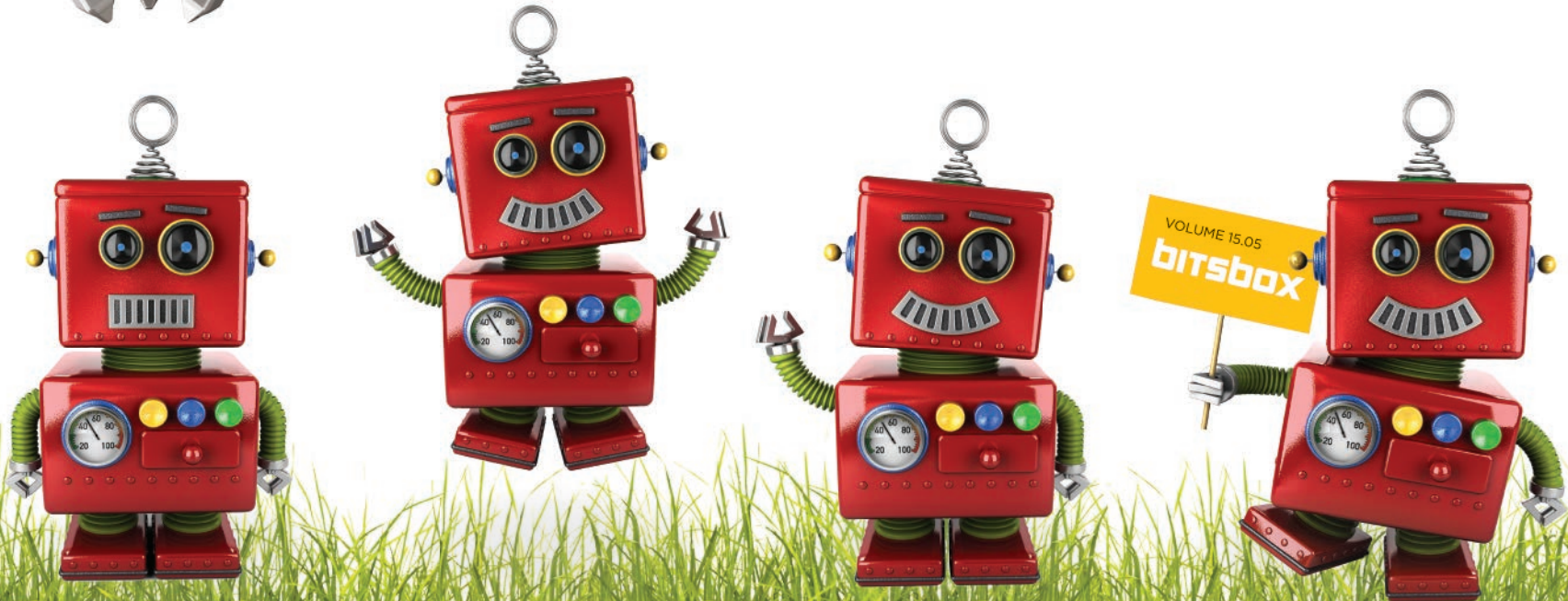


22 OUTLANDISH APPS TO CODE

WELCOME TO THE
FUTURE



Want to make your own apps? You've come to the right place!

With Bitsbox, you can code real apps that work on gadgets like phones and tablets. All you need is a computer with internet and a working brain.


HOW TO BITSBOX

- 1 Find a computer with a physical keyboard.**
The coding part of Bitsbox isn't meant to be done on tablets just yet.
- 2 Open a web browser and go to bitsbox.com**
We recommend Chrome, Firefox, Safari, or Internet Explorer 11.
- 3 Click Get Started.**
Have fun!



How can I run my apps on a phone or a tablet?

Before you do these steps, install a QR reader app on your gadget. Go here for a couple of suggestions: bitsbox.com/QRapps

- 1** On a computer, open the Bitsbox app you built.
- 2** Click the sharing icon  in the corner of the screen.
- 3** Scan the QR code with your gadget.

It looks like this!



When you change the code on your computer,
the app on your gadget changes, too!

Welcome to the Future

We hope the future is filled with friendly robots, amiable aliens, and space travel that's as easy as catching the bus to school. Our cars may not fly, but they'll drive themselves. We can't wait.

As you code your way through this issue's apps, pay special attention to *ZapChat* on page 12. Our BNF (Big New Feature) is code that lets more than one person do stuff in the same app, at the same time. It's super powerful—what will you make with it?

To the future!

Scott, Aidan, Anastasia & Jeff
(the whole Bitsbox team)

GROWNUPS READ THIS!

You'll find the *Grownup Guide* for this issue at bitsbox.com/grownups

In the Grownup Guide, you'll find:

- An FAQ section about Bitsbox in general, (and this issue in particular)
- Descriptions of the coding concepts we're using
- Line-by-line explanations of each app's code
- Suggestions for extending the apps to make them even more fun

MINI APPS TO GET YOU STARTED

8 2 4 3 What will you see?



```
1 fill('black')
2 stamp('earth')
```

Maybe you'll see Earth from outer space!
Can you change it to a different planet you'd like to visit?

3 7 8 3 Who will you be?

```
1 stamp('crystal ball')
2 text('You will be a ninja',200,500)
```

The text command writes whatever you want it to.

2 3 4 6 What will you create?

```
1 function drag() {
2   stamp('airbrush',x,y,100)
3 }
```

Draw with a touch! Remember—anything is possible.

7 3 9 5 Which pet will you adopt?

```
1 fill('pets')
2 line(530,900,10,'black')
```



Change line 2 to point at your favorite critter.

6 9 7 8

Where will you park your Hoverhouse?

Flying cars are so 2075.
How about a flying *house* instead?



Start by typing in this code:

```
1 fill('hoverland')
2 stamp('hoverhouse',200,300)
```

Now move your cursor across the tablet on your screen.

Do you see the little orange numbers? They help you figure out where to place stamps.

Move your Hoverhouse.

Change the numbers on line 2 like this:

```
1 fill('hoverland')
2 stamp('hoverhouse',600,200)
```



Where else can you park your Hoverhouse?

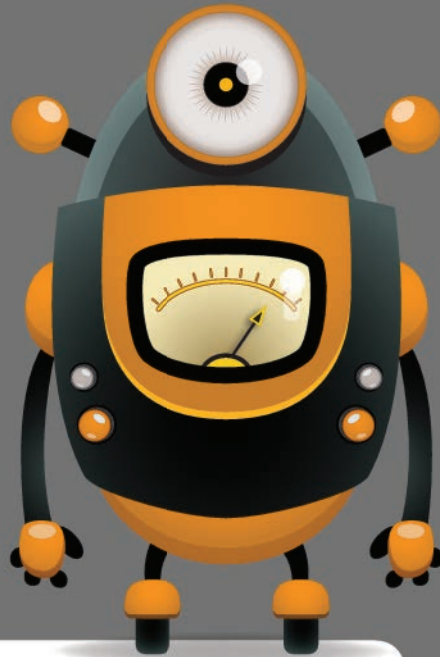
How would you add another Hoverhouse to the app?



7 6 8 8

Bitsbox Boxbot

Draw a robot using basic shapes. World domination happens one box at a time.



```

1 color1 = 'bitsbox yellow'
2 color2 = 'outrageous orange'
3 color3 = 'cornflower blue'
4
5 fill('robots')
6 box(300,350,200,300,color1)
7 circle(370,420,10,color2)
8 circle(430,420,20,color2)
9 line(350,500,450,480,color3,20)

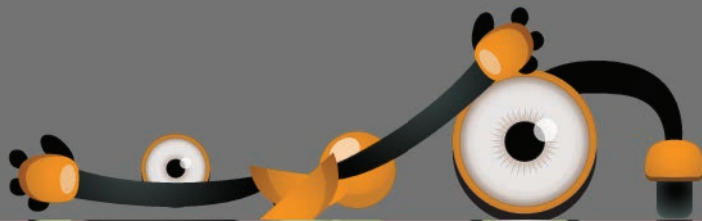
```



Make changes super quick!

See the code on line 1? It means that typing `color1` is the same thing as typing `'bitsbox yellow'`. Whenever you want to use the color `'bitsbox yellow'` in your app, you can just type `color1`.

In this app, `color1` is a *variable*. Variables are super common in coding because they make it easy to make big changes very quickly.



8 7 7 9

MAZE RUNNER

Wherever you drag your finger, Mazebot follows.
Can you find the way out?

```

1 fill('maze')
2 bot = stamp('mazebot',100)
3
4 function drag() {
5   bot.move(x,y,300)
6 }

```



Lines 4, 5 and 6 tell
the robot to move
when you drag.



This app is a total drag.

`drag()` is a basic Bitsbox command. It tells your app what to do when someone drags their cursor (or their finger) on the screen.




Pssst! Try substituting `'maze2'` on line 1.

6 5 5 3

Tap Addict

Tap tap tap to make the score go up.
How high can you go?*

*Please stop before your finger falls off.

 This is how you can cheat.


Line 7 increases the score by 1 every time you tap.
How would you change the code to increase the score by 10 instead?

```
1 fill('steel hexes')
2 stamp('scoreboard')
3 score = 0
4 display = text('TAP!',200,570,200,'yellow')
5
6 function tap() {
7   score = score + 1
8   display.change(score)
9 }
```

4 6 8 1

Replicator Roulette

Can't decide what's for dinner?
Let the replicator randomly
select a delicious meal.

 Gag is disgusting (unless you're a Klingon).

Every time you run this app, line 2 chooses what's for dinner from among the choices in the `random()` command.

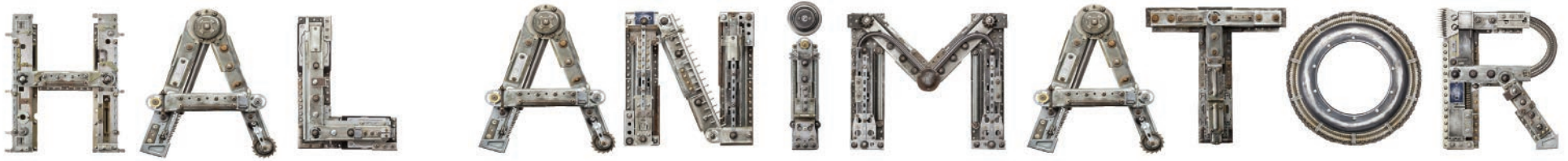
Using `random()` is like telling your app to roll imaginary dice. You won't know what's coming until it appears!

What e/se can you tell the replicator to serve?

```
1 fill('replicator')
2 dinner = random('tribble','sushi','gagh')
3 speed = 1000
4 food = stamp(dinner,10)
5 food.size(350,speed)
```



8 8 9 4

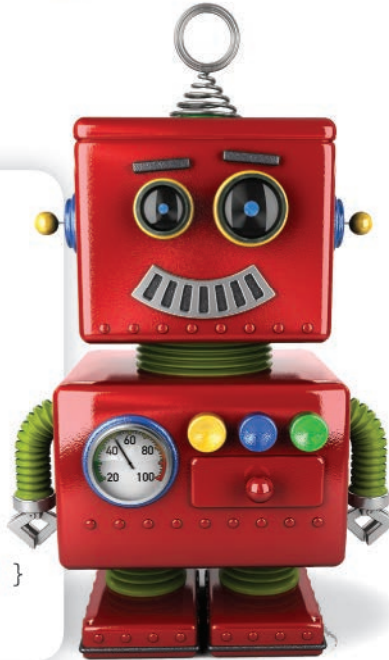


Tap the screen to make Hal dance.
Bust a move until he blows a circuit.

```

1 hal = stamp('hal')
2 poses = []
3 poses[1] = 'hal surprised'
4 poses[2] = 'hal wave'
5 poses[3] = 'hal party'
6
7 i = 1
8 function tap() {
9   hal.change(poses[i])
10  i = i + 1
11  if (i == poses.length) { i = 1 }
12 }

```



🧠 Teach Hal some new moves.

When you're coding, sometimes it's handy to make lists. These lists are called *arrays*. Line 2 creates a new array called **poses**. Lines 3, 4 and 5 each put a different stamp into the array. Each of these stamps is Hal in a different pose.

Can you make your own dance? Hal has even more poses. Try adding these lines of code after line 5:

```

poses[4] = 'hal hello'
poses[5] = 'hal jump'
poses[6] = 'hal scared'

```

6 4 7 2

Trouble with Tribbles

Those pesky tribbles are filling up the cargo hold.
Tap them fast to make them pop!

```

1 fill('pink')
2
3 function loop() {
4   x = random(800)
5   y = random(1000)
6   size = random(50,300)
7   item = stamp('tribble',x,y,size)
8   item.tap = pop
9 }

```

🧠 Loops loop forever and ever.

The **loop()** function repeats over and over again. Every time it does, it adds a random-sized, randomly positioned tribble to the screen.



8 4 5 5

Countdown Timer

Create your own egg timer.
Or doomsday clock.
It depends on your outlook.



```

1 seconds = prompt('How many seconds?',10)
2 words = text('GO',100,370,270,'red')
3
4 function loop() {
5   if (seconds > 0) {
6     seconds = seconds - 0.05
7     words.change(seconds)
8   } else {
9     loop = null
10    words.change('END')
11    sound('alert',100,3)
12  }
13 }

```

 It all depends... on line 5.

Line 5 looks to see if the timer has reached 0 yet:

If it hasn't, it reduces the time left by 1/20th of a second.
If it has, it stops the loop and writes the word "END".

5 9 9 5



Send a homemade digital card
to someone special.

Father's Day is in June!

```

1 fill('mottled sky')
2 text('- Love from me',250,800)
3 card = stamp('envelope2',1070).rotate(90)
4 thing = stamp('waxseal',400)
5 message = text('Tap to open',150,150,55)
6
7 function tap() {
8   thing.change('hal presents')
9   thing.dance()
10  message.change('You are a great dad!')
11  card.size(1,500)
12  card.rotate(720,500)
13 }

```

 Change is in the air.

Line 8 uses the `.change` command to
make the red wax seal turn into the robot
when someone taps the screen.



Visit bitsbox.com/howtoshare to learn how to email this app!

STAR PILOT

Fly through space and watch the stars whiz by!

Stock up on dilithium crystals.

The loop on line 19 only does two things, but each of those things does a lot:

`create()` is a function that makes new stars, one at a time.

`warp()` is a function that sends each star flying off the screen.

Want to see something cool? Change the number on line 20 to generate more stars.

```

1 fill('nebula')
2 hud = stamp('hud')
3
4 function warp(star) {
5   star.size(star.width * 1.2)
6   star.speed = star.speed * 1.2
7   star.move(UP, star.speed)
8   if (offscreen(star)) {
9     star.hide()
10  }
11 }
12
13 function create() {
14   star = stamp('flare', hud.x, hud.y, 10)
15   star.speed = random(2, 10)
16   star.rotate(random(360))
17 }
18
19 function loop() {
20   repeat(create, 2)
21   find('flare').forEach(warp)
22 }
23
24 function tap() {
25   hud.move(x, y, 1500)
26 }

```


1884

Gravity Dance

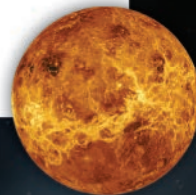
Tap and hold to grow your very own planets.
Tap and drag to send them careening
through space.

Can you make a solar system that
doesn't fly off the screen?

Hint: start with a big planet in the middle.

```
1 fill('nebula')
2
3 function touch() {
4   moon = stamp('jupiter',x,y,5)
5   drag()
6 }
7
8 function drag() {
9   moon.vx = x - moon.x
10  moon.vy = y - moon.y
11 }
12
13 function touching() {
14   moon.size(moon.width + 5)
15 }
```

You can click the green run button after
typing in this section of code, if you like.



```
16
17 function gravity(puller) {
18   if (this == puller) { return }
19   dist = distance(this,puller)
20   mass = puller.width * puller.width * 0.005
21   force = mass / (dist * dist)
22   this.vx = this.vx + (puller.x - this.x) * force
23   this.vy = this.vy + (puller.y - this.y) * force
24 }
25
26 function physics(body) {
27   body.move(body.x+body.vx, body.y+body.vy)
28   find('jupiter').forEach(gravity,body)
29 }
30
31 function loop() {
32   find('jupiter').forEach(physics)
33 }
```

Math is all around us.

This app is called a *simulation*. A simulation uses
code (and a lot of tricky math) to model a version of
something that exists in real life. Make small changes
to the code and see what happens!

The Bitsbox Team's Favorite Robots



JEFF: THE MECHANICAL TURK

I love the Mechanical Turk even though it was a hoax. Created in 1770, it seemed like one of the world's first functioning robots. For over eighty years, it played chess against politicians and kings. It even played Napoleon! Sadly, it was all a trick. A human chess master hid inside the cabinet and controlled the Turk's arm. There's an important lesson in here somewhere....



ANASTASIA: FALGOR

What's 3 feet tall, hot pink, and able to shoot Nerf balls across a room? The robot that got me interested in engineering! In high school, I joined the FIRST robotics team, working with my teammates to design and build Falgor. I'm a little disappointed that I didn't work on the programming team, but I did learn to run a milling machine. And I ended up as a mechanical engineer.



SCOTT: SOUNDWAVE THE TRANSFORMER

Okay, yes, he's a bad guy. But he's a sensitive bad guy. He transforms into a cassette player*, see? And his cassettes transform into pterodactyls and cheetahs and stuff! I always imagined that Soundwave cares for his little bots, nestled just so in his chest-hatch. It's downright heartwarming.

*Ask your parents what a cassette player is.



AIDAN: SELF-DRIVING CARS

Okay, okay—so these don't *look* like robots. But I think they're going to completely change the way we live. In a few years, you'll be able to get into a car, tell it where to go, and watch a movie. Or do your homework. Or take a nap. Sound crazy? Think again. Cars that can drive themselves already exist, and they've been tested over thousands of miles of roadway. It's only a matter of time before kids can travel wherever they need to go without their parents. How amazing would *that* be?





JUMBLED JARGON

Unscramble these coding words! When you're done, copy the highlighted letters to the white boxes below. Unscramble *those* to discover a secret stamp you can use in your apps!

EZOR

RAYAR

PMAST

ATTORE

NCOUNTIF

Unscramble the letters to reveal a secret stamp!

stamp(' _____ ')

SPHERO DRIVER

Drive your own Sphero™. Hit the flags in order to win!

```

1 fill('grass')
2 base = stamp('joystick base',120,900)
3 stick = stamp('joystick',120,900)
4 flags = ['portal1', 'portal2', 'portal3', 'portal4']
5 sphero = stamp('sphero3',100)
6 start = new Date()
7
8 function place(name) {
9   stamp(name,random(700),random(700))
10 }
11 flags.forEach(place)
12
13 function drag() {
14   stick.move(x,y)
15 }
16
17 function untouch() {
18   stick.move(base.x,base.y,500)
19 }

```

```

20
21 function collide() {
22   collision = sphero.hits(flags[0])
23   if (collision) {
24     collision[0].pop()
25     flags.shift()
26   }
27   if (flags.length == 0) {
28     seconds = (new Date() - start) / 1000
29     msg = 'You did it in ' + seconds + ' seconds!'
30     text(msg,50,90,'white')
31     loop = null
32   }
33 }
34
35 function loop() {
36   angle = (stick.x - base.x)/3
37   speed = (base.y - stick.y)/3
38   sphero.rotate(RIGHT,angle)
39   sphero.move(UP,speed)
40   sphero.wrap()
41   collide()
42 }

```



Add more targets!

Add more elements to the array on line 4:

```
'portal5', 'portal6', 'portal7'...
```

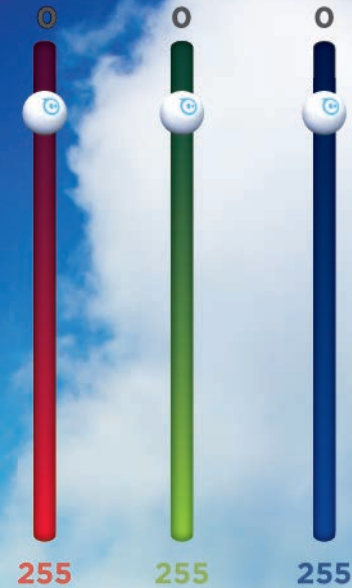
You get the picture.

Move the sliders to change the color of the big Sphero™.

7 8 5 6

Mood Ball

```
1 fill('black')
2 stamp('big sphero')
3
4 handle = 'tiny sphero'
5 red   = stamp(handle,200,100)
6 green = stamp(handle,400,100)
7 blue  = stamp(handle,600,100)
8
9 function drag() {
10   slider = red
11   if (x > 300) { slider = green }
12   if (x > 500) { slider = blue }
13   slider.move(slider.x,y)
14   fill(red.y, green.y, blue.y)
15 }
```



Feeling blue? How about red?

All the colors on your screen are made from a combination of red, green and blue. This app lets you mix your own colors to change Sphero's mood.



ZapChat

People running this app at the same time on different gadgets can send messages in real time!

More people = more fun!

```
1 fill('zap messenger')
2
3 function tap() {
4   message = prompt('Type your text:')
5   text(message,x,y,'white')
6   send(message,x,y)
7 }
8
9 function get(message,x,y) {
10  text(message,x,y,'yellow')
11 }
```



Send and Get

This simple little app uses a pair of powerful commands to make it possible for lots of people to use it at the same time.

The `send()` command takes a piece of data and stores it on the Bitsbox server.

The `get()` function watches the server for new data and downloads any that it sees.

Visit bitsbox.com/howtoshare to learn how to share this app with other people!

hi there!

what are you eating?

mouseburgers

hello!

who is this?

I'm here, too!

This is amazing!

What are mouseburgers?

Yuck.

I'm out of here.

Escape Pod

Help these stranded astronauts get back to their flying saucer. Watch out, though—they don't fly straight!

```

1 fill('stars2')
2 pod = stamp('saucer',350,900)
3
4 function thrust() {
5   this.change('flyer')
6 }
7
8 function create() {
9   astro = stamp('drifter',100)
10  astro.move()
11  astro.rotate()
12  astro.rotate(5000,100000)
13  astro.tap = thrust
14 }
15 repeat(create,5)

```

```

16
17 function launch(target) {
18   target.move(375,-600,2000)
19 }
20
21 function fly(astro) {
22   astro.move(UP,30)
23   if (astro.hits(pod)) {
24     astro.rotate(0).change('astro')
25   }
26 }
27
28 function loop() {
29   find('flyer').forEach(fly)
30   saved = find('astro')
31   if (saved.length >= 4) {
32     pod.change('flying saucer')
33     launch(pod)
34     saved.forEach(launch)
35   }
36 }

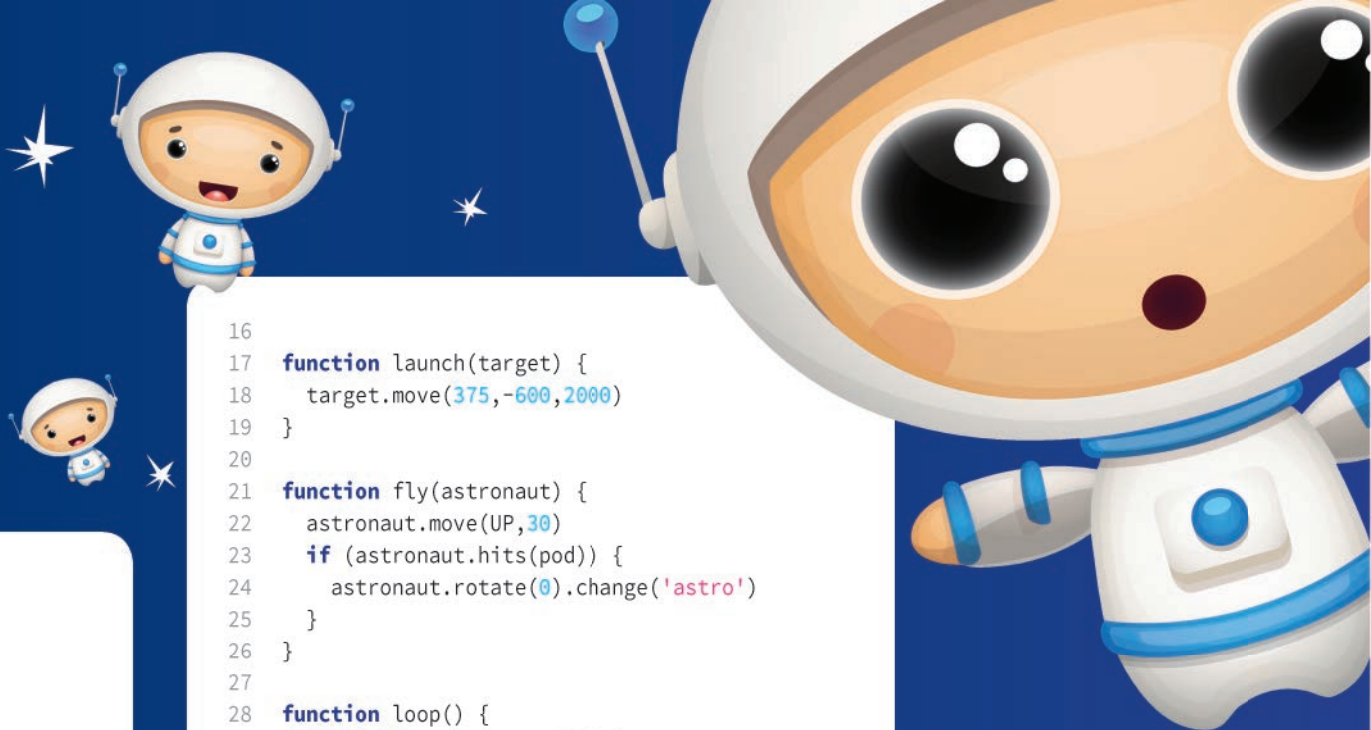
```

🧠 Dizzy much?

This app is a showcase for Bitsbox's `.rotate` command.

On line 11, there's nothing between `rotate`'s parentheses `()`. This tells the app to rotate each new astronaut randomly—so they're not all facing the same way when the app starts.

On line 12, `rotate` has two huge numbers in its parentheses. The first tells the astronaut to spin 5000 degrees, which is almost 14 times. The second tells the astronaut *how long* to spin: 100,000 *milliseconds*, which is 100 seconds, which is 1 minute and 40 seconds.



Nano Doctor

Oh no! Your patient has been infected by Dr. Dastardly's nano-germs. Use your nanobot to cure the infection by yo-yo zapping the germs into oblivion.

```

1 fill('blood vessel')
2 lure = stamp('flash',200,220,150)
3 stamp('germbot',80).move()
4 stamp('germbot',80).move()
5
6 function wander(germ) {
7   germ.move(EAST,5).rotate(LEFT,5)
8   germ.size(200,5000)
9   germ.wrap()
10 }
```

```

11
12 function loop() {
13   find('germbot').foreach(wander)
14   catches = lure.hits('germbot')
15   if (catches) {
16     catches[0].pop()
17     lure.move(200,220,1000)
18     stamp('germbot',80).move()
19   }
20 }
21
22 function tap() {
23   lure.move(x,y,500)
24 }
```



Move over, and while you're at it, rotate.

Line 7 is pretty cool. Two different Bitsbox “dot” commands are *chained* together on a single line: `.move` and `.rotate`. The germ moves 5 pixels in the **EAST** direction (to the right), and rotates **LEFT** (counter-clockwise) by 5 degrees.

In this case, both commands are included on the same line to save space—doing so makes the code one line shorter. It could also have been written on two lines, like this:

```
germ.move(EAST,5)
germ.rotate(LEFT,5)
```


CUBEBOT

YOGGA

Cubebot is remarkably flexible. Help him achieve nirvana by striking all the right poses.



```

1 fill('yoga studio')
2 arm1 = stamp('botarm',314,444)
3 arm2 = stamp('botarm2',456,444)
4 body = stamp('botbody')
5 leg1 = stamp('botleg',320,600).rotate(195)
6 leg2 = stamp('botleg2',430,605).rotate(165)
7
8 mover = arm1
9 function drag() {
10   mover.aim(x,y)
11 }
12
13 function touch() {
14   if (x < 370 && y < 600) { mover = arm1 }
15   if (x > 370 && y < 600) { mover = arm2 }
16   if (x < 370 && y > 600) { mover = leg1 }
17   if (x > 370 && y > 600) { mover = leg2 }
18 }

```



Ready. Aim. Downward Dog.

Check out line 10: `.aim` is one of Bitsbox's niftier "dot" commands. You use it to aim an object at any point on the screen. Here, the code tells the limb to aim at the exact spot (x,y) where someone is touching the screen.

1094

Steer your Claw Snake to grab the drooly alien without hitting anything!

THE CLAW

1st

```
1 fill('gears')
2 head = stamp('claw',60)
3 food = stamp('alien3',70)
4 speed = 30
5
6 function slither() {}
7 function collide() {}
8
9 i = 1
10 function loop() {
11   food.rotate(RIGHT,5)
12
13   i = i + 1
14   if (i < 5) {
15     return
16   } else {
17     i = 1
18   }
19
20   slither()
21   head.move(UP,speed)
22   collide()
23 }
```

Type in one section of code at a time, then click the green run button. The game gets better with every new section you add.

Draw the claw and the alien.

2nd

```
24
25 function tap() {
26   head.aim(x,y)
27   direction = Math.round(head.rotation/90)*90
28   head.rotate(direction)
29 }
```

Drive the claw around the screen!

3rd

```
30
31 body = []
32 end = head
33 function grow() {
34   segment = stamp('chain',end.x,end.y,30)
35   body.push(segment)
36   end = segment
37 }
38
39 grow()
40
41 function follow(link) {
42   link.aim(head)
43   link.move(head.x,head.y)
44 }
45
46 function slither() {
47   tail = body[body.length - 1]
48   body.length = body.length - 1
49   body.unshift(tail)
50   follow(tail)
51 }
```

Start turning the claw into...
the CLAW SNAKE!

4th

```
52
53 function collide() {
54   if (head.hits(food)) {
55     food.move()
56     grow()
57     speed = speed + 1
58   }
59   if (head.hits('chain') || offscreen(head)) {
60     loop = null
61     points = body.length
62     text('GAME OVER with score of ' + points,150,500)
63   }
64 }
```

Finish coding the game!

Coding onward...

See if you can add a sound effect that plays
when the claw eats the alien.

STAMPS

Use these stamps (& fills & songs & sounds) to make any app your own! ✨
Just don't forget to put single quotes around them in your code, like this:

```
stamp('foxboy') fill('iguana') song('storm') sound('duck')
```



SONGS

bigrock
darkhop
explosions
hunter
love
merrygo
parade
run
whistle

FILLS



SOUNDS

alligator
balloon
blip
caw
chicken
cloud
damage
drip
droid
evillaugh
explosion
fish
ghost
hedgehog
horn
junglebird
laser
lightning
lion
mermaid
playball
quack
rain
rewind
rocket2
sax
teleport
toad
ufo2
yowl
zwoop



More stuff
online!

Look for more when
you're coding at
bitsbox.com!