screen). This allows the menu to always sit in the first column of each web page it's on. **B**

This is great for desktop viewing, but as mobile viewing is becoming more prevalent, we want the site to display with "mobile first" in mind. We can use a CSS media query to check if the user is on a small screen first and, if so, the width of the columns will be 100% (with the content running lengthwise). If the user is viewing with a screen size of 768px or more, the page will display using the column widths that were established.

We can experiment with changing where elements appear on a page. On the home page, we have 3 columns (the menu is 25% of the width, the text is 50% of the width and the sidebar is 25% of the width). We could change the layout to only 2 columns by keeping the menu at 25% and changing the text to 75% of the page. We could push the side bar onto the next row. It's all up to you and by using grid-view, anything is possible.

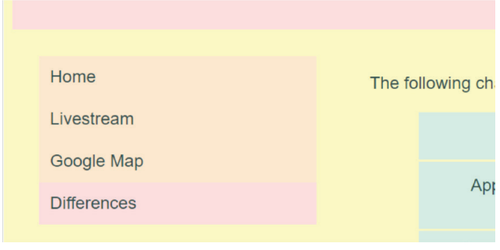
A

```
/* For desktops: */
[class*="dcol-"] {
  float: left;
  padding:2%;
  width: 100%;
}

/* For desktops: */
@media only screen and (min-width: 768px) {
  .dcol-1 {width: 8.33%;}
  .dcol-2 {width: 16.66%;}
  .dcol-3 {width: 25%;}
  .dcol-4 {width: 33.33%;}
  .dcol-5 {width: 41.66%;}
  .dcol-6 {width: 50%;}
  .dcol-7 {width: 58.33%;}
  .dcol-8 {width: 66.66%;}
  .dcol-9 {width: 75%;}
  .dcol-10 {width: 83.33%;}
  .dcol-11 {width: 91.66%;}
  .dcol-12 {width: 100%;}
}
```

B

```
<div class="row">
  <div class="dcol-3">
    <nav>
      <ul class="nav">
        <li><a class="active" href="index.html">Home</a></li
        <li><a href="livestream.html">Livestream</a></li>
        <li><a href="map.html">Google Map</a></li>
       <li><a href="differences.html">Differences</a></li>
      </ul>
    </nav>
    <img src="chick.jpg" alt="Northern albatross chick and a
  </div>
```



BY JOE STOUT

An Apple a Day...

. . . keeps space travelers healthy.

The fearless starship crew has been in space for months. Forced to subsist on nothing but bread and Plutonian rock slime, they have begun to suffer from a serious Vitamin C deficiency. Fortunately they make it to the Apple Nebula, where nature's bounty awaits. Gather as many apples as you can before the time is up!

In recent issues, we covered how to install a virtual computer called *Mini Micro* and run games. You can always find it at: **https://miniscript.org/MiniMicro**. This time, let's create a real game you can play with the keyboard or a game controller.

Launch *Mini Micro*, use the **edit** command to open the code editor, and then carefully type in **Listing 1**. Click the **Close & Run** button at the top of the editor, or exit the editor and type **run** at the prompt, to give it a try. If you get any errors, click **edit** again and double-check your typing, being sure to match the capitalization and punctuation shown. If all is working correctly, you should be able to fly your spaceship around the bottom of the screen, collecting apples that fall from the top. How many can you catch in 30 seconds?

```
1  import "mathUtil"
2  clear
3  ship = new Sprite
4  ship.image = file.loadImage("/sys/pics/Spaceship.png")
5  ship.x = 320; ship.y = 100; ship.rotation = 90
6  display(4).sprites.push ship
7
8  target = new Sprite
9  target.image = file.loadImage("/sys/pics/food/Apple.png")
10 display(4).sprites.push target
11
12 snd = file.loadSound("/sys/sounds/pickup.wav")
13 score = 0
14 endTime = time + 30
15
16 // main loop
17 while time < endTime
18       yield
19       text.row = 25
20       print "Time: " + ceil(endTime-time) + "  Score: " + score + " "
21       ship.x = ship.x + key.axis("Horizontal") * 10
22       ship.y = ship.y + key.axis("Vertical") * 5
23       if ship.y > 200 then ship.y = 200
24
25       if mathUtil.distance(ship, target) < 40 then
26             snd.play
27             score = score + 1
28             target.y = -100
29       end if
30
31       target.y = target.y - 10
32       if target.y < -32 then
33             target.y = 700
34             target.x = 30 + 900 * rnd
35       end if
36 end while
```

This program illustrates some very common game elements. The objects *ship* and *target* are "sprites"—little pictures that can be efficiently moved around the screen. These are set up near the top of the program, in lines 3-10. Lines 12-14 load a sound, clear the score, and define the game end time.

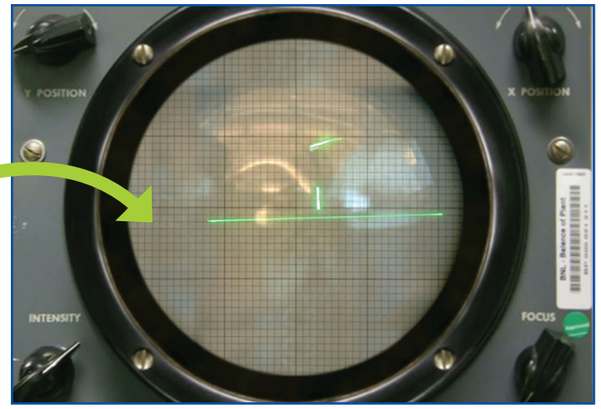
Next comes the main loop (lines 17-36). The main loop is the part of the program that executes over and over, about 60 times every second. This prints the current time and score at the top of the screen, and then moves the ship using the **key.axis** method. This is a great way to move a sprite because it works with the arrow keys, or the WASD keys, or even a gamepad or joystick.

Line 25 checks whether the ship and target (apple) are close enough to catch. And finally, the target is moved on line 31. If it goes off the bottom of the screen, the **if** block on lines 32-35 will move it back up to the top.

Want to take this program further? Try giving a little extra time whenever an apple is caught (hint: you'll need to add to the **endTime** variable). Or try changing the score so that instead of 1 point per apple, you get more points for catching it higher up (hint: **ship.y** is the ship's distance from the bottom of the screen). Or, copy the code from **/sys/demo/stars.ms** and insert it before line 3, to give the game a colorful starry background. Happy coding.

BY TIM SLAVIN AND TOM SLAVIN

It Takes All Kinds!



WIKIMEDIA COMMONS

There are so many kinds of video games out there, both old and new. Let's take a look at some. The first video game, *Tennis for Two*, was created to amuse people at a party at Brookhaven National Laboratory, on Long Island, NY in October 1958. The game was played on an oscilloscope, a scientific instrument, and looked a lot like Pong with blips of light that could be moved back and forth.

In the 1970s and 1980s, early video games anyone could play were found in arcades, large rooms with standalone games like *Asteroids, Tetris, Donkey Kong*, and many others. These video games looked like early mechanical pinball games with flippers you controlled to shoot small balls up and around a board to score points. Unlike modern games, early video games were either score-based or limited you to a small number of lives. You needed lots of change to keep playing when your character died or the game got too fast and you lost.

Modern video games have some similarities to early arcade games but the vast range of types of games is amazing. You can play by yourself or with friends. And some games are immersive experiences, either puzzles like *Myst* where you wander around an empty world trying to figure out what happened, or adventure games like *Legend of Zelda*. I remember being stunned watching my son's game character sitting on a horse looking down on Jerusalem in the 1100s when he played an early *Assassin's Creed* game on our large TV. It was extremely realistic and beautiful with air full of dust, the buildings, and people walking past.

Here are a few of the many types of games available. Perhaps find out about a few you haven't heard of and play with friends and family. The online version of this article has links to many more types of games.

Life Simulation

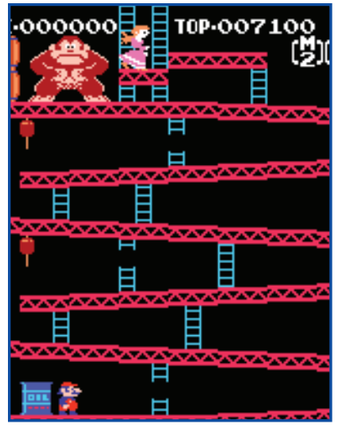
Animal Crossing, a life simulation game, happens within a day with new people and content. Players return for short intervals to play and complete the game. *Tamagotchi* is similar because you feed your digital pet on a regular schedule. *Harvest Moon* and *Stardew Valley* are role playing games like *Pokemon*. In *Harvest Moon*, you take over a farm and gameplay happens around the farm. In *Pokemon*, you're a 10 year old child collecting animals to engage in contests to earn badges.

Animal Crossing

Platformer

Mario is a platformer game with running, climbing, and jumping as you explore and work your way through many levels. Like early arcade games, your character can die and that sends you backwards in the game. But you also can learn how to beat each level. Other examples of platformers are *Donkey Kong, Sonic, Crash Bandicoot*, and *Kirby*. NES stands for Nintendo Entertainment System.

NES Super Mario Bros.



NES Donkey Kong

Third Person/ Battle Royale

The popular *Fortnite* game is a kind of Third Person/Battle Royale game.drops players from the sky onto a map and the goal is to be the last one alive. You're in an open world, like *Minecraft*, with resources to craft tools and shelter to survive as long as possible.

Fortnite

Sports

Some modern video games provide sports experiences. You can play as your favorite teams and players. EA Sports offers the games *NBA, Madden, FIFA, NHL, PGA Tour*, and F1 video games. If you want to take your gaming to a whole new level, Sports video games are a way to get into e*sports*, which are video games that are played in a highly organized competitive environment.

EA Sports Madden

EA Sports PGA Tour

Sandbox/ Survival

Minecraft is an example of a Sandbox/Survival game. Players are dropped into a world they can explore, gathering resources to craft new things to survive in a sometimes hostile environment, especially at night or in dark places like caves. Players also can build houses and complicated structures for the joy of building. The key to sandbox/ survival games is managing your resources.

Minecraft

BY CLARISSA LITTLER

Dithering Down on Images

There's an old trick for enhancing computer images that was once done out of necessity, but has found new importance in today's permacomputing culture... and honestly it's a cool aesthetic, too. Welcome to *dithering*.

This funny sounding word that means trembling or quivering is a way to make images use fewer colors while appearing to have more. Originally, older computer monitors were very limited in the number of colors they could display, as opposed to the over 16 million we can display on most monitors today, So there was a need to make photos look as good as possible with those limitations. (For more on the nature of color and computers, check out *Color My World* in this issue, page 2.)

The basic idea of dithering is that you use fewer colors but then mix the pixels up to give the impression of different colors. For example, the image below left (**https://commons.wikimedia.org/wiki/File:Black-and-white_photo_made_with_infrared_photography.jpg**) is a monochrome photo with lots of rich gray tones.

And below right, the same image is dithered so the only two colors are black and white.

It still gives the impression of grays though, right? That's because, if we intermix white and black pixels next to each other, our eye gets the vibe of different shades of gray even if, on some level, you know it's just black and white. Aside from the cool aesthetic, dithering has really practical uses, because it makes images so much smaller. If you take a photo with a smartphone, the image will often be several megabytes (MB) in size. If you take a photo with a nice standalone camera, the image could be dozens of megabytes. For our modern storage that might not be huge, but what if you're concerned about permacomputing and want to make a solar-powered image server that you can keep adding photos to for a decade? Then the difference between a 10MB image and the 100 kilobytes (KB) dithered version is pretty huge.

But how does dithering make images smaller? We've talked about compression in a past article but we'll just briefly review the idea here. Most file formats for images, like jpeg, compress the image by taking redundant data, like say 20 pixels in a row that are all black, and instead of storing 20 pixels worth of data, it stores the number 20 and then the color black. That's a little oversimplified but gives you the right idea for how image compression works.

If you reduce the number of different colors in an image then it's going to be more compressible.

There's a bunch of different dithering algorithms, but the basic idea is the same. First, you choose a set of colors you want to keep in the final image. In the case above, that's just black and white. Then you go through the image pixel by pixel and, for each pixel, you assign a final color closest to its original color. Then, and this is the important step, you take the difference between the original color of that pixel and its final color and use that to tweak the other pixels around it that you haven't converted yet. So, if you assign a pixel black, then make the pixels near it a little lighter. If you assign a pixel white, make the others a little darker. This process gives us that intermixing of white and black pixels to make gray tones.

If you want to play more with dithering on your own try the Dither It! website (**https://ditherit.com/**) or try implementing your own from linked articles. Either way, try it and see what cool things you can make. b

BY TIM SLAVIN

How I Spent My Summer Vacation

It's summer in much of the world, and that means no school. Lots of time to play and explore new things. But what if you get a case of the "I'm boreds"? Before your mom puts you to work washing windows, check out what this family has to say. It looks like they are re-thinking their summer camping trip. Perhaps you can pick up some tips on some STEAM projects. Just think how much you will have to write about when you get back to school next year for your "How I Spent My Summer" assignment.

Ideas:
https://www.instructables.com

Mycroft:
https://mycroft.ai/get-started/.

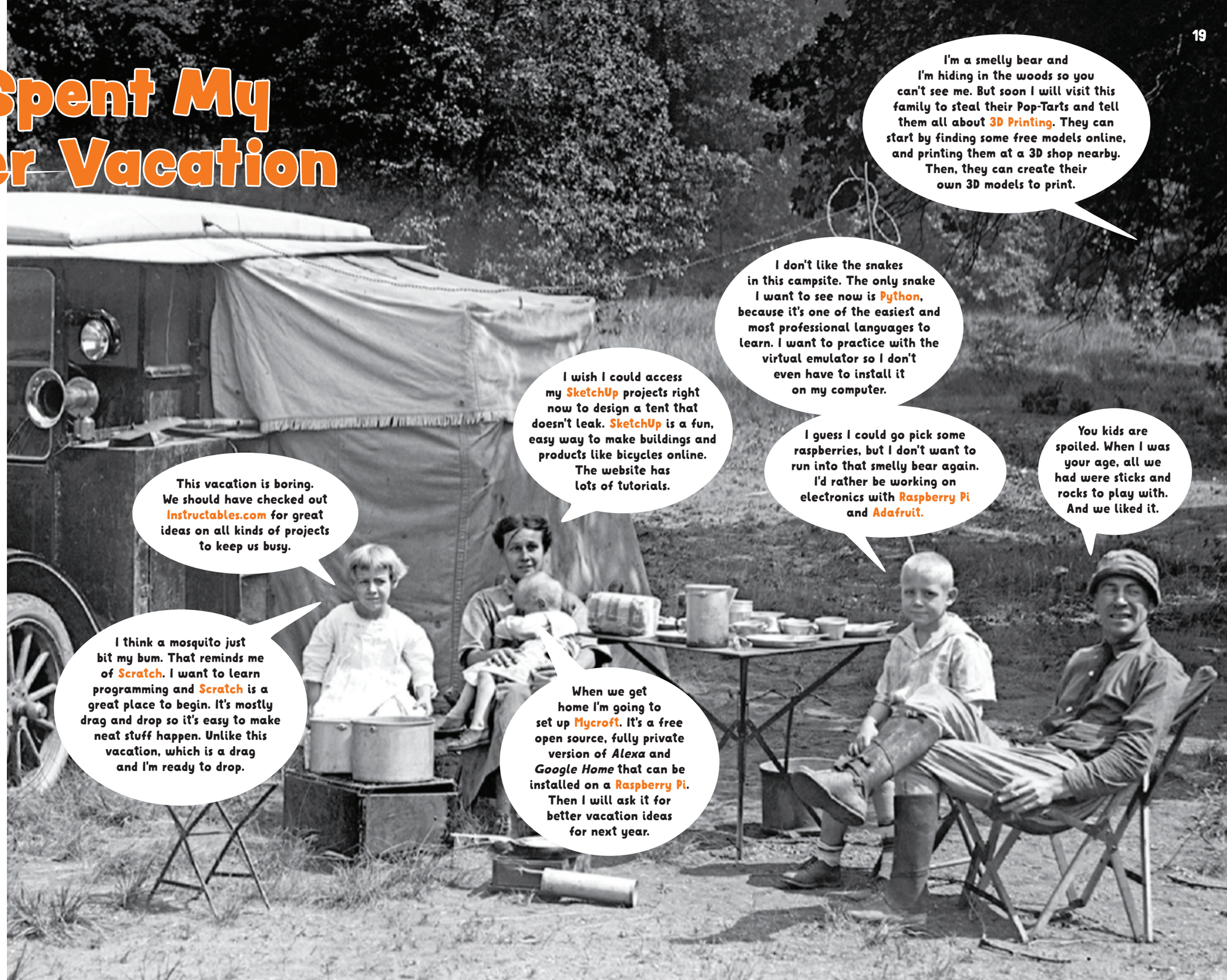
Python:
https://www.python.org/
https://realpython.com/start-here/
https://learnpythonthehardway.org/python3/

Scratch:
https://scratch.mit.edu/

SketchUp:
https://www.sketchup.com/try-sketchup#for-personal
https://kidscodecs.com/sections/sketchup/
https://3dwarehouse.sketchup.com/

Electronics:
https://hackspace.raspberrypi.com/
https://magpi.raspberrypi.com/
https://www.adafruit.com/
https://make.techwillsaveus.com/

3D Printing:
https://www.thingiverse.com/
https://all3dp.com/2/3d-printing-for-beginners-all-you-need-to-know-to-get-started/
https://3dprintingindustry.com/3d-printing-basics-free-beginners-guide



BY JO HINCHLIFFE

The Many Uses of Digital Multimeters

The great thing about electronics projects is you can learn a huge amount and make lots of interesting circuits with inexpensive components and very few tools. As you develop, you might think that you need a fully equipped laboratory with all kinds of specialized equipment, but you can do a lot with very little. One item that's extremely handy (in addition to pliers, wire cutters and screwdriver) is a budget digital multimeter.

Digital Multimeters (often referred to as DMMs) can take a huge variety of measurements; in fact more than we can cover right now. So, let's look at just some of the ways a DMM can be useful in your project.

Most DMMs have two probes plugged into sockets, one black and one red, signifying positive and negative. **A** Most of the time your probes will be connected to two sockets that are marked XXX which means that the DMM is set up to measure voltages and other items (but not current, which requires the DMM to be connected to circuits in a different fashion. More on that later).

A really common use for multimeters in electronics projects is measuring DC voltage. **B** Say you have connected up a breadboard experiment and then hooked up a battery, but the experiment hasn't powered up. Your first thought might be to check the battery. Setting the multimeter to DC mode and holding the probe tips to either end of the battery will show the voltage, which may indicate that your battery needs replacing. Likewise, you might use the DC voltage setting to probe different parts of your circuit to look at the voltage in different areas. Keep the black probe connected to the negative end of a battery and then use the red positive probe to explore the voltage.

DC voltage measurements can sometimes have other uses such as checking the polarity of items. A good example is that some batteries come without distinguishing features at either end and it can be difficult to work out which end is which. If you are measuring the voltage of such a battery, say a 18650 cell with no marking, you can connect the positive probe to an unknown end of the cell and if it turns out to be the negative end, the battery voltage will be presented as a negative value. This will still give the correct numerical value of the voltage and tell you that the probes are connected back to front, so you can correctly identify the polarity of the battery.

If you have tinkered with electronics, you will have come

continued on page 22

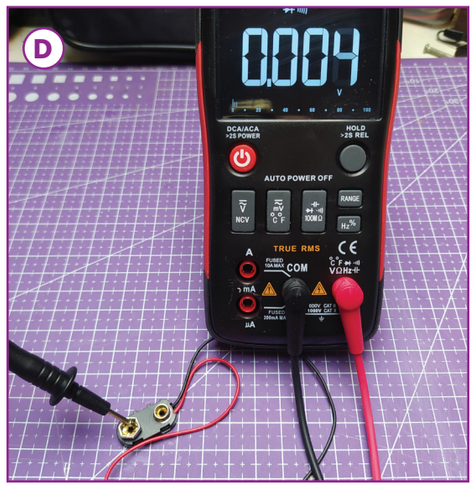
continued from page 21

across the use of resistors. **C** Resistors have stripes on them of differing colors, which you can decode the value of the resistor in ohms. This is a fun activity and worth learning how to do; but it can take time if you have a project with a lot of different resistors to sort out. Digital multimeters have the ability to measure resistance. Some are "automatic ranging", so you can connect the probes to either end of a resistor of any value and it will range-in to give you the correct value. Non-auto-range DMMs have separate settings that you have to switch often for each range of resistance value, so you will need to roughly know if the resistance is just a few ohms or a few million ohms. Auto ranging DMM are now very common and available at the budget end of multimeters. So, if it's your first meter, be sure to look for that feature.

Another very useful feature on most multimeters is continuity checking. **D** When selected, this feature means that the unit will bleep whenever there is a connection between the probes, which you can check by simply touching the metal probe tips together. Continuity checking is super useful for a variety of reasons like checking the wires in your project to make sure that they aren't broken inside the insulation. Using a continuity checker, you can identify connections. For example, when using a PP3 battery clip, the checker can determine which battery connector connects to positive and which to negative. Another great use for continuity checking is detecting solder joints that shouldn't be connected as in the case of tiny solder "bridges" that are created between two solder pads.

Finally you can swap the probes into other sockets enabling the multimeter to read the current usage of a circuit. **E** In this mode, you make the DMM part of the circuit, so that it has the current flowing through it - as if the DMM and it's probes are a single piece of wire. When you set the DMM to current mode, it measures and displays the current use on the screen. You need to be careful that you don't accidentally connect the DMM in a way that can create a short circuit. And, remember to change the probes before you measure any voltage as that can produce a short.

As a safety feature, most DMMs are fitted with fuses in case of a short and while many multimeters are rated for mains electricity work, we recommend only using a multimeter in battery and low power projects. **b**



Print That!

BY TIM SLAVIN

When someone asks, "can you print this?" do you think about using a computer to print something on a piece of paper? Most people do. But there's another kind of printing. It's used in software programming and its history goes back to the last century and last millennium. It's not as old as dinosaurs, but it is old.

Early computers output their data on punch cards after calculations were made. Eventually simple printers were attached instead of punch cards. Then computer monitors appeared in the 1970s and programmers had to find a word and concept for displaying output on their consoles and screens, not a printer. People called that "printing to the screen". Today most people don't realize viewing a web page on a computer screen really is printing the web page to their screen. We think of printing as separate from our computer screen.

Programming something and then asking the computer to show it on the screen is still called printing. You can learn to print to your screen with a software programming language. In fact, printing is the first basic skill to learn for most languages.

Go to **Python.org**, the main website for the Python programming language, and you'll find a virtual emulator, image **A**, below. This lets you type Python code and see results, without having to bother installing the Python language. Click the yellow **>** button in the interpreter to clear out the screen and load Python. Then type this command carefully:

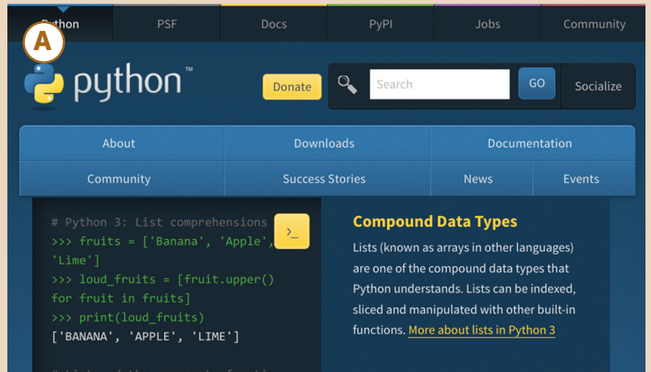
print("Hello World!")

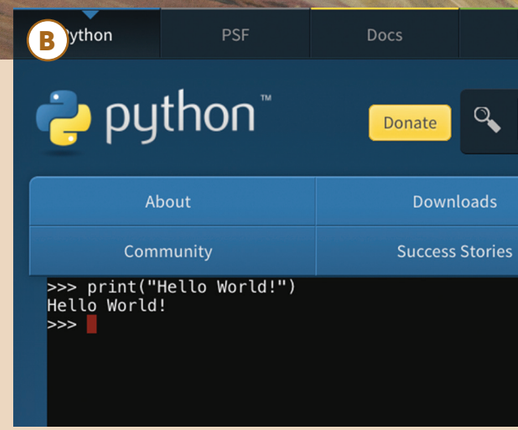
You'll see the correct output, image **B**, below. **Type Ctrl + L** to clear the screen, if that's useful.

Now type this command:

print("I'm a monkey")

Congratulations, you've now learned another meaning of "print". And, if you don't know Python, you've done a simple programming exercise in that language. **b**





BY ETHAN PATE

Move Over, James Bond. OSINT Is Here.

At some point in our lives, many of us have dreamed of being super spies. For me, most of my sneaky activities have involved getting an extra cookie in the middle of the night. But there are some actual ways you can feel like a super spy. One of the coolest is called *Open Source Intelligence*.

First things first: what is '*intelligence*'? While it may make sense to think intelligence as being smart, that is not totally true. Intelligence in this context means the activity of collecting, studying, and sharing information to help people make plans for the future, or better understand the present. The way I think about intelligence is that while reading a newspaper can tell you what happened in the past, intelligence can tell you about what will happen in the future.

Open Source Intelligence (OSINT) is an information-gathering tool that focuses on publicly available information. This means it uses everything from newspapers to social media to satellite images to get information about a question.

The coolest part of OSINT is that you can use it for so many different things. For example, it has become an important tool for journalists and people looking to fact-check information. A fun example of this can be seen in the website *Snopes.com,* investigating whether or not Goofy, the beloved *Disney* cartoon character is a dog or a cow.

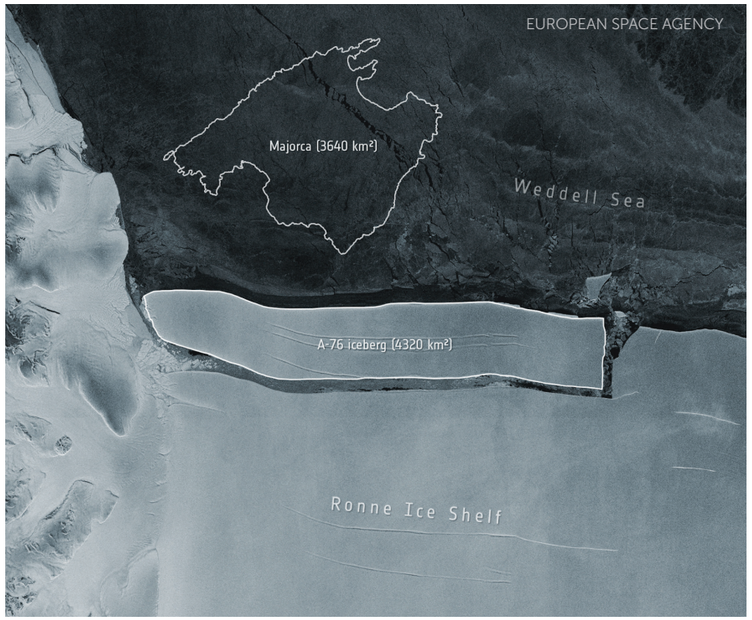
Another cool example is how OSINT can be used to better understand icebergs being created in the Antarctic Ocean. In May 2021, Iceberg A-76 broke off from Antarctica. Iceberg A-76 was the world's biggest at the time it was formed, being 110 miles long and 16 miles wide. Many wondered if the iceberg broke off because of climate change (a topic discussed in an article from a previous issue) or if it was a natural separation.

This incident brought out some amateur OSINT investigators who were able to watch how A-76 broke away from the main ice shelf. They did this by first looking at satellite images of the area the iceberg came from before its creation. They then used images captured by *Sentinel-1*, a satellite that is part of the *Copernicus Programme* satellite constellation to watch the actual breakaway.

They then compared how this iceberg broke up to the way another iceberg had naturally broken off the same ice shelf to see if the break pattern was similar.

The OSINT investigators then used this information in combination with the opinions of various iceberg experts found on Twitter and in newspaper reports to conclude the break was natural and would not harm any ships or coasts while it melted in the sea.

OSINT is a super cool tool that more and more people are using to verify information. The best part is, you don't need to be an expert in any of it to get started; you learn as you go.



Time-lapse of A-76 from May 6, 2021, to June 6, 2021, via satellite imagery from Sentinel-1

BY SIMON BATT

Give It the BOOT!

Have you seen videos where someone sets off a chain reaction of dominos that cascade off of one another and reveal a picture at the end? Did you know that a computer operating system works on a similar principle?

When an operating system starts up, it has to load all the little program sequences that keep everything running. One program will load a bigger program, which in turn will load an even bigger program in a chain that continues until the operating system is ready to go.

But what starts the whole process? When you turn on the PC, its RAM (Random Access Memory) is empty. There's nothing there to actually launch the operating system loading process. You can have the user run the code themselves, but ideally you want the PC to load automatically. You need a "bootstrapper".

A bootstrapper is a tiny chunk of code that lives in ROM (Read Only Memory) pre-installed on the motherboard of your computer. Unlike RAM, ROM cannot not be changed by the user. When you power up your PC, this tiny piece of code sets off a chain of events that helps build the operating system, then goes back to sleep. It starts off with compiling the programming language, which in turn, sets up compilers for other programming languages, which then compile and run all the features in your operating system.

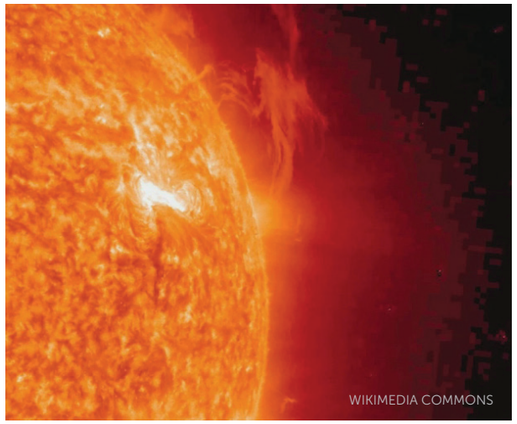
The job of the bootstrapper seems small at first glance, but much like domino art, its initial action blossoms into something much greater.

But why is it called a "bootstrapper"? The term originates from the phrase "to pull yourself up from your own bootstraps," which means to do something without any external help. And we call this code the "bootstrapper" because it helps the PC boot without the user needing to manually run code.

So the next time you push the power button on your computer, think about that tiny "seed" of code that helps everything else load without you needing to do a thing. Computers are really amazing when you look at them closely.

LYNNELLE CLEVELAND, UNSPLASH

tidbitz



Space Weather...

We know the weather here on earth: maybe it's sunny outside, or raining. What about out in space? Turns out space does have weather, much of it created by our sun shooting flares. The sun may be 93 million miles from Earth but its weather can damage our satellites and electronic devices. The US National Weather Service has a space weather page that includes links to NASA solar missions. b

https://www.weather.gov/phi/spacewx

https://soho.nascom.nasa.gov/spaceweather/

https://www.nasa.gov/subject/3165/space-weather/

Cloud Computing's Carbon Footprint


EPA.USA.GOV, WIKIMEDIA COMMONS

Computers are everywhere in our daily lives. Not only laptops and desktop computers but also phones, traffic cameras, and other devices. Many are connected to the internet and send data to and from other computers that collect and serve data, for example, collecting video from an intersection with heavy traffic. Turns out our digital world with all these computers has a huge impact on the ecology and climate. This computer infrastructure now has a bigger carbon footprint than airlines. b

https://thereader.mitpress.mit.edu/the-staggering-ecological-impacts-of-computation-and-the-cloud/

Totally Rocking It

A planned Taiwanese building project, called a *Sun Rock*, is designed to use solar panels to generate massive amounts of power. The building is almost totally covered with solar panels and capable of generating roughly 1 million kWh of energy a year. The building has a round rock-like shape to make the most of the sunlight falling on the building. b

https://www.inceptivemind.com/solar-panel-covered-sun-rock-generate-1-million-kwh-clean-energy-per-year/22989/


IMAGE FROM INCEPTIVEMIND.COM

KGROOVY, FLICKR

The 4004 at 50

Believe it or not, there was a time when personal computers did not exist. In November 1971, Intel released the 4004 computer chip which was the first highly usable microprocessor. In plain language, the 4004 chips were used to make portable digital calculators. Mechanical calculators became obsolete. Other companies then made their chips to power watches, typewriters, phones, and other devices. Computer chips evolved to power massive computers linked together into networks as well as control satellites and space ships. b

https://www.intel.com/content/www/us/en/newsroom/news/intel-marks-50th-anniversary-4004.html

...and Space Trash

A company that tracks satellites noticed very strange behavior from a Chinese satellite, something out of Star Wars. The *Chinese SJ-21* satellite changed its orbit to approach then connect to a long dead *Compass Go* satellite, changing its orbit. The Compass satellite is a Chinese satellite in a fixed orbit around the Earth. The SJ-21 then tossed the Compass into a graveyard orbit like throwing out trash before the SJ-21 returned to his fixed orbit above the Congo in Africa. b

https://www.dw.com/en/chinese-space-cleaner-spotted-grabbing-and-throwing-away-old-satellite/a-60658574

https://youtu.be/vrLfHv6sze0

You Resemble a Double Helix

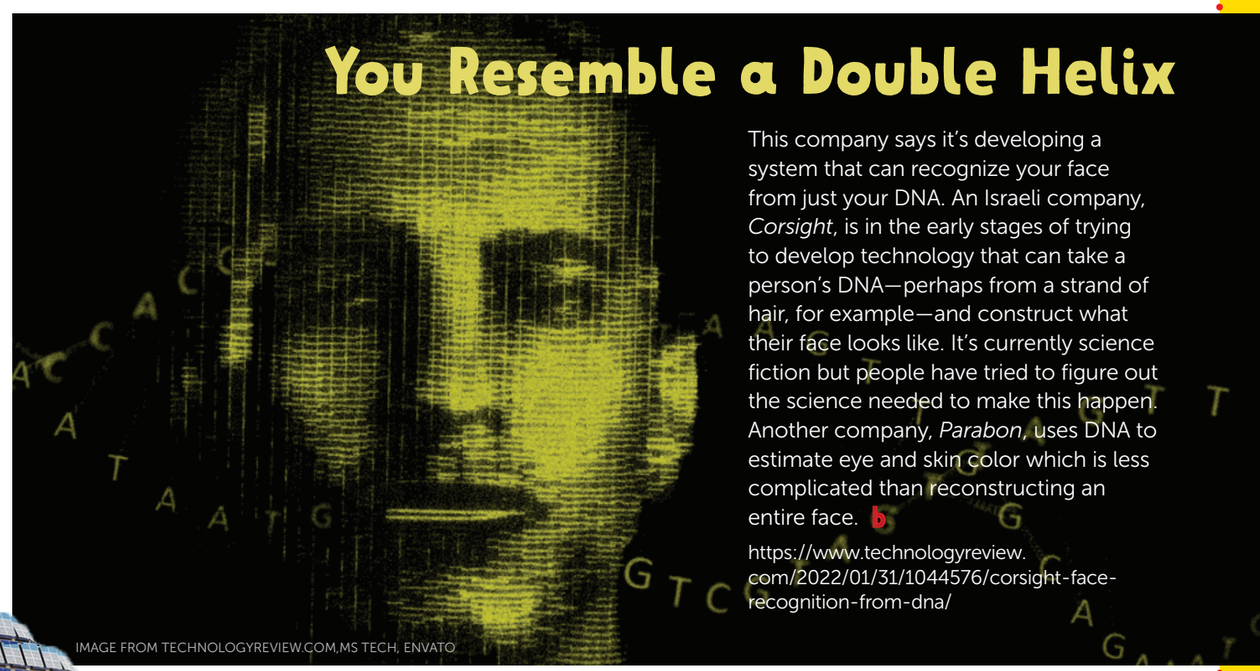


IMAGE FROM TECHNOLOGYREVIEW.COM,MS TECH, ENVATO

This company says it's developing a system that can recognize your face from just your DNA. An Israeli company, *Corsight*, is in the early stages of trying to develop technology that can take a person's DNA—perhaps from a strand of hair, for example—and construct what their face looks like. It's currently science fiction but people have tried to figure out the science needed to make this happen. Another company, *Parabon*, uses DNA to estimate eye and skin color which is less complicated than reconstructing an entire face. b

https://www.technologyreview.com/2022/01/31/1044576/corsight-face-recognition-from-dna/

BY TIM MCGUIGAN

Let Them Play! Or, The 80/20 Rule for Young Learners

During a recent admissions event for my school, I was speaking to a group of parents of younger children. When speaking to prospective parents, I usually try to give a thorough rundown of my class and my methods. After explaining my approach to teaching cognitive skills through technology, a parent of a seven-year-old asked me how his child should be learning to code to set her up with the best skills for the future.

He wondered if she should be learning languages like C++ through tutorials on *YouTube* or even working with a tutor. A few other parents replied that they were wondering similar things. I understand these instincts as a parent. Your child has an interest in technology, and you want to encourage them in their passions. However, in my mind, learning to "code" is approaching the problem from an adult's perspective. One of the best age-appropriate technology skills elementary school students should focus on is imaginative design through play.

Certainly, every child is a different case, but I like to apply a rule that I call the 80/20 rule. To be balanced, 80% of a child's time should be focused on generative, creative design. This tinkering or making can take many forms, but anything that springs from their imagination and is driven by their interests works towards this goal. Designing mini-games in *Scratch*, building marble mazes, assembling *Arduino* robots, or just crafting with *Lego* are all great examples. In short, this is what we call play, and it is the best technology tutor in my opinion. The remaining 20% is for what I call "end user" time: playing video games, following tutorials on YouTube, or practicing a more advanced language.

To give an example from my own home, my son absolutely adores *Lego* and *Star Wars*, and he frequently gets sets for birthdays and holidays. Putting together these sets teaches really great skills like following linear directions and spatial reasoning. However, if he were only putting together pre-packaged sets, he would be missing out on the problem-solving and design benefits of *Lego*. So, I maintain a large collection of loose pieces for him to create with. After a time, the majority of his beloved sets get relegated to the sundries bin.

It should be said that when he builds from pure imagination, his designs are mismatched and clunky. A far cry from the sleek art that *Lego* engineers design on a regular basis. However, his designs contain an added layer of complexity that I can't see. When he looks at a *Lego* plane he built from scratch, his imagination fills in the gaps in his design and adds a creative sheen that only he can see. Moreover, he has figured out thousands of small problems with his design along the way—problems he never would have encountered without the process of play.

I see this same dynamic in my middle school students. Their first platformer or open-world game designs are slow, pixelated, and buggy. However, when the design of these games flows entirely from their own minds, you can see the effect of the design process on their faces. They simply beam with pride when I play their games and find some strange mini-boss or a hidden level. If we had instead spent this time learning a language through step-by-step tutorials we would have missed many opportunities for authentic problem solving.

Well, this does beg the question: what if a student is truly adept with programming and shows a proclivity towards learning new technologies? This is an interesting question, but I believe the 80/20 rule is even more crucial for bright, motivated students. I have taught quite a few students like this throughout my career, and it all comes down to opportunity cost. There will always be time for them to learn languages and concepts like version control, but access to childhood imagination is time-sensitive.

An interesting case that illustrates this concept is the childhood of computer scientist, philosopher, and virtual reality pioneer, Jaron Lanier. Jaron's father famously took him out of school at the age of nine to help build a geodesic dome of his own design in the desert. This would seem counterintuitive to many parents, although this effort did little to slow his education in technology. He enrolled in New Mexico State University by the age of 13, and helped to commercialize virtual reality hardware like VR goggles by the age of 25.

While not every child needs or would benefit from this level of creative engagement, the power of childhood imagination still remains ever-constant. When the time comes, coding will still be there to be explored, but the time spent with creativity and play is still some of the most meaningful work we do in childhood.

NATHAN DUMLAO, UNSPLASH

Give a gift of beanz magazine at 30% OFF!

What better way to learn than with a childhood friend that generates lasting memories?

Use code GIVEBEANZ when subscribing
https://beanzmag.com/give-beanz

"If people never did silly things, nothing intelligent would ever get done."
—Ludwig Wittgenstein

Thank you for reading this issue of beanz! Check out the links below to read stories from this issue online with links to learn more.

Color My World
https://beanzmag.com/what-is-color-explained

Practice Makes Perfect
https://beanzmag.com/james-webb-telescope

A Family Heirloom... Computer?
https://beanzmag.com/permacomputing-explained

May I Borrow a Cup of Bandwidth?
https://beanzmag.com/amazon-sidewalk-bandwidth

Back to the Stone Age
https://beanzmag.com/minecraft-sevtech-ages-mod

Where in the World Am I?
https://beanzmag.com/geoguesser-game

Responsive Web Design Using Grid View
https://beanzmag.com/grid-view-responsive-design

An Apple a Day...
https://beanzmag.com/mini-micro-space-game-project

It Takes All Kinds!
https://beanzmag.com/types-video-games

Dithering Down on Images
https://beanzmag.com/dithering-explained

How I Spent My Summer Vacation
https://beanzmag.com/summer-technology-projects

The Many Uses of Digital Multimeters
https://beanzmag.com/how-to-digital-multimeters

Print That!
https://beanzmag.com/print-to-screen-coding-history

Move Over, James Bond. OSINT Is Here.
https://beanzmag.com/open-source-intelligence-tool

Give It the Boot!
https://beanzmag.com/bootstrapper-explained

tidbitz
https://beanzmag.com/june-2022-news-wire

Let Them Play!
https://beanzmag.com/80-20-rule-young-learners

Sssssssubscribe!
It'ssss Sssso Worth It.

https://beanzmag.com/subscribe