

Designing for Future Hardware

Session 801

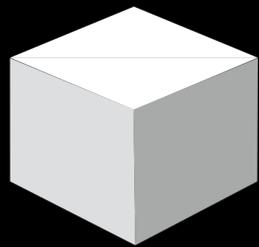
Jeffrey Traer Bernstein

Matthaeus Krenn

Bill Lindmeier

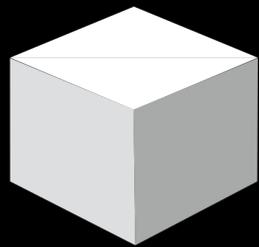
Who Is This Session For?

Who Is This Session For?



A thing

Who Is This Session For?

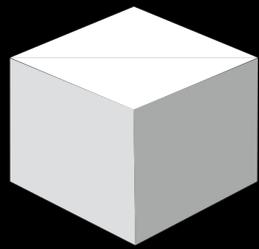


A thing



connected to

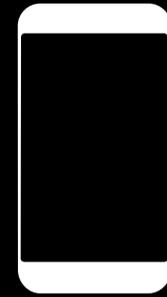
Who Is This Session For?



A thing

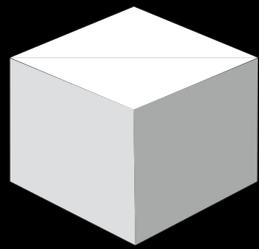


connected to



an app

Who Is This Session For?

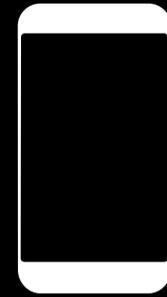


A thing

Dishwasher

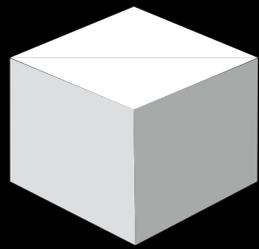


connected to



an app

Who Is This Session For?

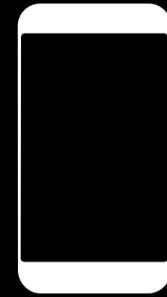


A thing

Dishwasher
Drone

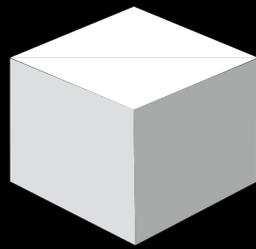


connected to



an app

Who Is This Session For?



A thing

Dishwasher

Drone

Golf Club

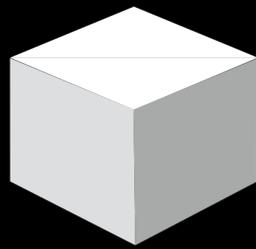


connected to



an app

Who Is This Session For?



A thing

Dishwasher

Drone

Golf Club



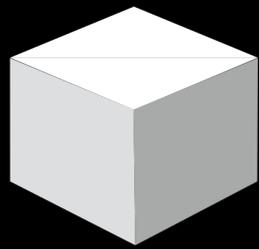
connected to



an app

Game

Who Is This Session For?



A thing

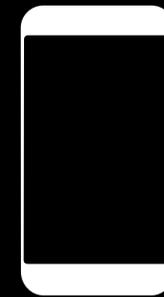
Dishwasher

Drone

Golf Club



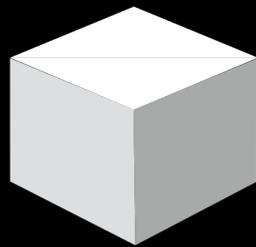
connected to



an app

Game
Messaging

Who Is This Session For?



A thing

Dishwasher

Drone

Golf Club



connected to

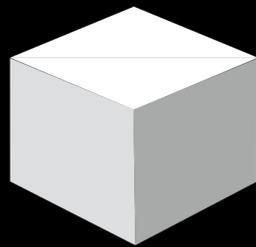


an app

Game
Messaging

Dating?

Who Is This Session For?



A thing

Dishwasher

Drone

Golf Club



connected to



an app

Game
Messaging

Dating?

A device you don't have yet...

Who Are We?

Who Are We?

Team of inventors, designers, prototypers

Who Are We?

Team of inventors, designers, prototypers

Work across all of Apple's products

Who Are We?

Team of inventors, designers, prototypers

Work across all of Apple's products

Explore what new interfaces mean to our devices, operating systems and applications

Who Are We?

Team of inventors, designers, prototypers

Work across all of Apple's products

Explore what new interfaces mean to our devices, operating systems and applications

Brought you Multitouch Gestures, Force Touch, Taptic Engine

Session 223

Prototyping

#WWDC14

Toast Modern



Why Prototype?

Why Prototype?

Test ideas

- Save time and money building the right things

Why Prototype?

Test ideas

- Save time and money building the right things

Get new ideas

- Make the experience of your product even better

How to Prototype

How to Prototype

Make fake apps

How to Prototype

Make fake apps

Show people

How to Prototype

Make fake apps

Show people

Learn from their feedback

How to Prototype

Make fake apps

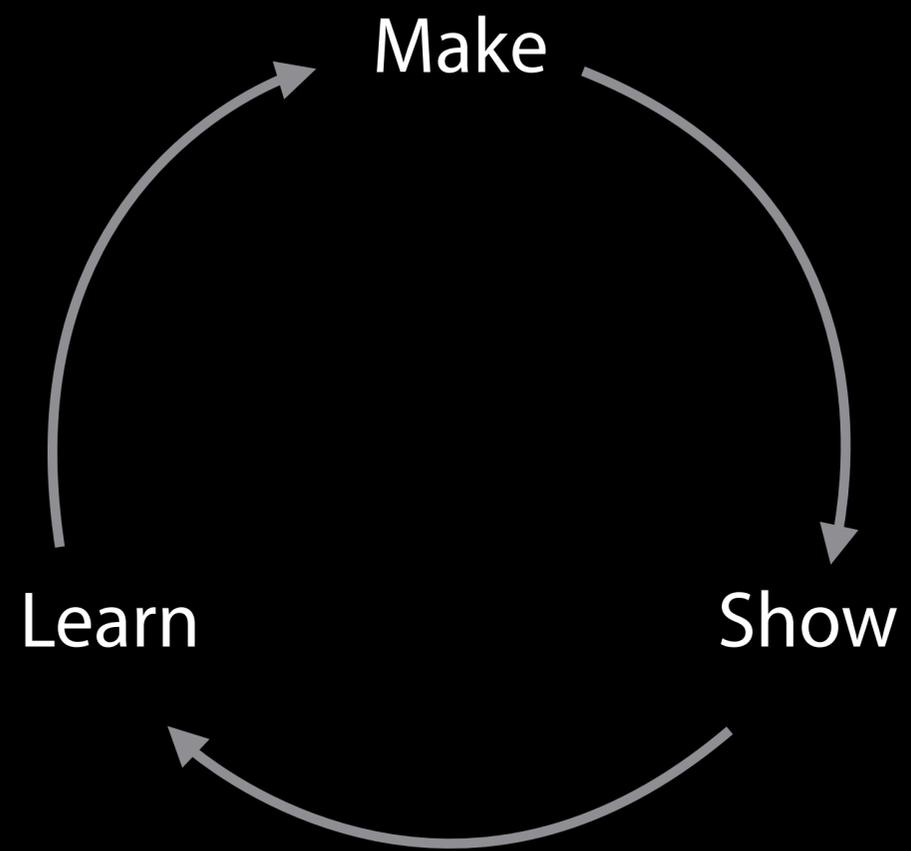
Show people

Learn from their feedback

Make

Learn

Show



“The workflow shown is something I may try.”

“The workflow shown is something I may try.”

“The prototyping video from wwdc is great. Grab a plate of toast.”

“The workflow shown is something I may try.”

“The prototyping video from wwdc is great. Grab a plate of toast.”

“Prototyping takeaways: 1. Make fake apps. 2. Uhm, missed that part.”

“The workflow shown is something I may try.”

“The prototyping video from wwdc is great. Grab a plate of toast.”

“Prototyping takeaways: 1. Make fake apps. 2. Uhm, missed that part.”

“I’m so proud of you.” –mom

Today

Today

Unveil a brand new, revolutionary, Toast Modern product

Today

Unveil a brand new, revolutionary, Toast Modern product

Peek behind the curtain at how Apple and partners prototyped the watch

Today

Unveil a brand new, revolutionary, Toast Modern product

Peek behind the curtain at how Apple and partners prototyped the watch

Create a device the connects to an app

Today

Unveil a brand new, revolutionary, Toast Modern product

Peek behind the curtain at how Apple and partners prototyped the watch

Create a device the connects to an app

Show you a few prototyping strategies for hardware and software

Revolutionary

Revolutionary

Disruptive

Revolutionary

Disruptive

Connected

Revolutionary

Disruptive

Connected

Social

Revolutionary

Disruptive

Connected

Social

Big Data-y

Revolutionary

Disruptive

Connected

Social

Big Data-y

Internet of Things Thing

Toastal Service

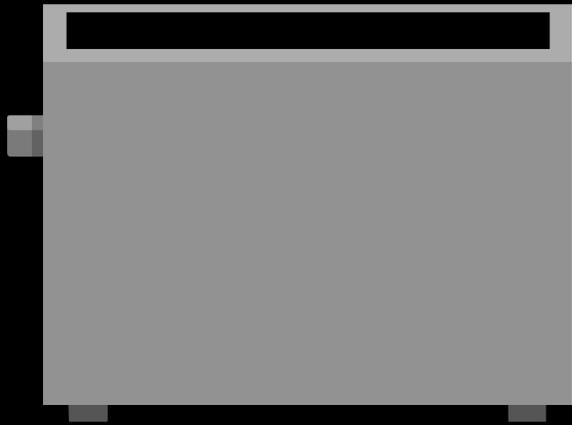
Toastal Service

The world's first social toaster

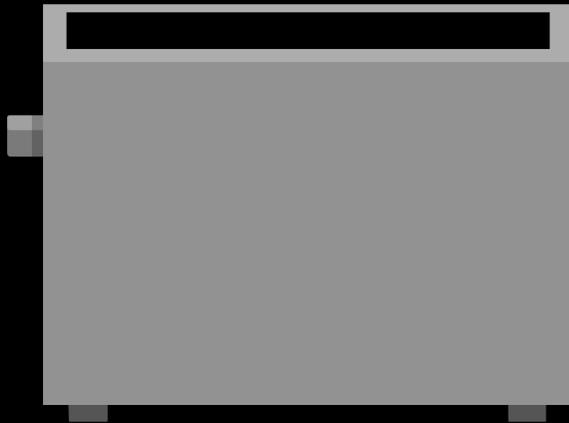
Toastal Service

The world's first social toaster

Send toast messages to your friends and loved ones



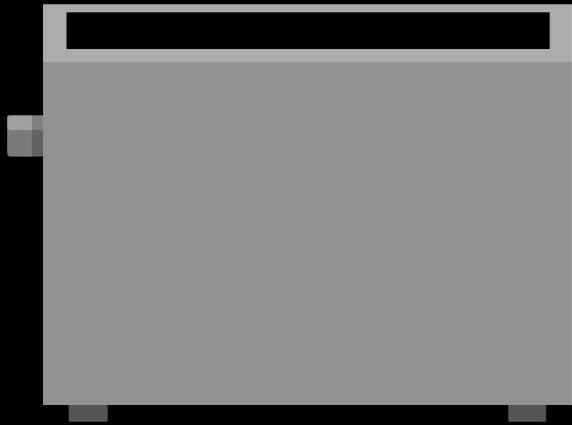
Silver



Silver



Space Grey



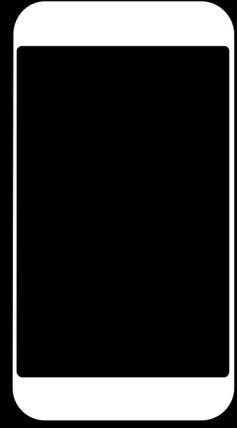
Silver



Space Grey



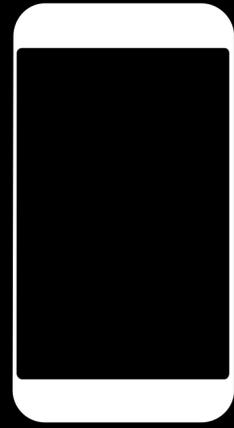
18-Karat Gold



iPhone



Toaster

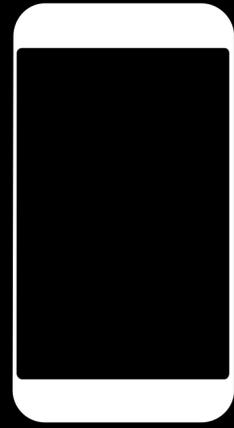


iPhone

Great interface



Toaster



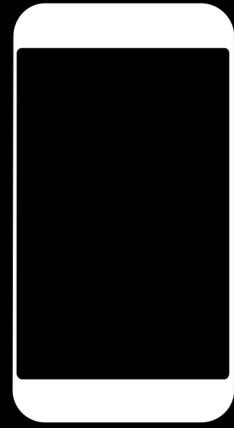
iPhone

Great interface

Internet



Toaster

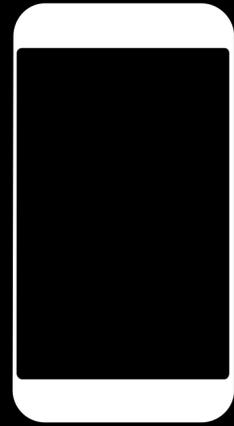


iPhone

Great interface
Internet
Always with you



Toaster



iPhone

Great interface
Internet
Always with you



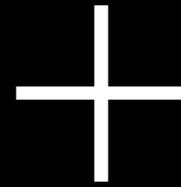
Toaster

Can turn bread into toast



iPhone

Great interface
Internet
Always with you



Toaster

Can turn bread into toast



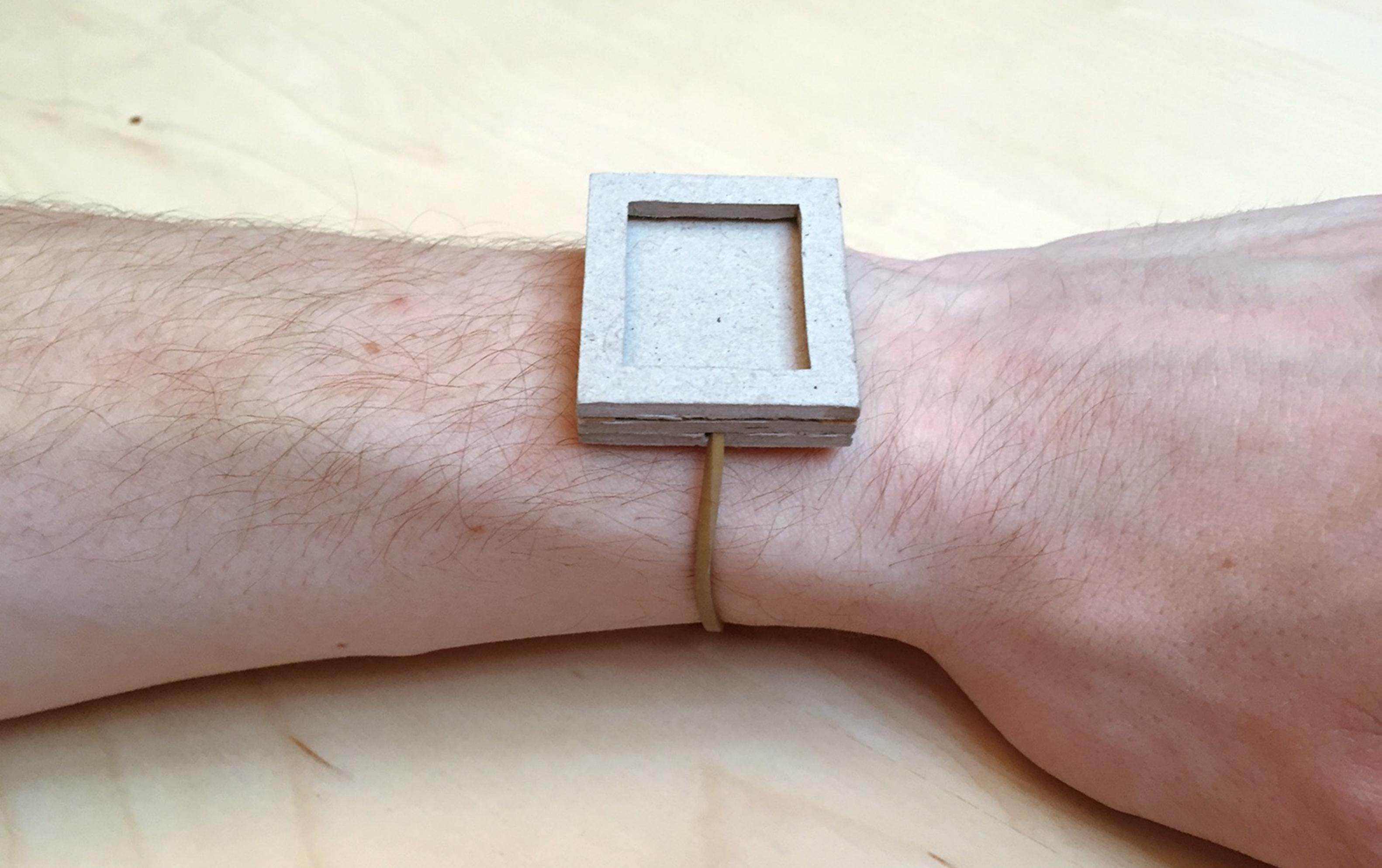














C.Train

09:28

2h08

Evian-les-Bains



C.Train

09:58

32min



TGV 6578



zurück

DACHZEILE STORY
Headline Story und
die mal 2 zeilig

zurück

DACHZEILE STORY
Headline Story und
die mal 2 zeilig

zurück

DACHZEILE STORY
Headline Story und
die mal 2 zeilig

WAS IST DAS?

Technische 1
Lorem ipsum wir
versuchen mal 1

Lorem ipsum dolor sit
amet, consetetur
s adipscing elitr, sed
diam nonumy eirmod
tempor invidunt ut
labore et dolore magna
aliquyam erat, sed diam
voluptua. At vero eos et

do

Prototyping Hardware and Software

Prototyping Hardware and Software

Fake hardware on screens

Prototyping Hardware and Software

Fake hardware on screens

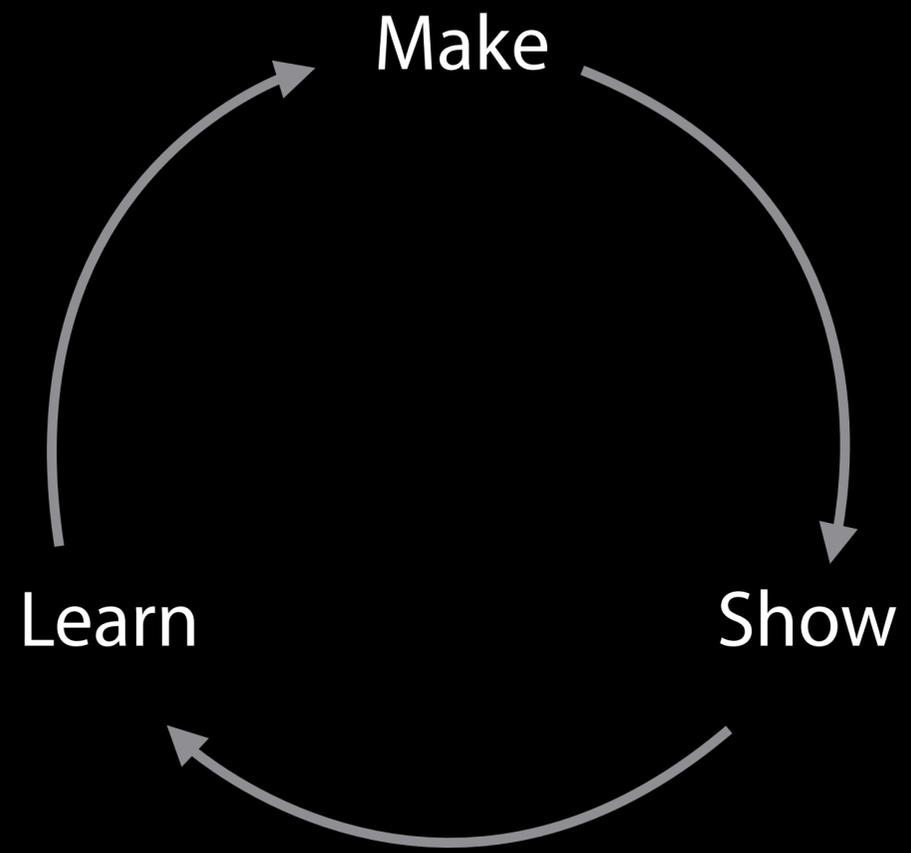
Fake software with pictures

Prototyping Hardware and Software

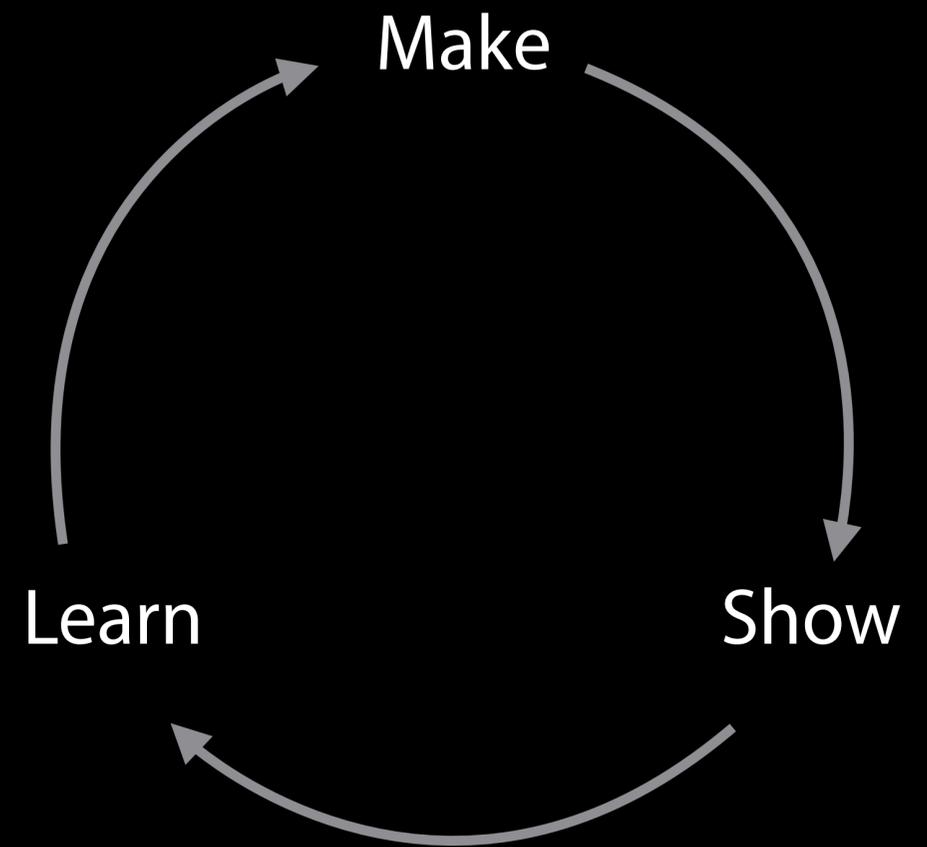
Fake hardware on screens

Fake software with pictures

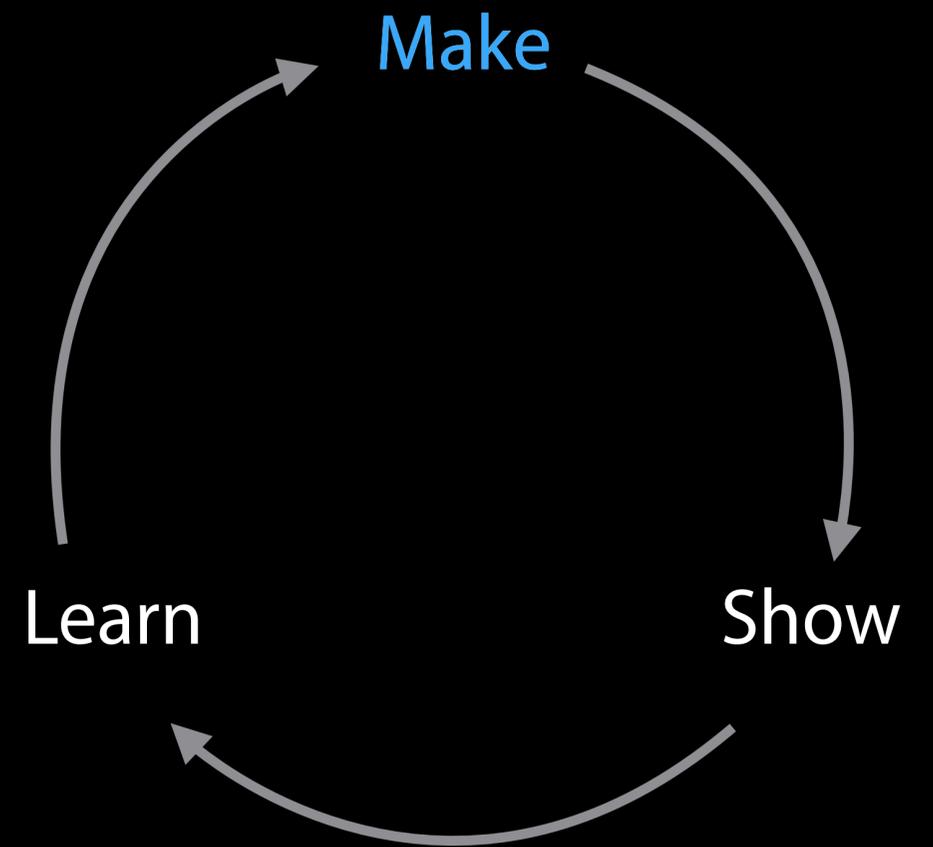
Try it in context, at the right size, in the right place



Make Fake Hardware and Software

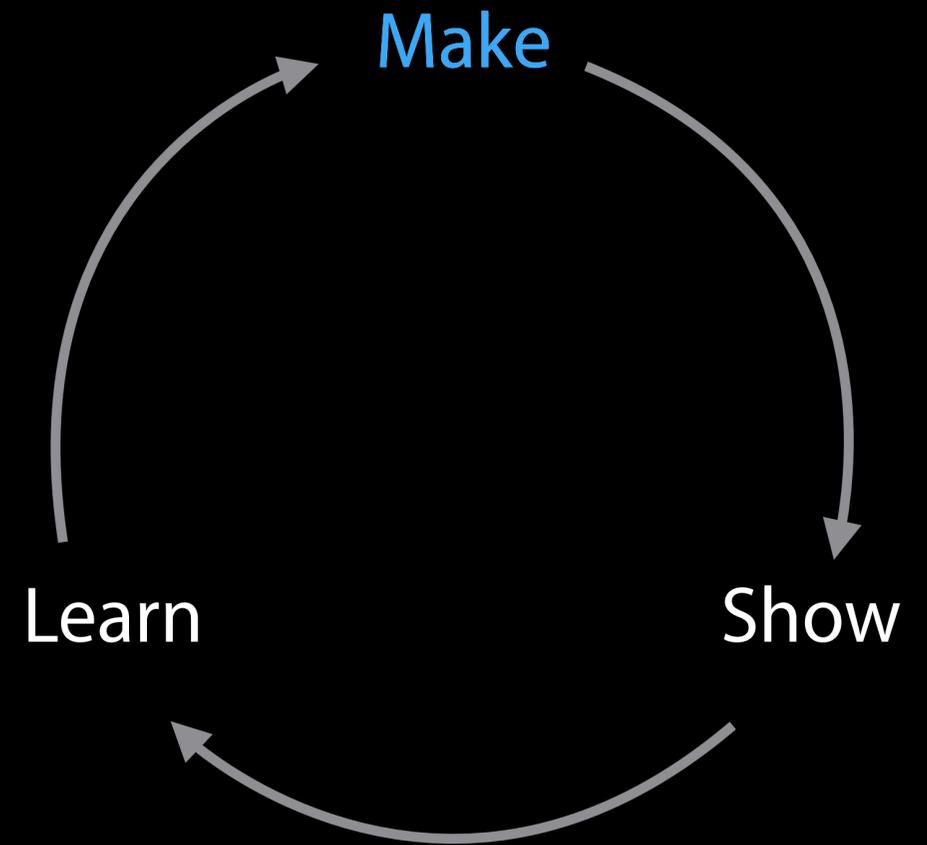


Make Fake Hardware and Software



Make Fake Hardware and Software

Fake hardware Fake app

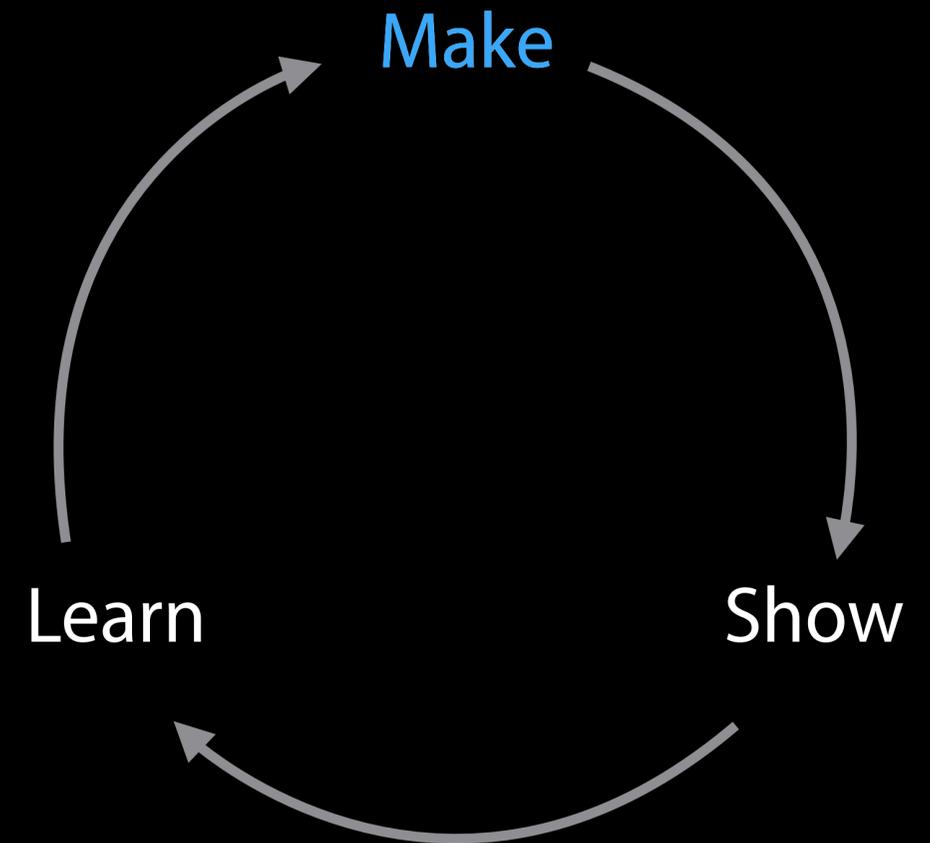


Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be
more real?



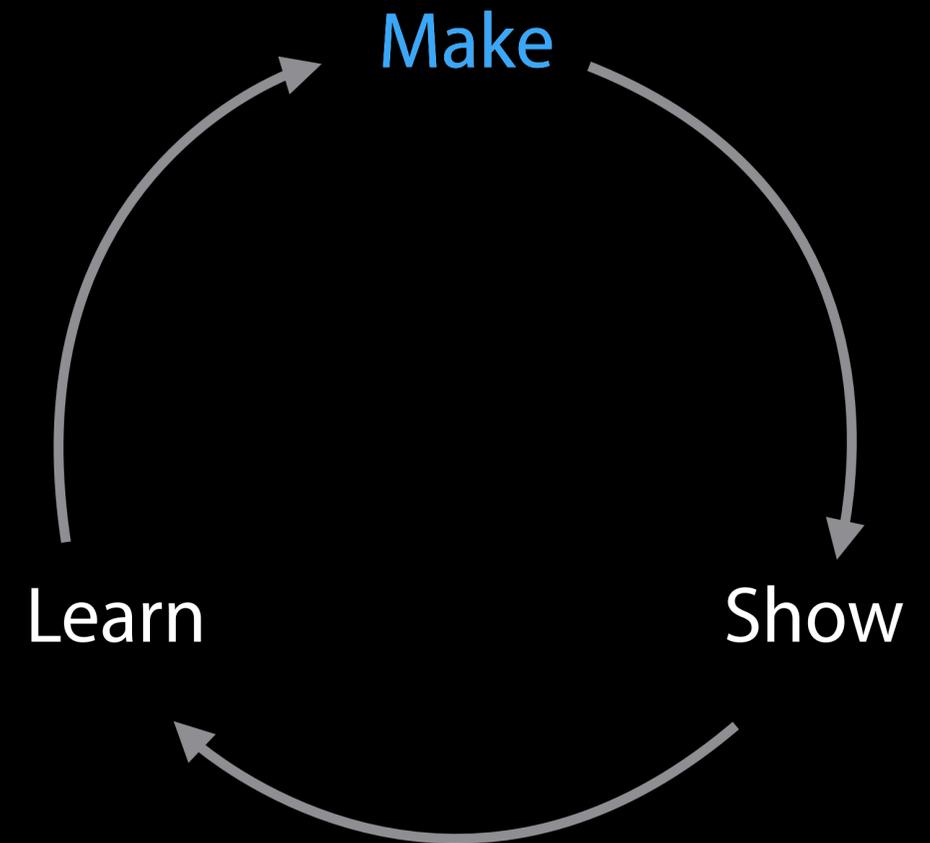
Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

What can be faked?



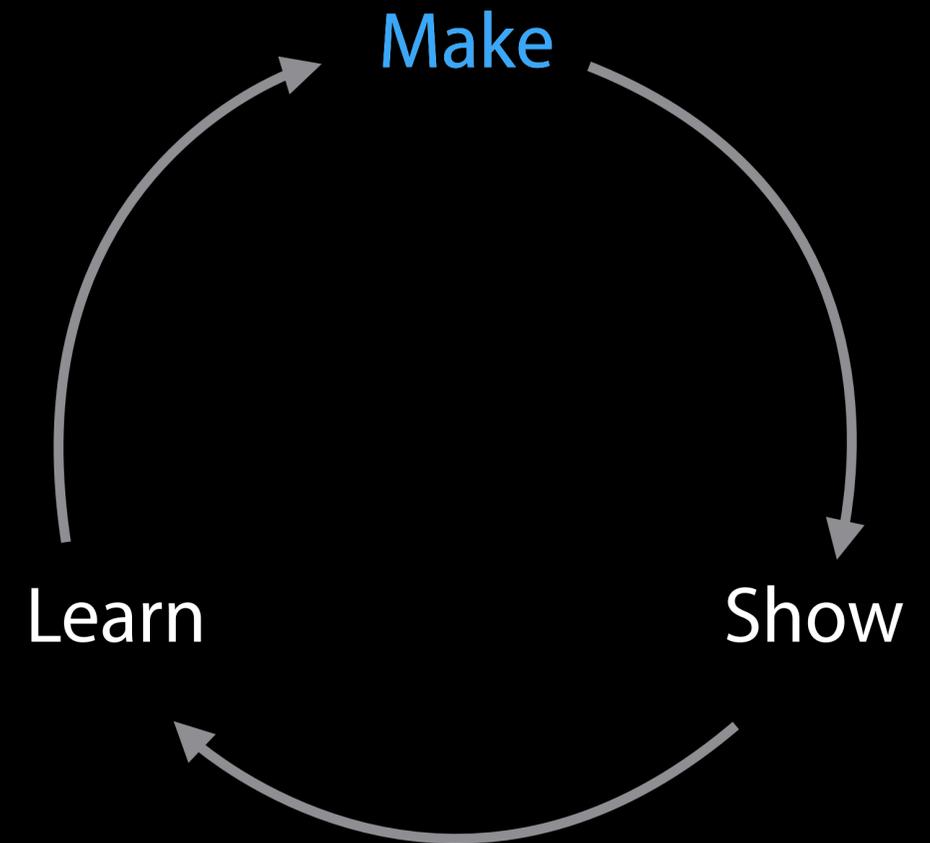
Make Fake Hardware and Software

Fake hardware Fake app

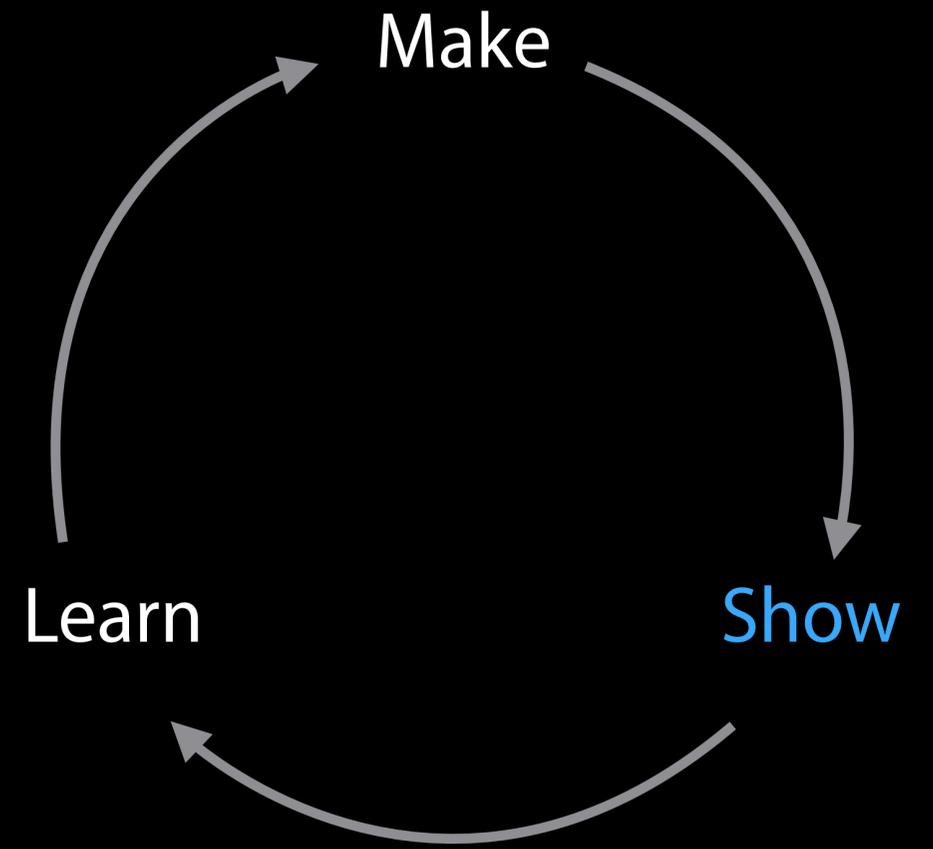
What needs to be more real?

What can be faked?

Where will it be used?

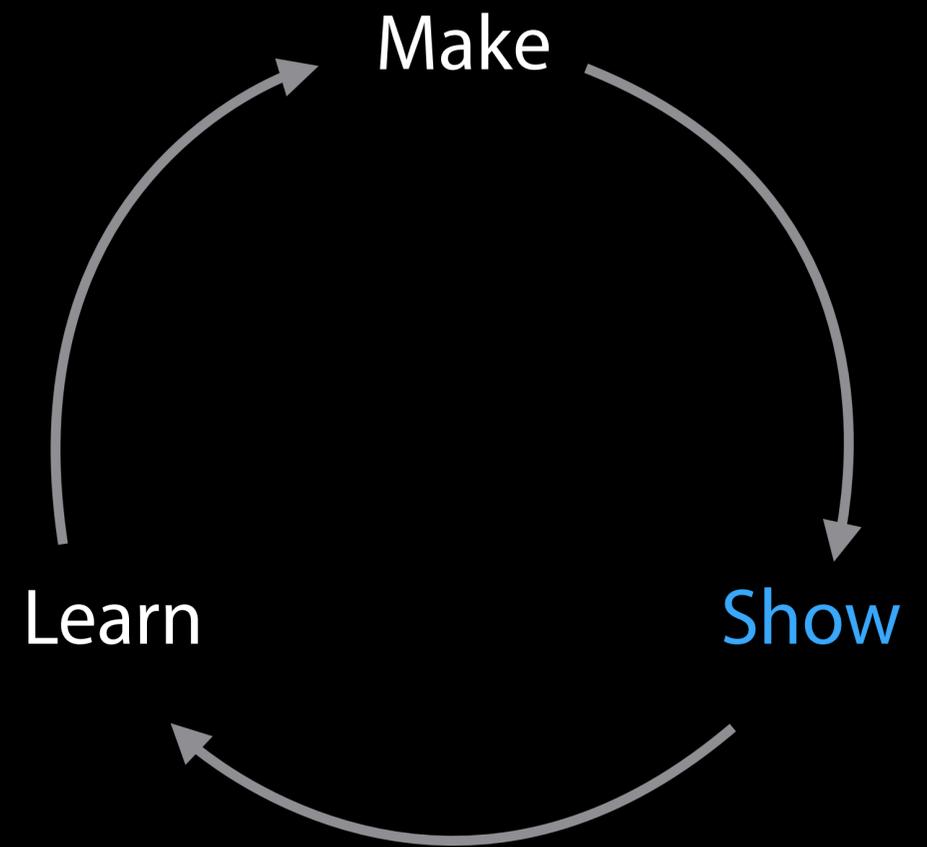


Show People



Show People

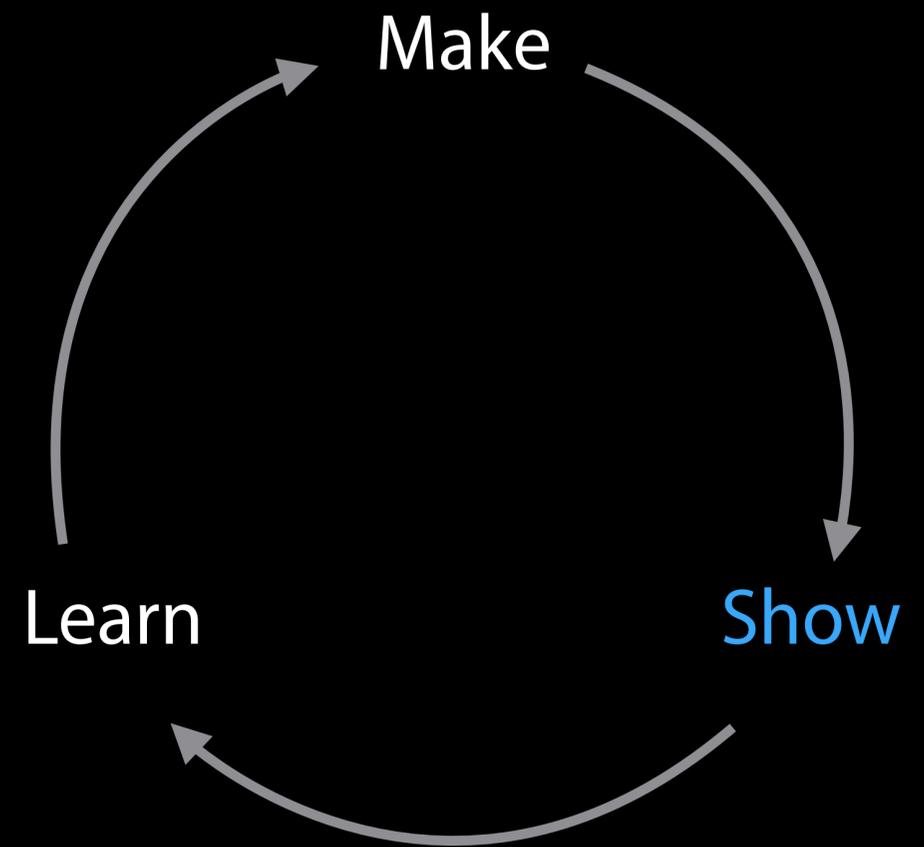
Who?



Show People

Who?

The people your app is for

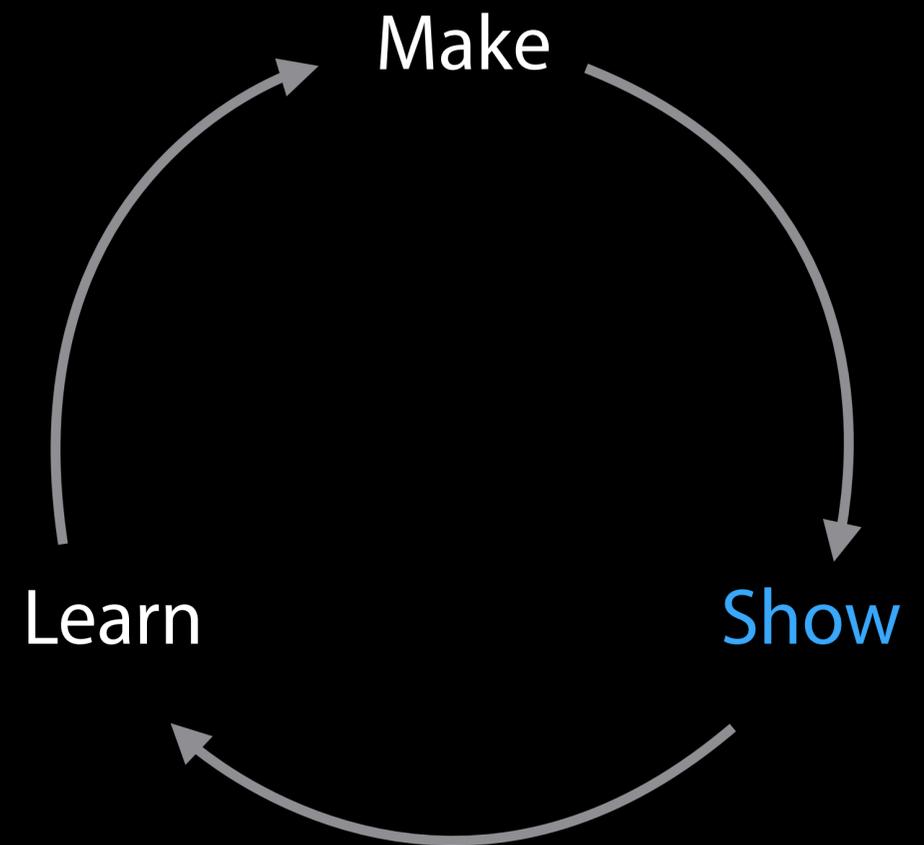


Show People

Who?

The people your app is for

Where?



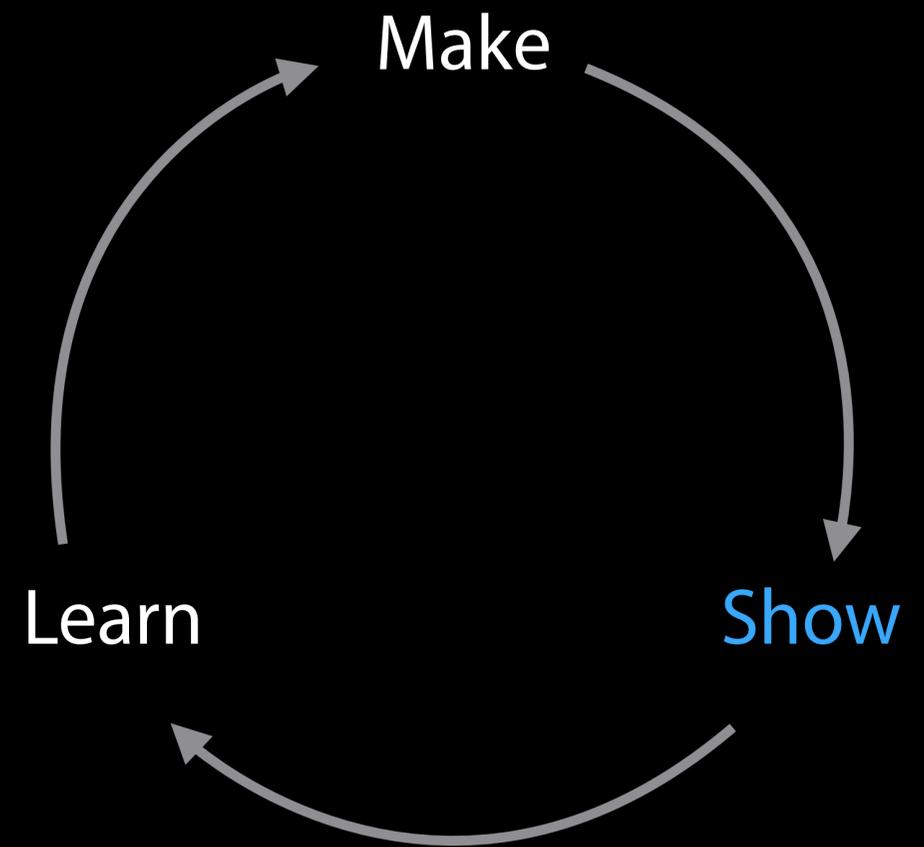
Show People

Who?

The people your app is for

Where?

In the place where they will use it



Show People

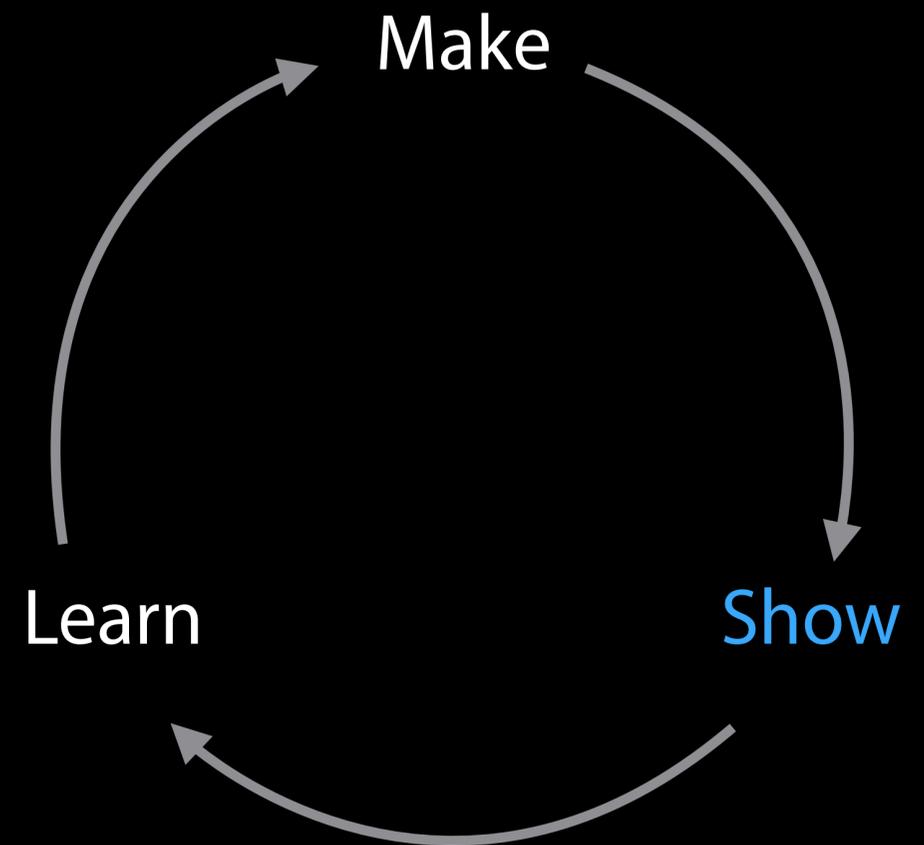
Who?

The people your app is for

Where?

In the place where they will use it

Don't



Show People

Who?

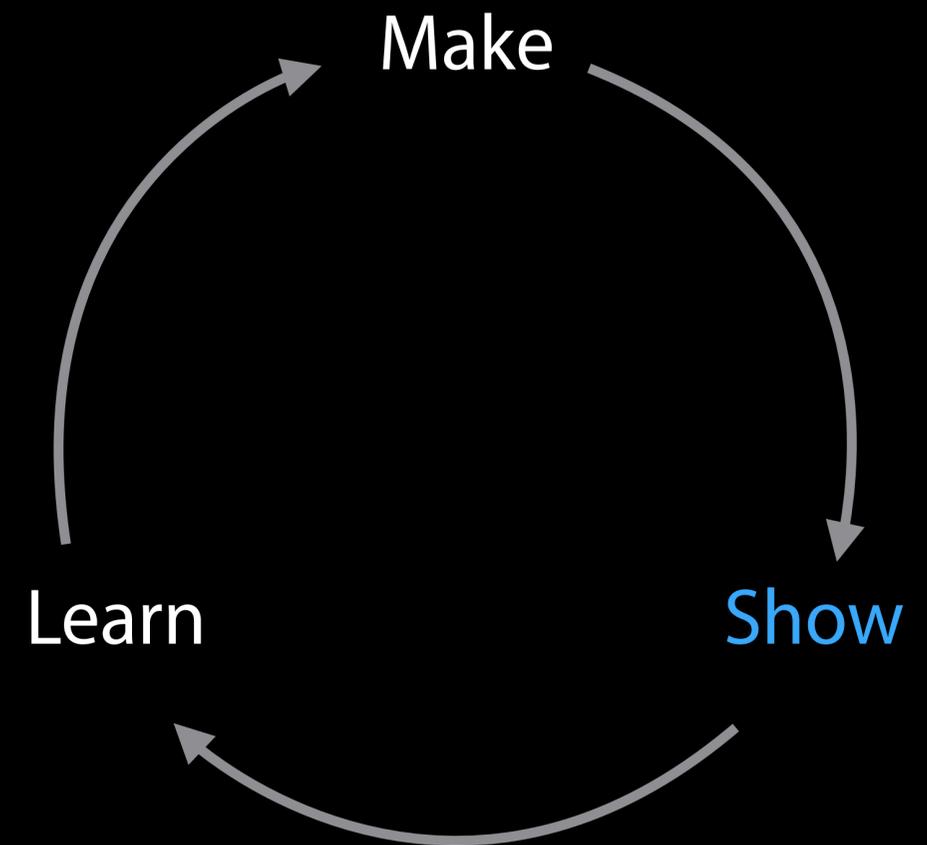
The people your app is for

Where?

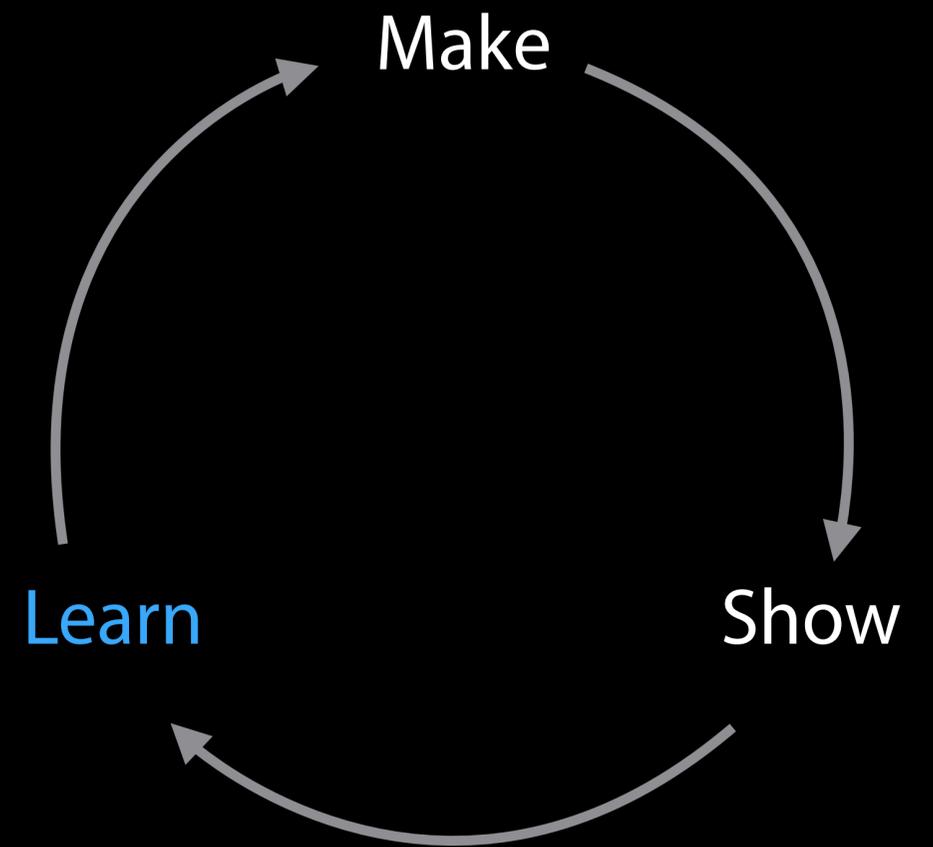
In the place where they will use it

Don't

Argue, defend, dismiss

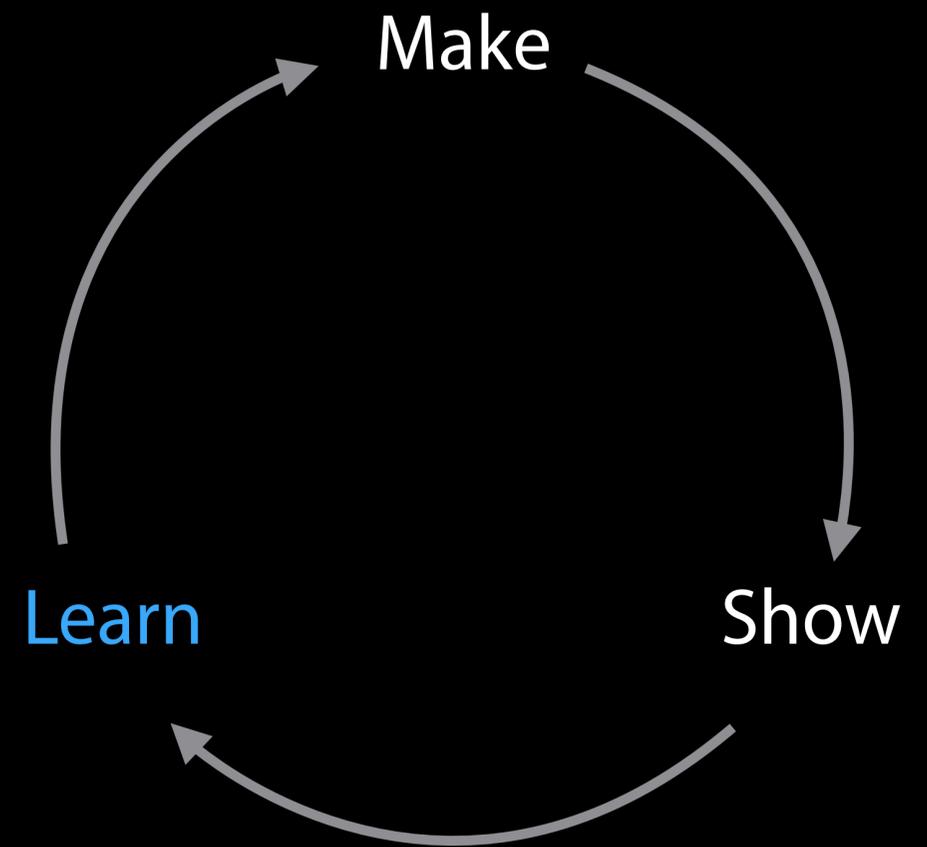


Learn from Their Feedback



Learn from Their Feedback

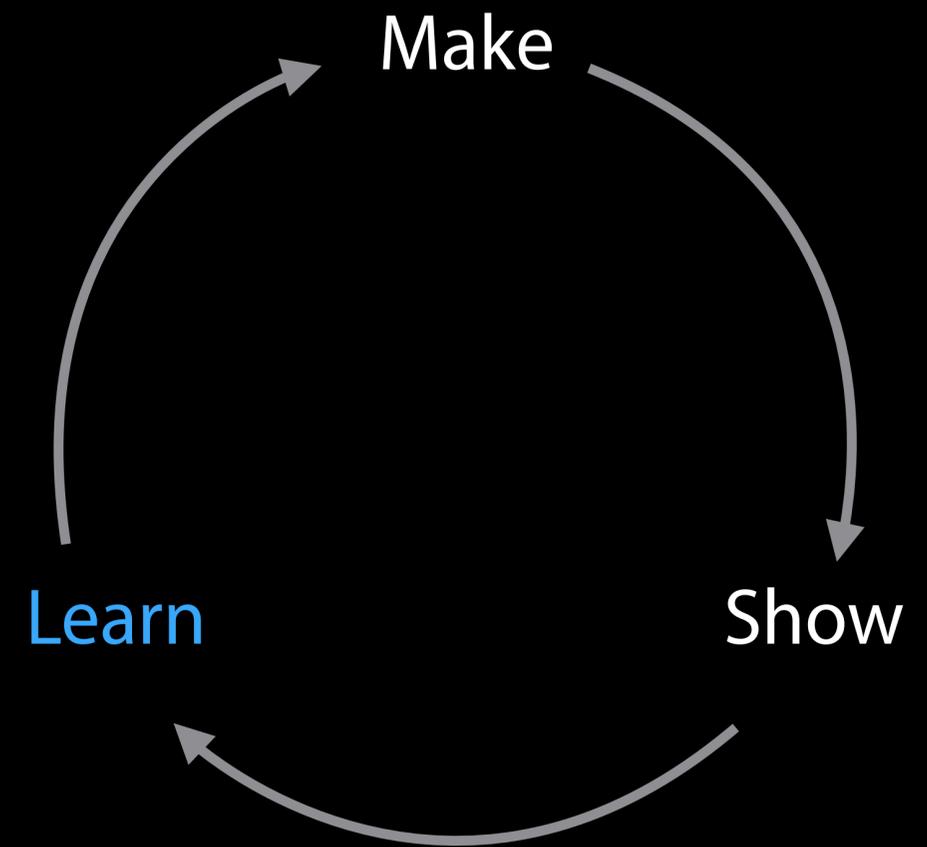
Three questions



Learn from Their Feedback

Three questions

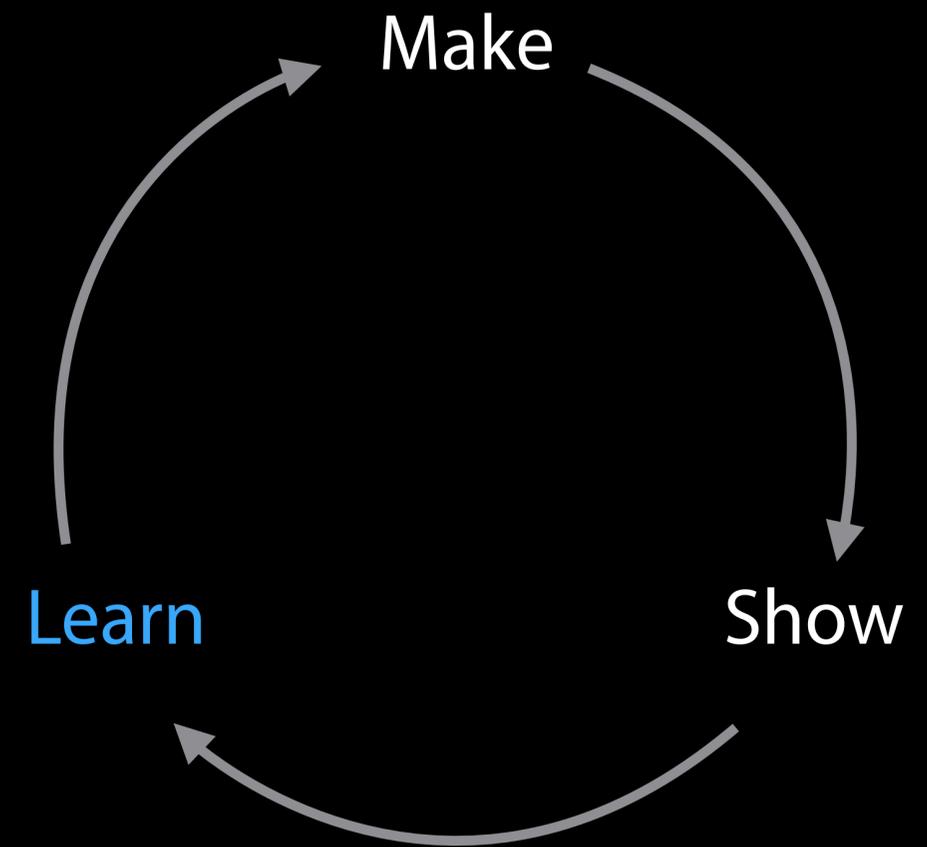
- What's working?



Learn from Their Feedback

Three questions

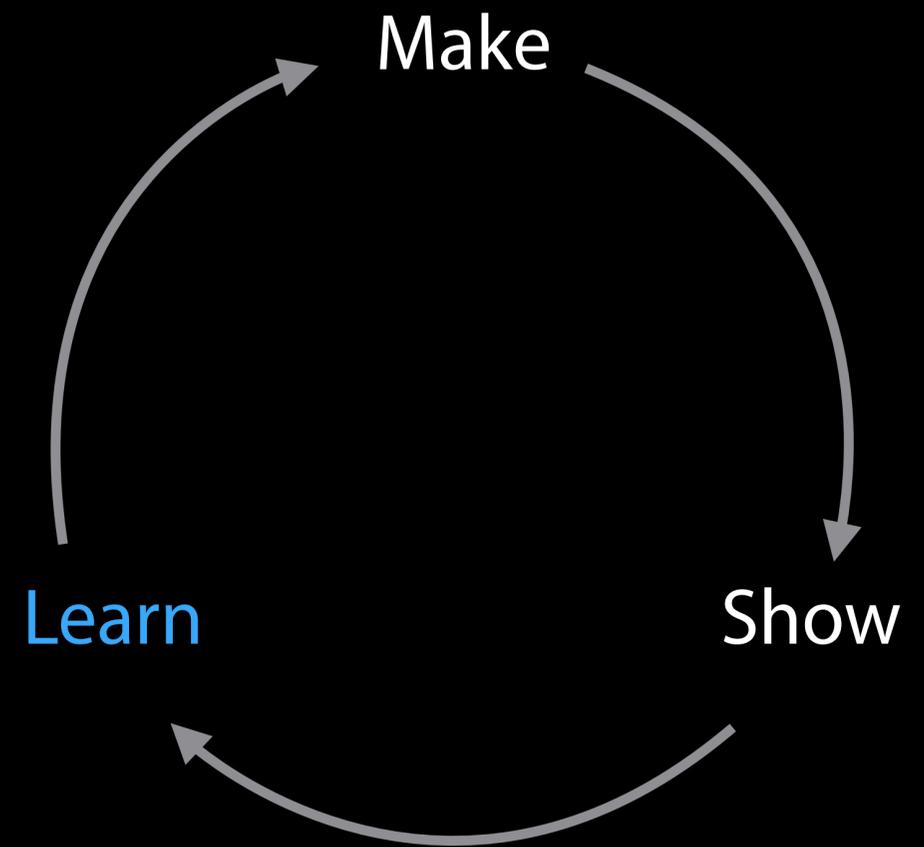
- What's working?
- What's not working?

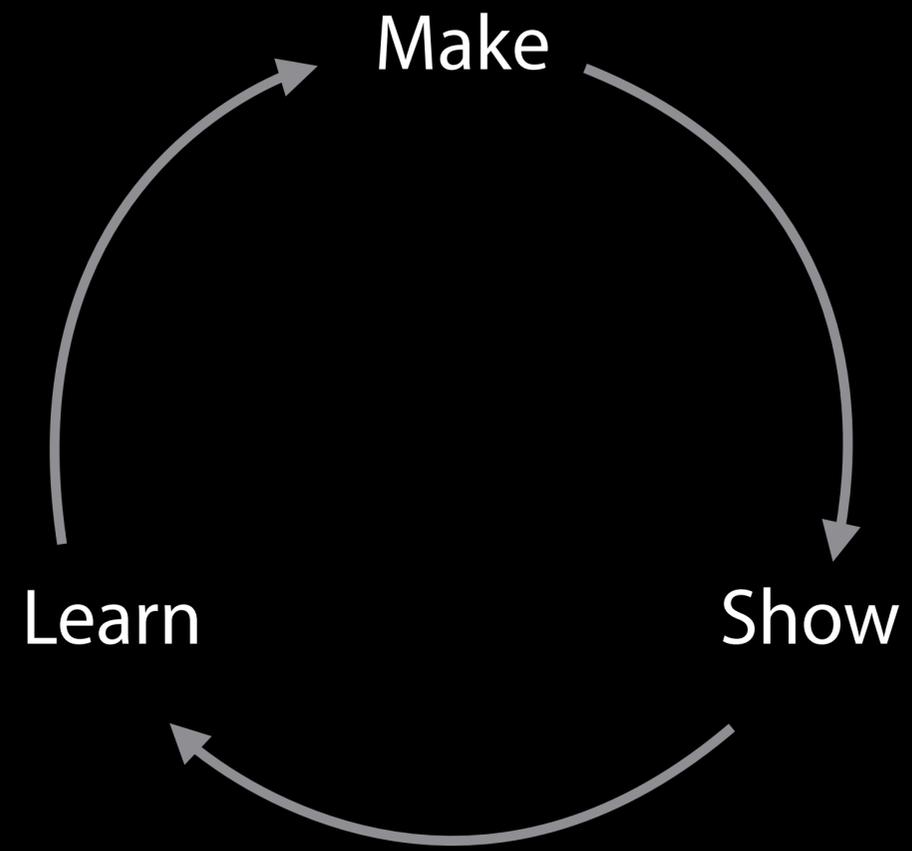


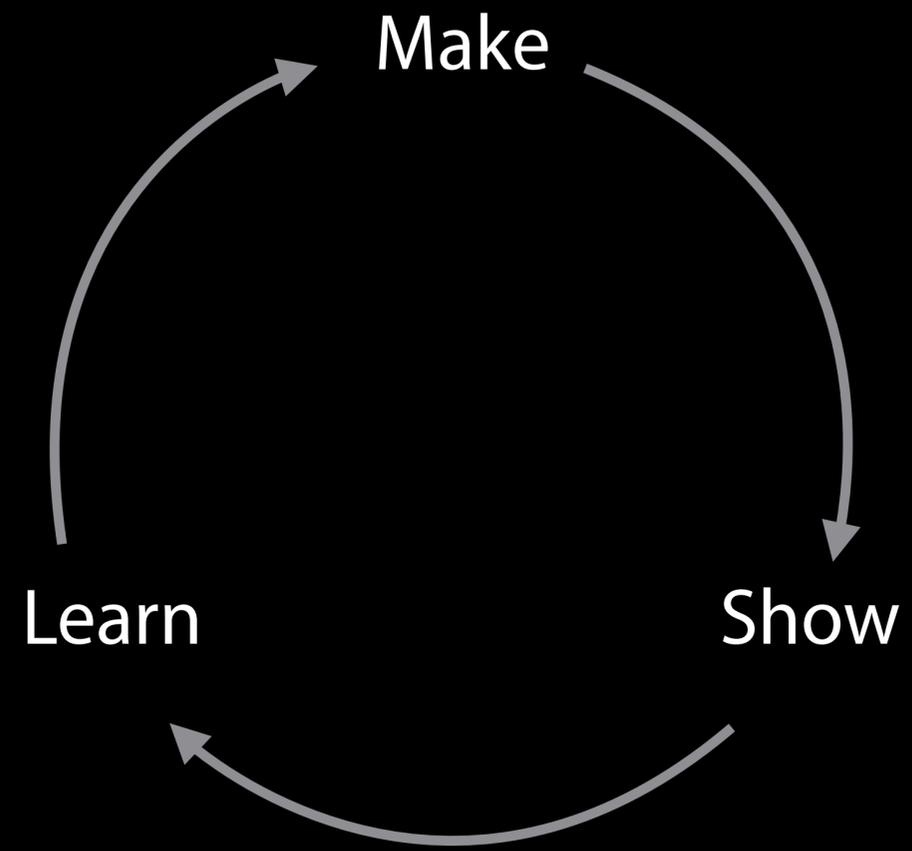
Learn from Their Feedback

Three questions

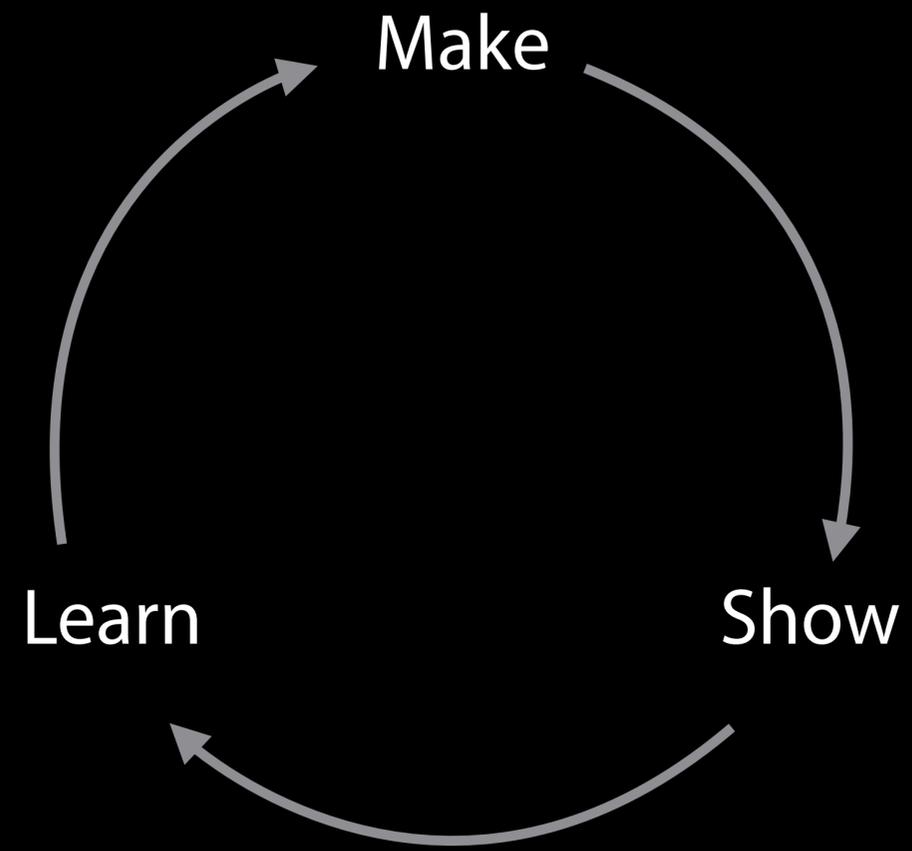
- What's working?
- What's not working?
- What other ideas does this give us?



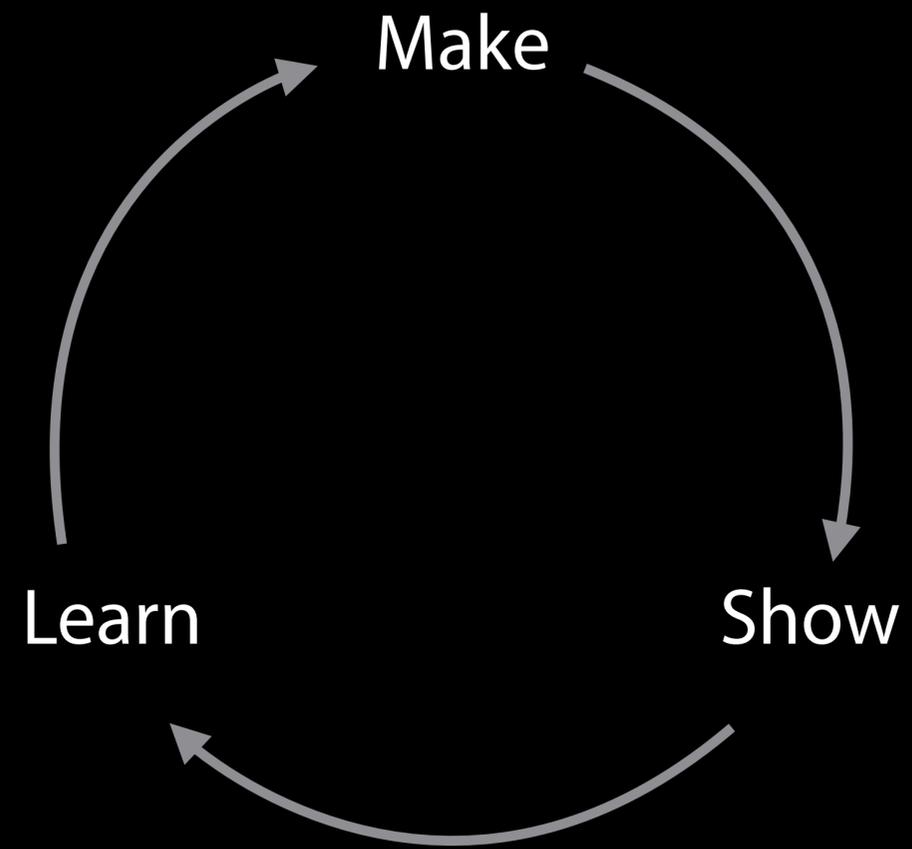




Pictures and animation



Pictures and animation → Behind the curtain



Pictures and animation → Behind the curtain → Interactive and connected

Toastal Service

Toastal Service

The world's first social toaster

Toastal Service

The world's first social toaster

Receive toast messages—Toasties

Toastal Service

The world's first social toaster

Receive toast messages—Toasties

Send toast messages

Toastal Service

The world's first social toaster

Receive toast messages—Toasties

Send toast messages

Magical

Make Fake Hardware and Software

Make Fake Hardware and Software

Fake hardware

Fake app

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

How to tell that there is a new Toastie.

Notification when receiving a Toastie.
Display info about Toastie.

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

How to tell that there is a new Toastie.

Notification when receiving a Toastie.
Display info about Toastie.

What can be faked?

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

How to tell that there is a new Toastie.

Notification when receiving a Toastie.
Display info about Toastie.

What can be faked?

We won't build a toaster.
We'll just use pictures.

Everything will be pictures.

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

How to tell that there is a new Toastie.

Notification when receiving a Toastie.
Display info about Toastie.

What can be faked?

We won't build a toaster.
We'll just use pictures.

Everything will be pictures.

Where will it be used?

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

How to tell that there is a new Toastie.

Notification when receiving a Toastie.
Display info about Toastie.

What can be faked?

We won't build a toaster.
We'll just use pictures.

Everything will be pictures.

Where will it be used?

The kitchen.

Anywhere.









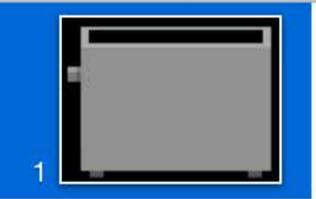
1

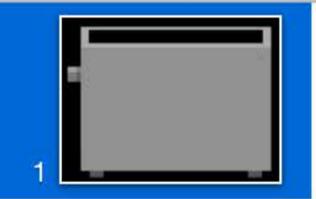
A small thumbnail of the current slide is shown in the top-left corner of the slide navigation pane. It is a black square with the number '1' in the bottom-left corner.

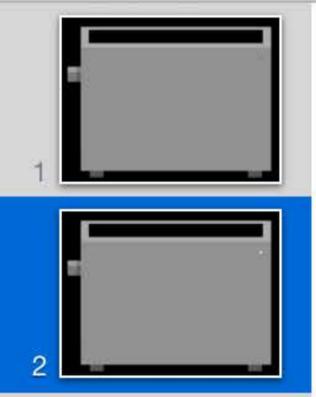


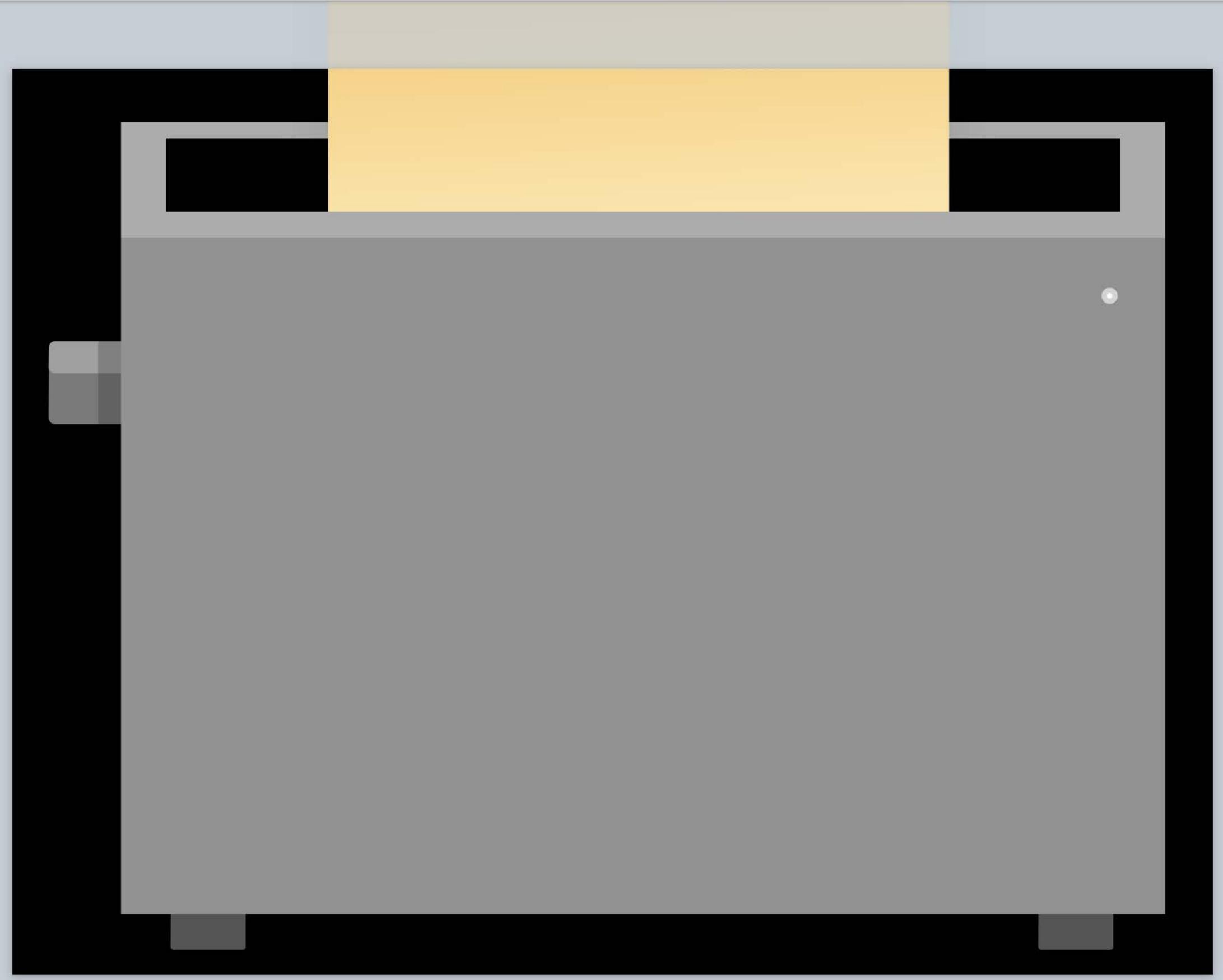
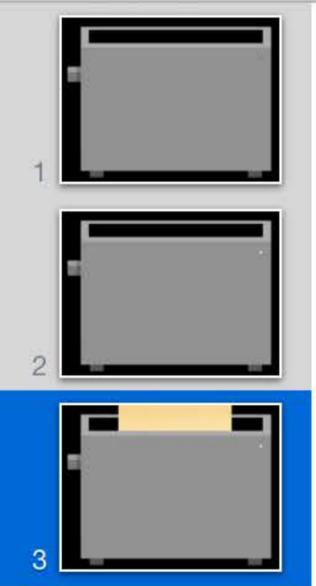








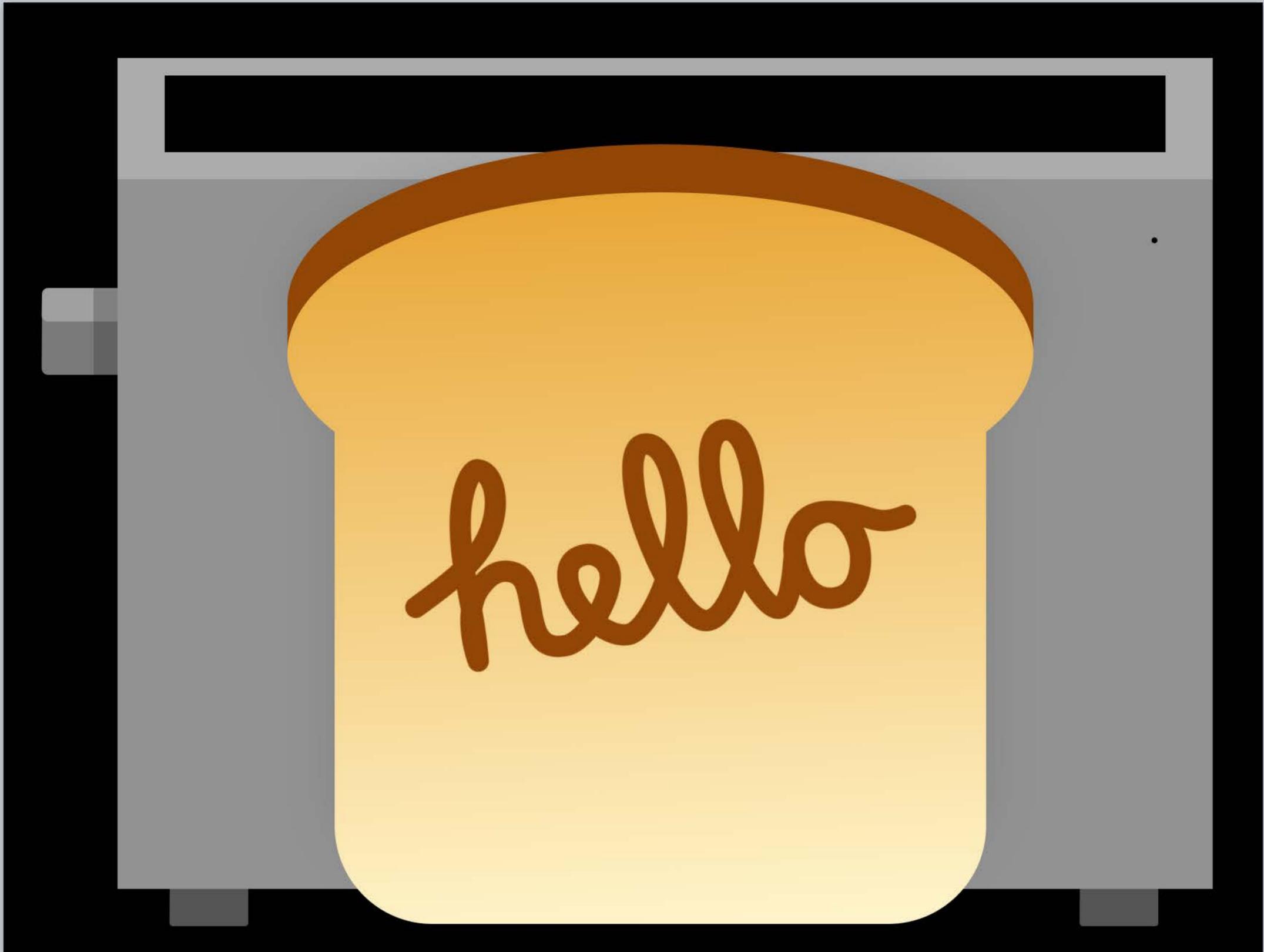




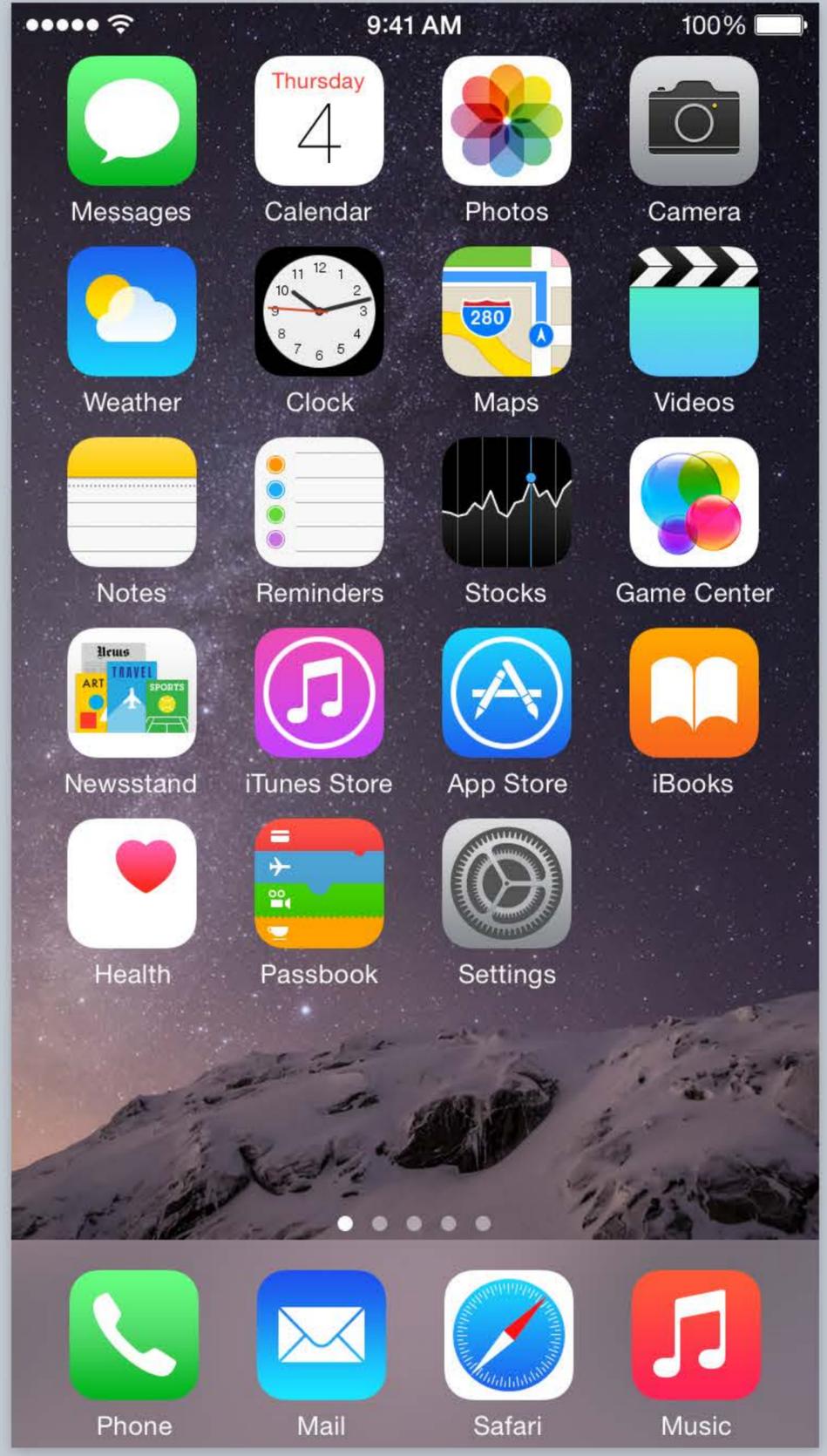
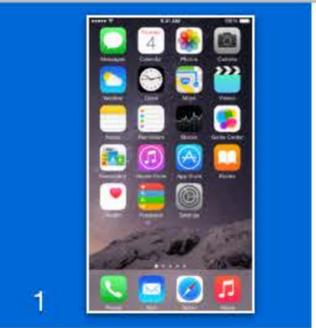
- 1
- 2
- 3
- 4

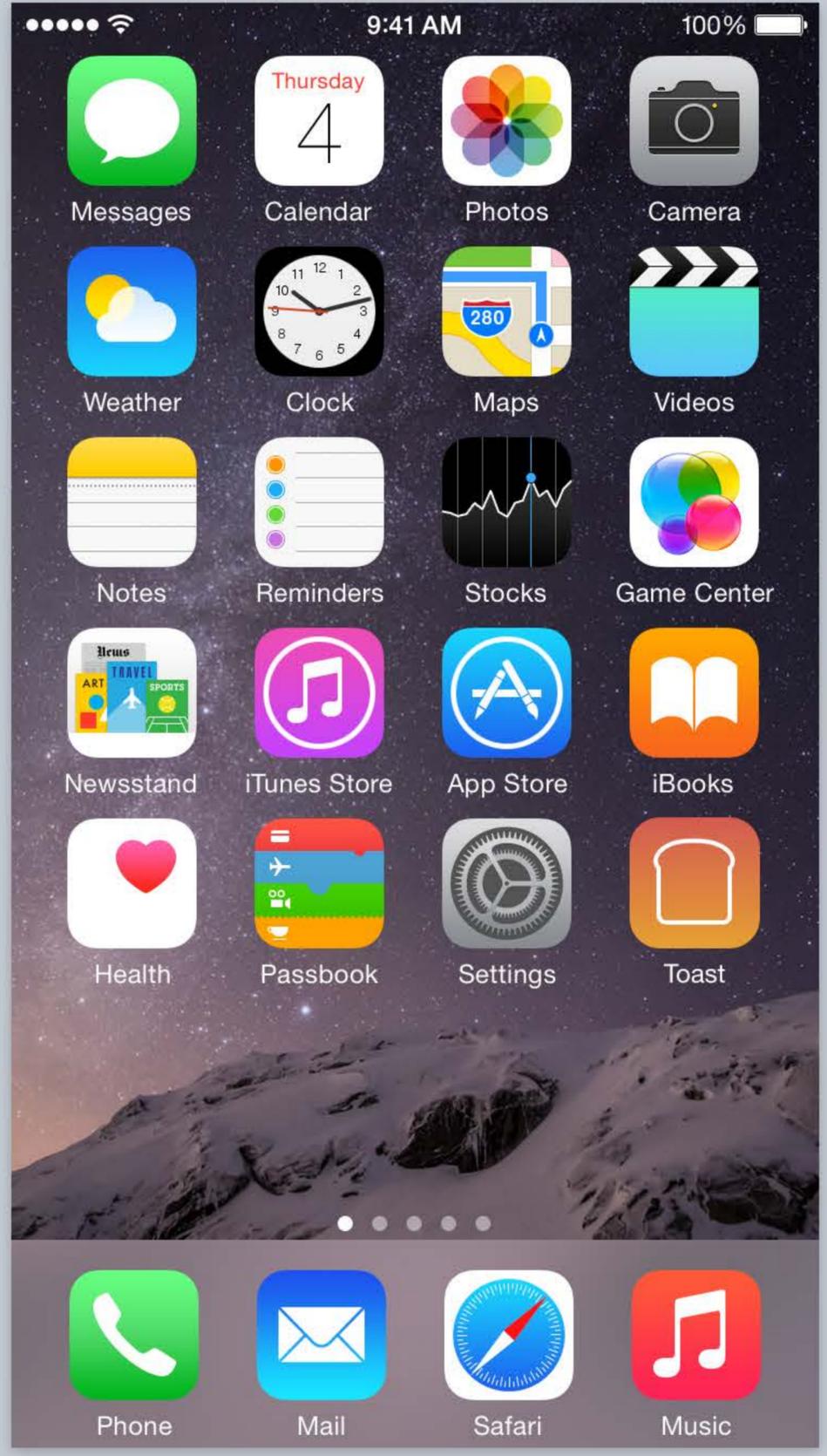
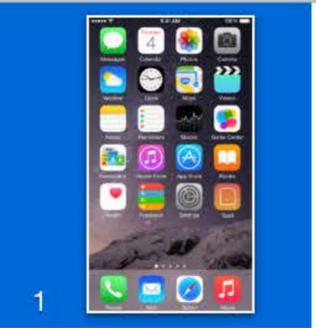


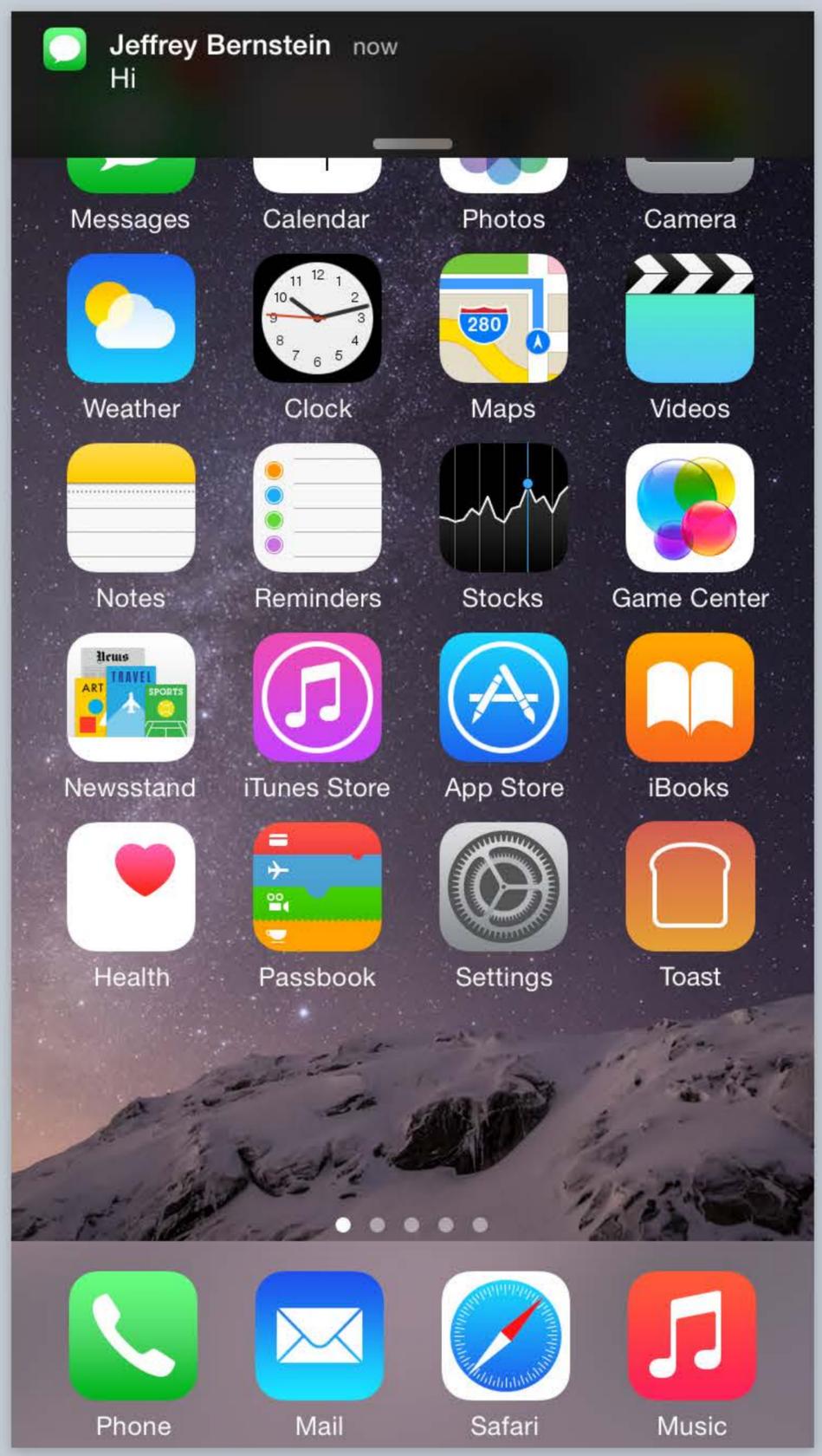
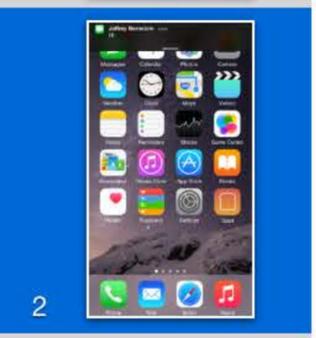
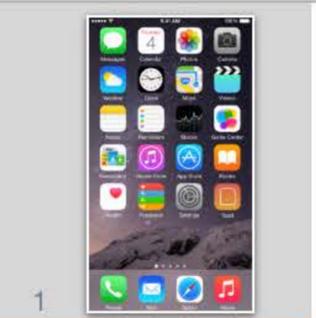
- 1
- 2
- 3
- 4
- 5

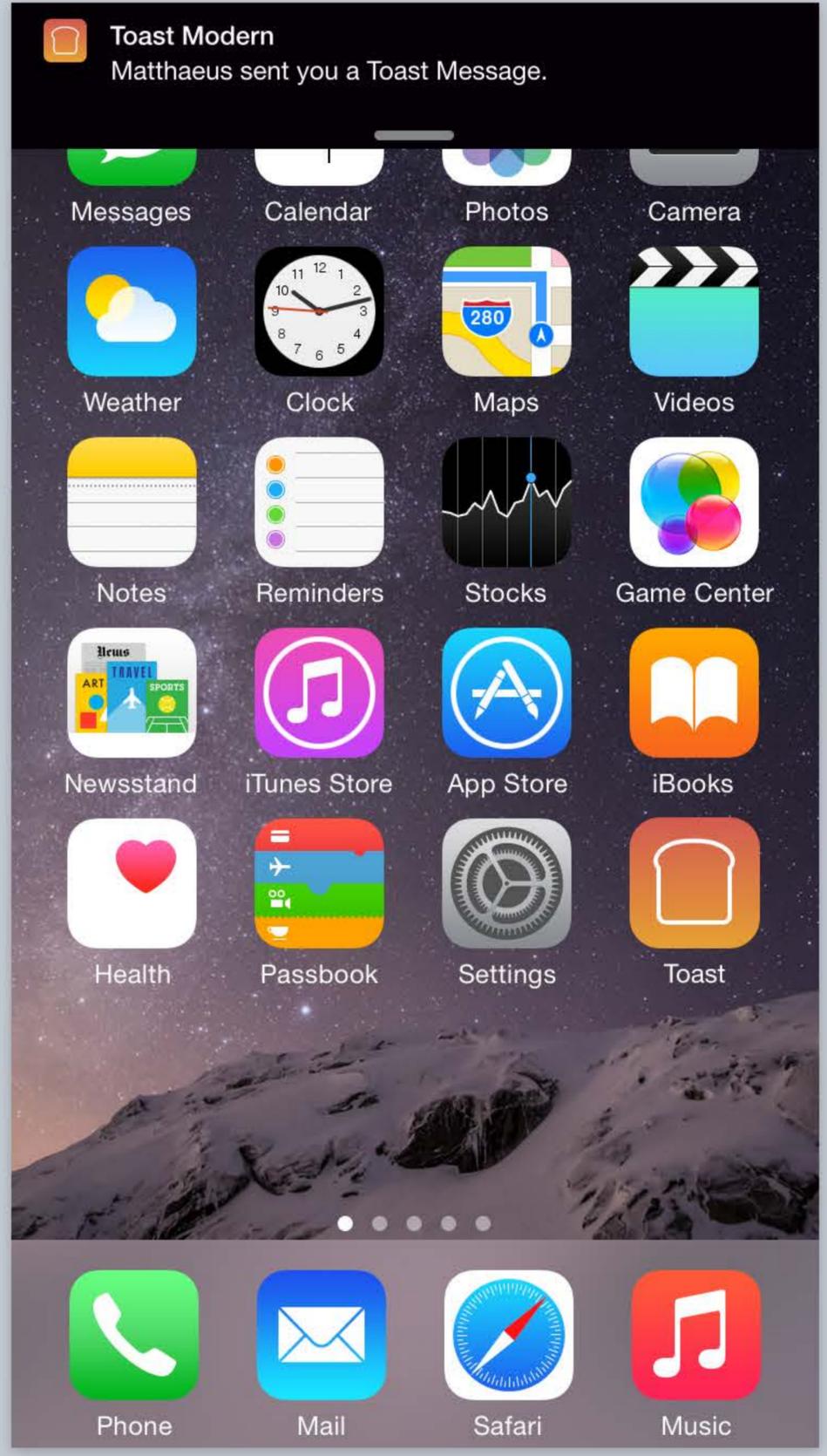
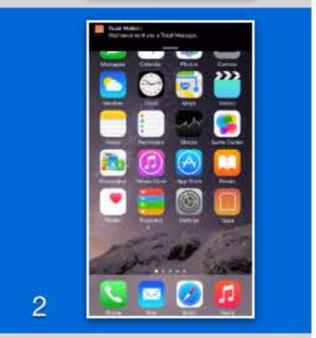
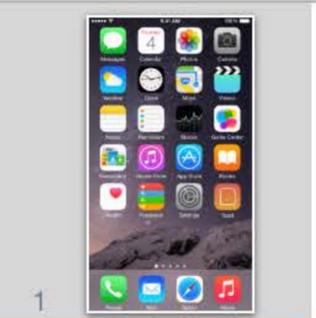














9:41 AM 100%

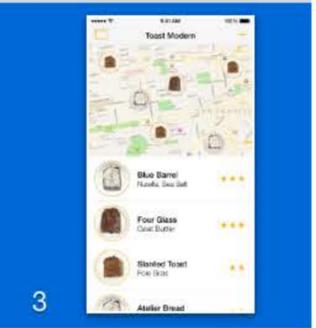
Toast Modern

Blue Barrel
Nutella, Sea Salt ★★★★★

Four Glass
Goat Butter ★★★★★

Slanted Toast
Foie Gras ★★★

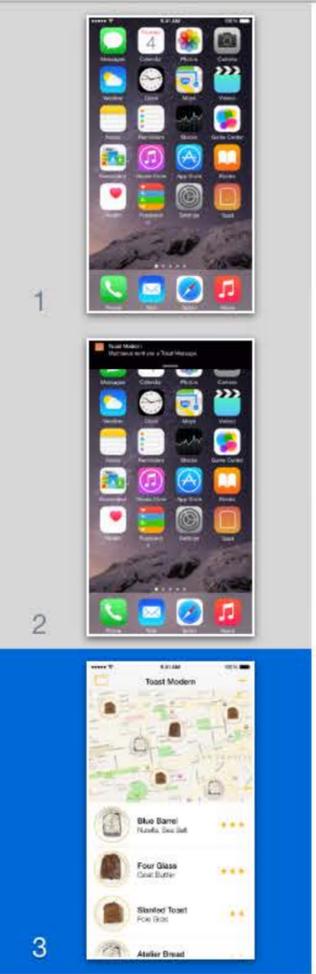
Atelier Bread
Creme Fraiche ★★



9:41 AM 100%

Toast Modern

- 
Blue Barrel
 Nutella, Sea Salt ★★★★★
- 
Four Glass
 Goat Butter ★★★★★
- 
Slanted Toast
 Foie Gras ★★★
- 
Atelier Bread
 Creme Fraiche ★★



9:41 AM 100%

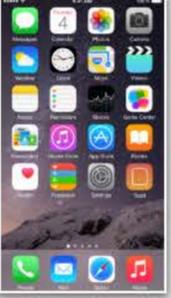
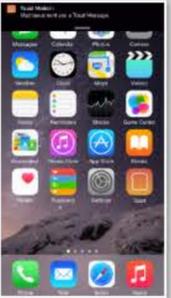
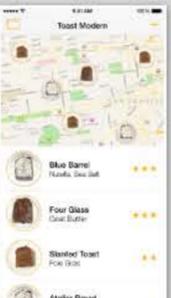
Toast Modern

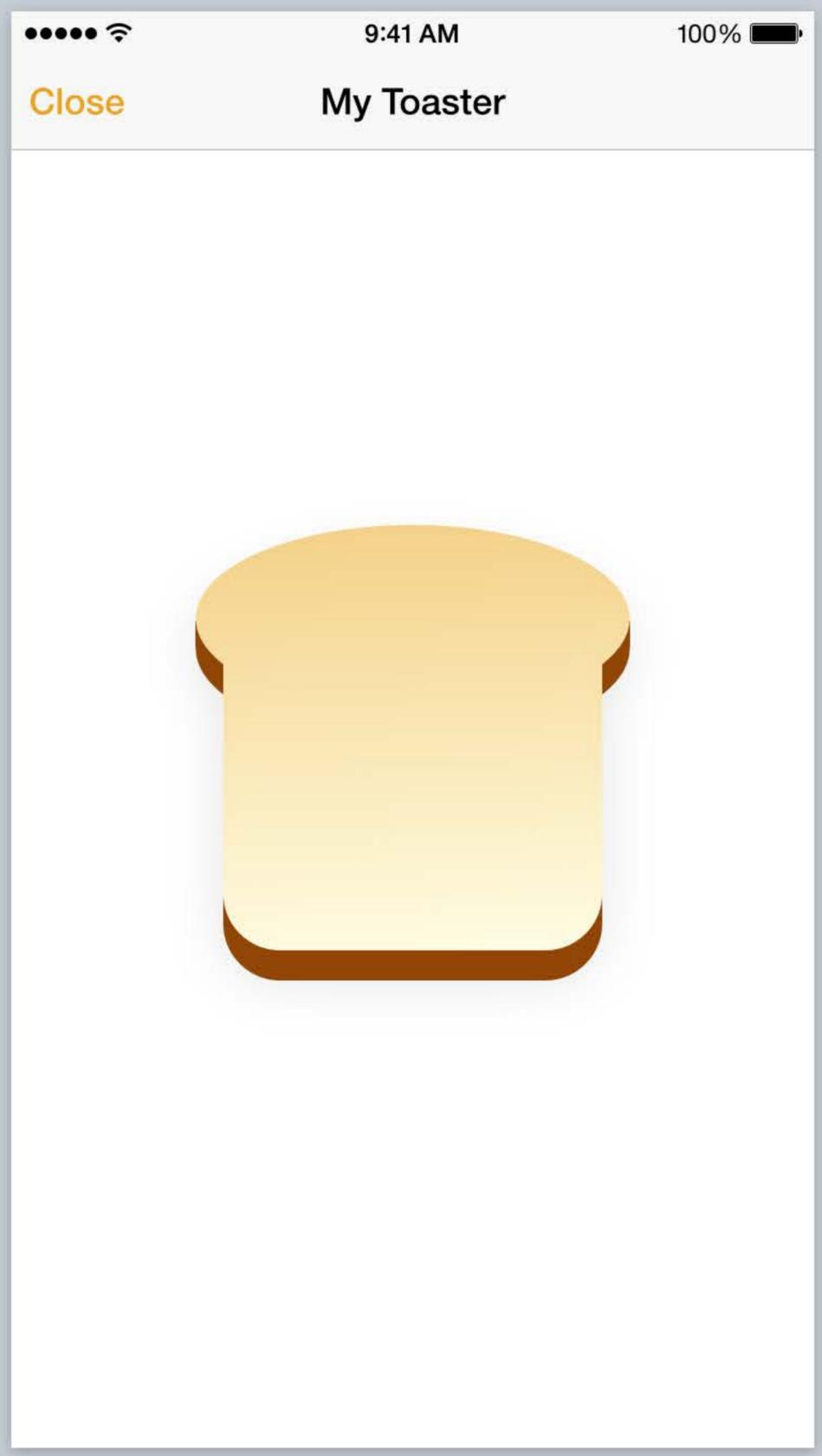
Blue Barrel
Nutella, Sea Salt ★★★★★

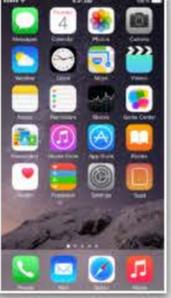
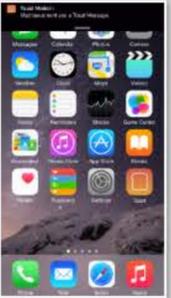
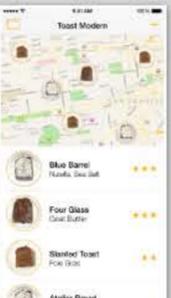
Four Glass
Goat Butter ★★★★★

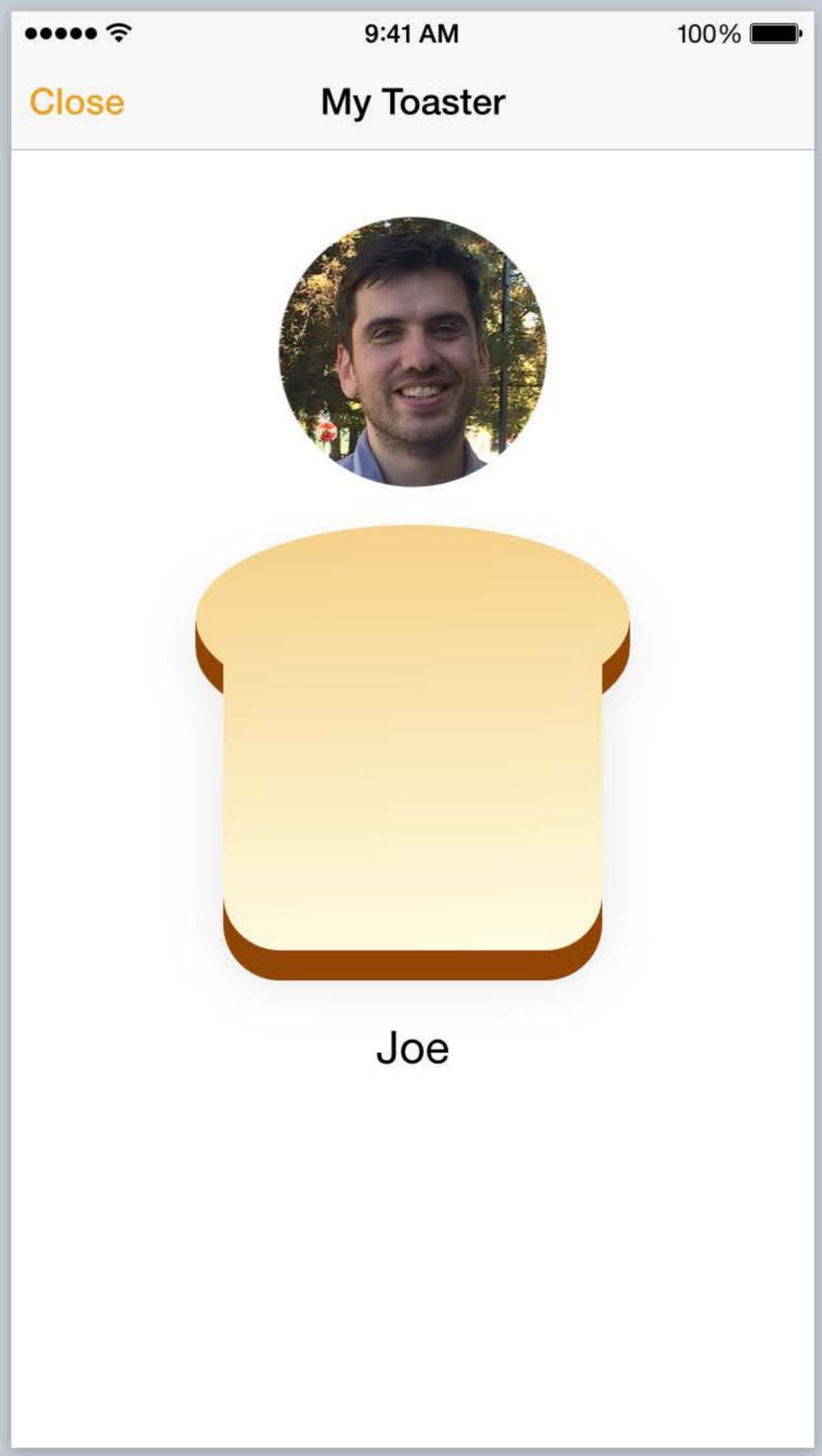
Slanted Toast
Foie Gras ★★★

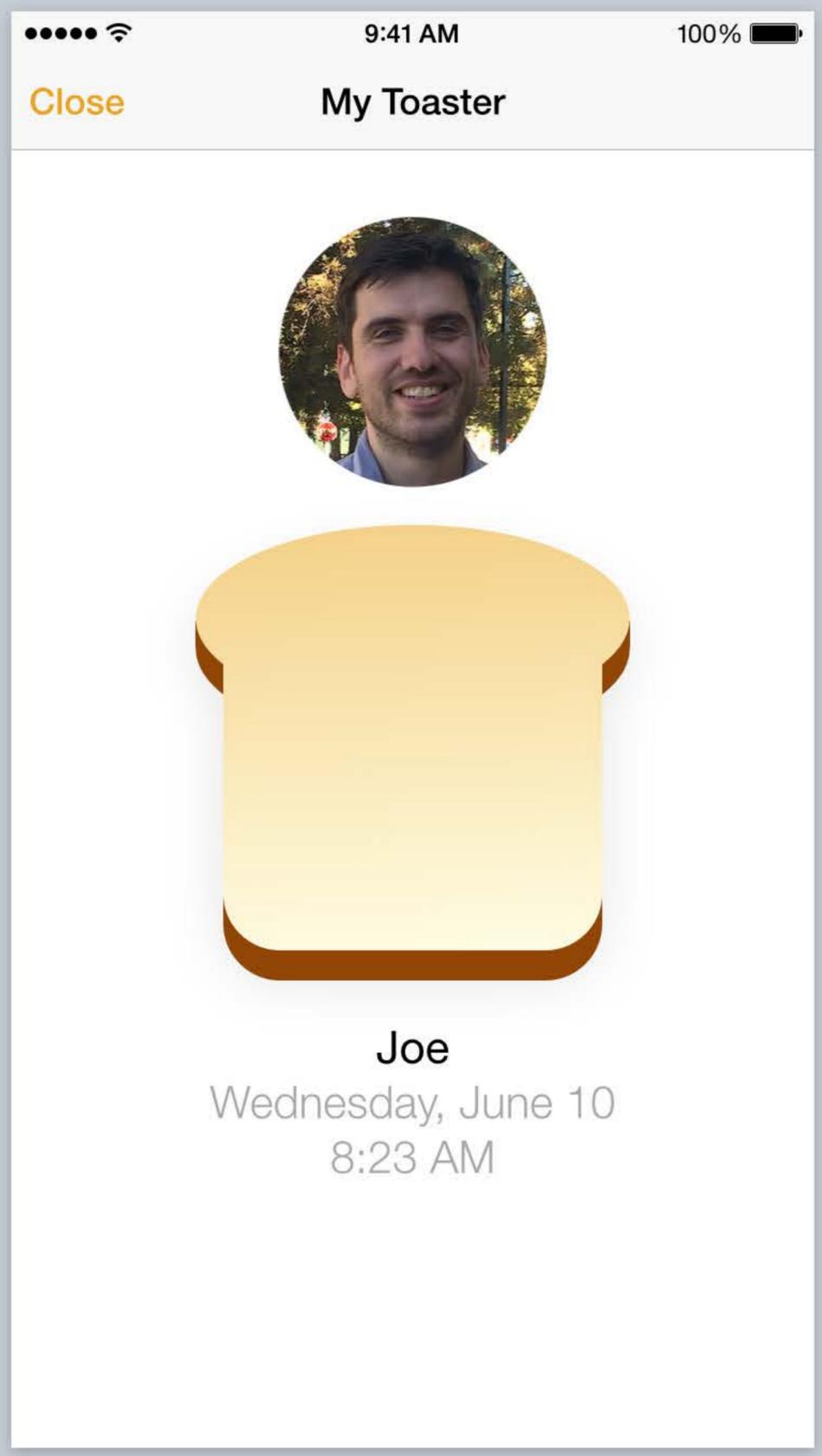
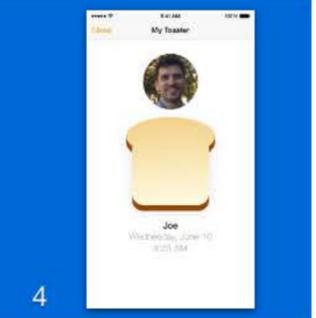
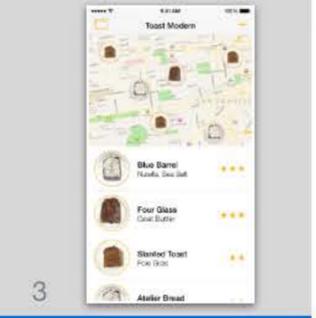
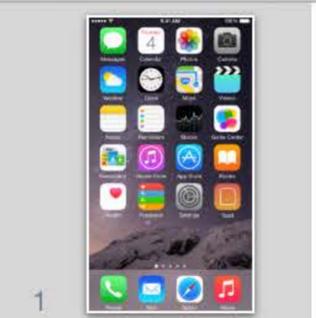
Atelier Bread
Creme Fraiche ★★

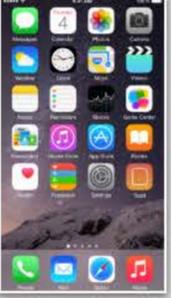
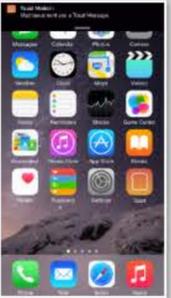
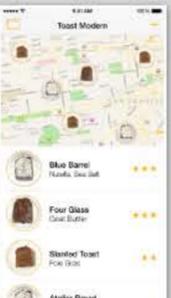
- 1 
- 2 
- 3 
- 4 



- 1 
- 2 
- 3 
- 4 

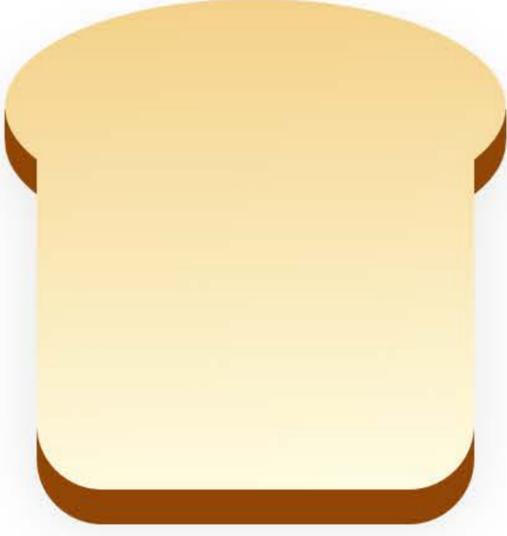




- 1 
- 2 
- 3 
- 4 

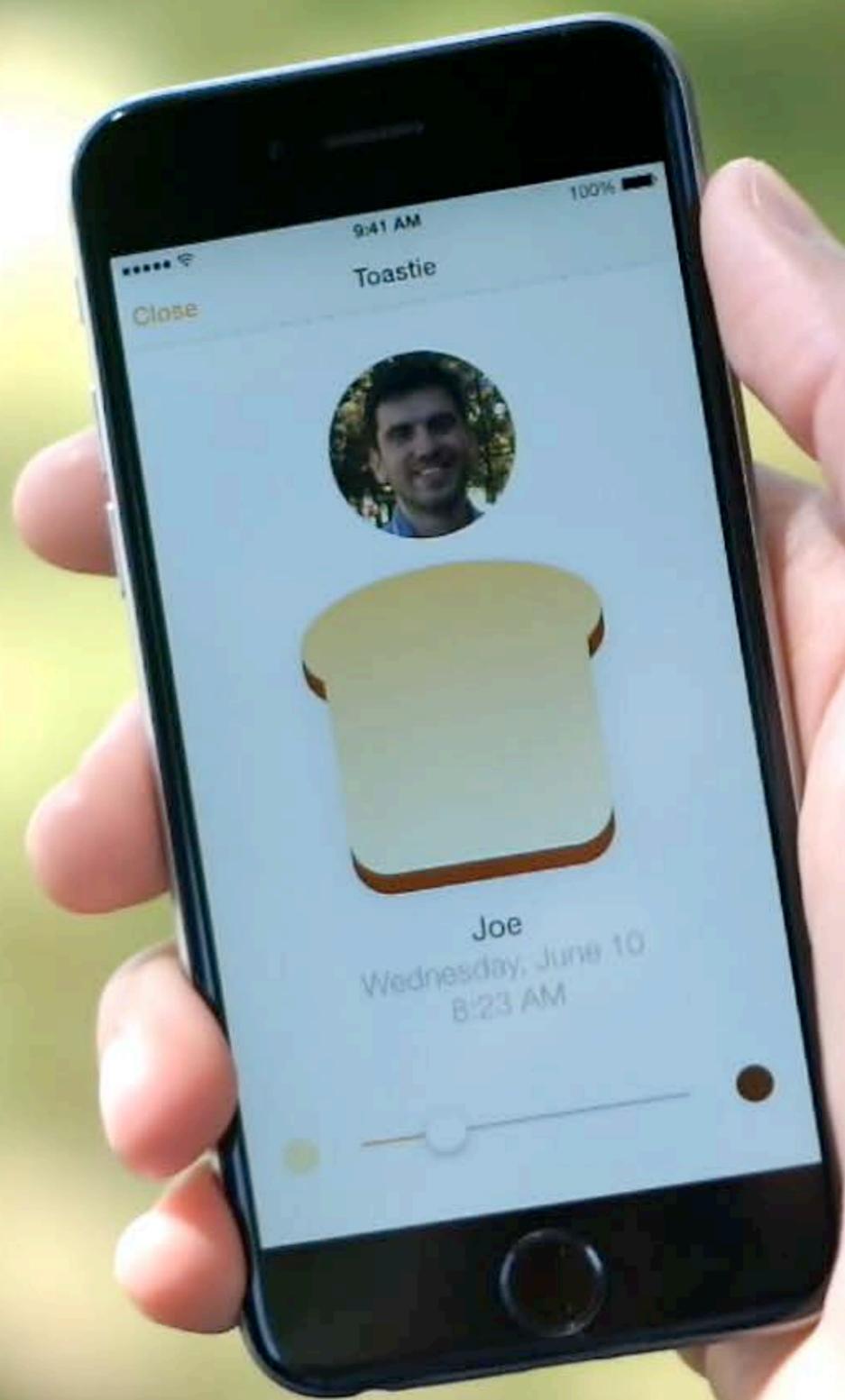
9:41 AM 100%

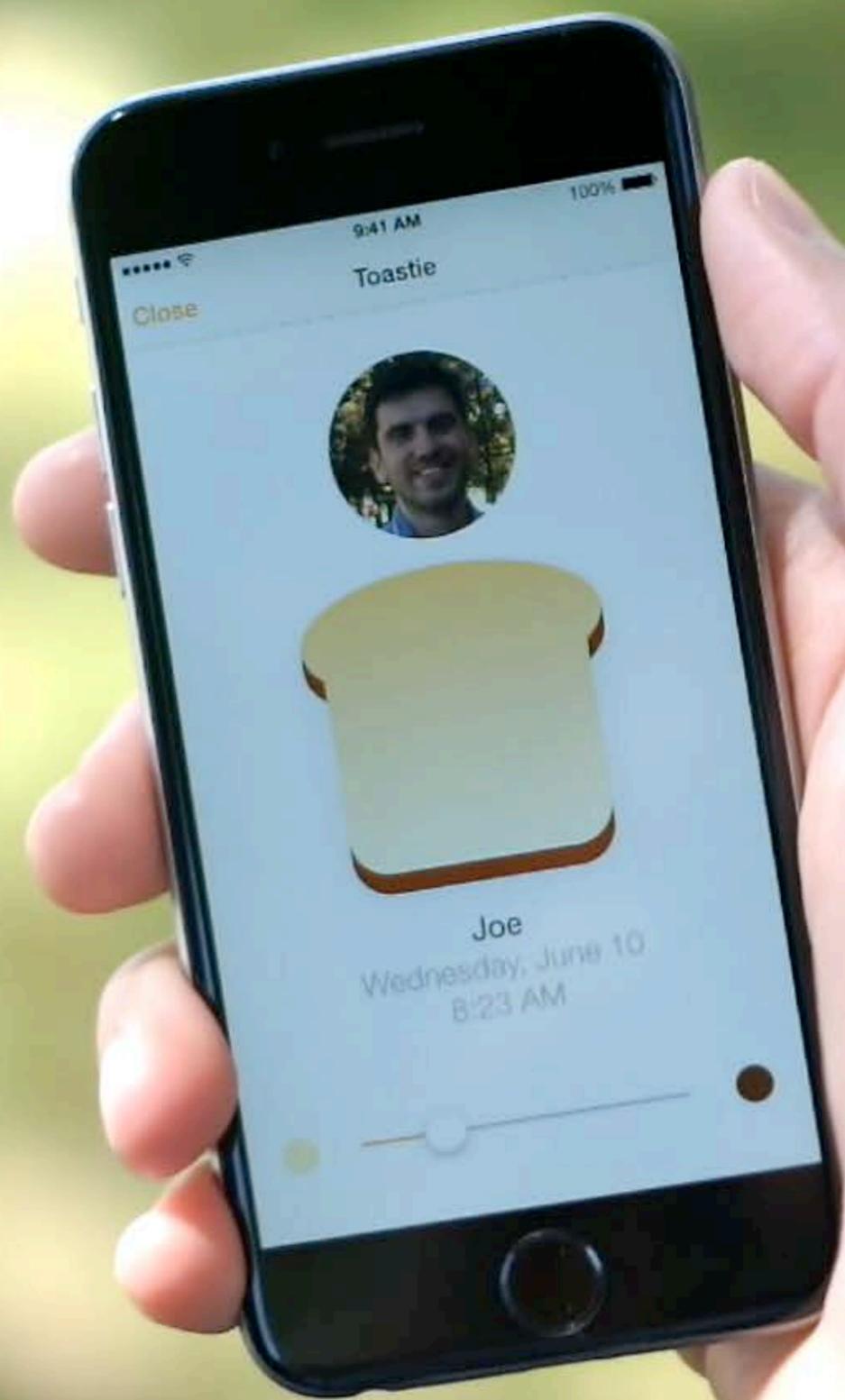
Close My Toaster



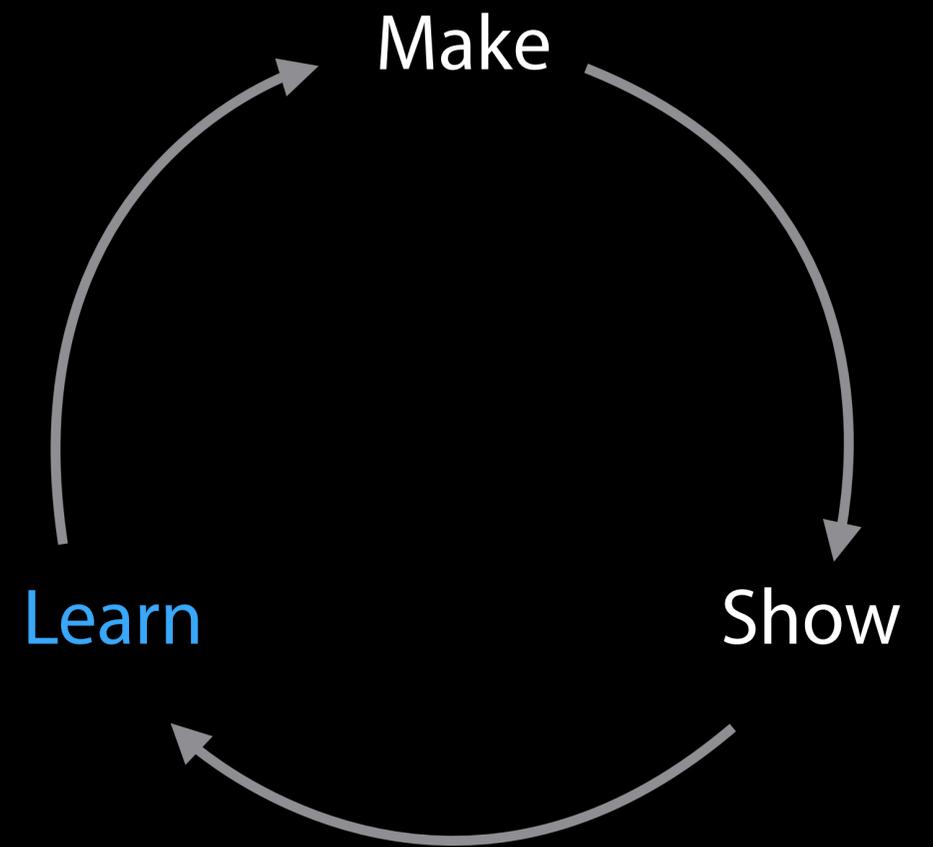
Joe
Wednesday, June 10
8:23 AM





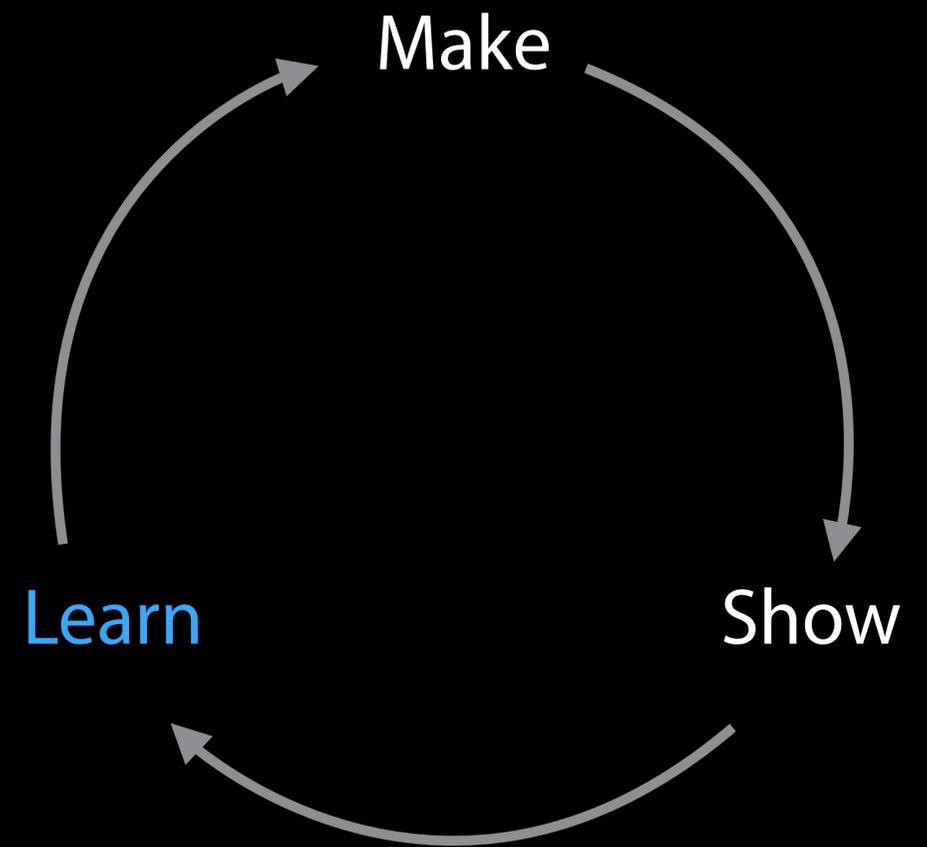


Learn from Feedback



Learn from Feedback

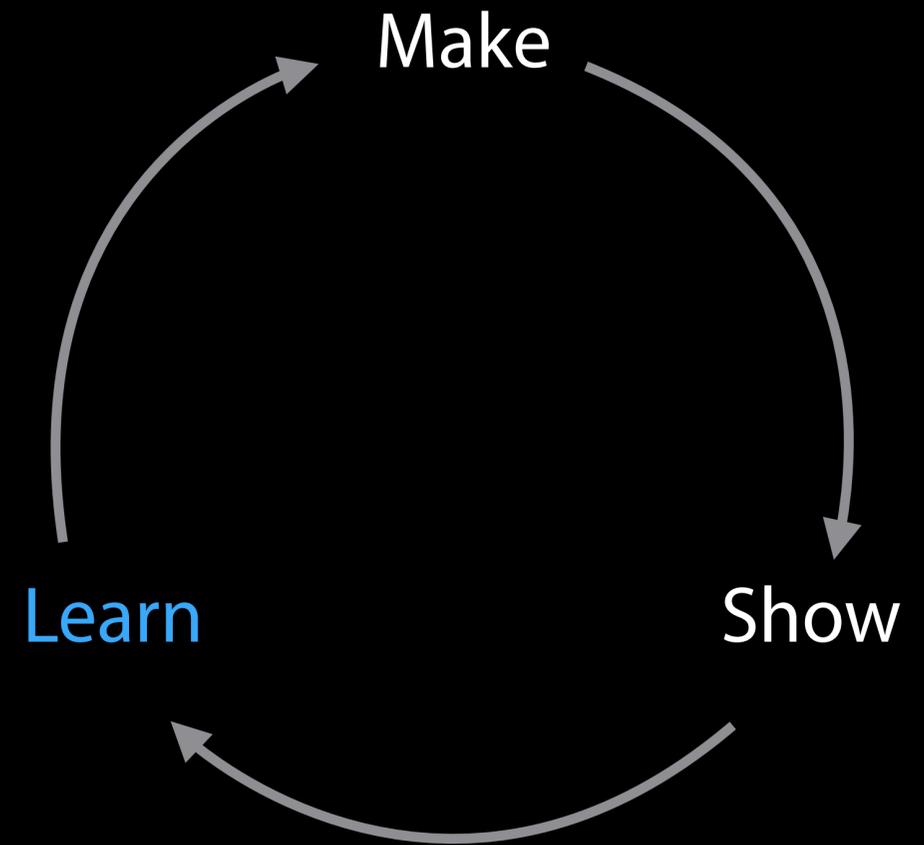
What's working?



Learn from Feedback

What's working?

The LED on the toaster works well. The notification lets us know that there's a new Toastie wherever we are

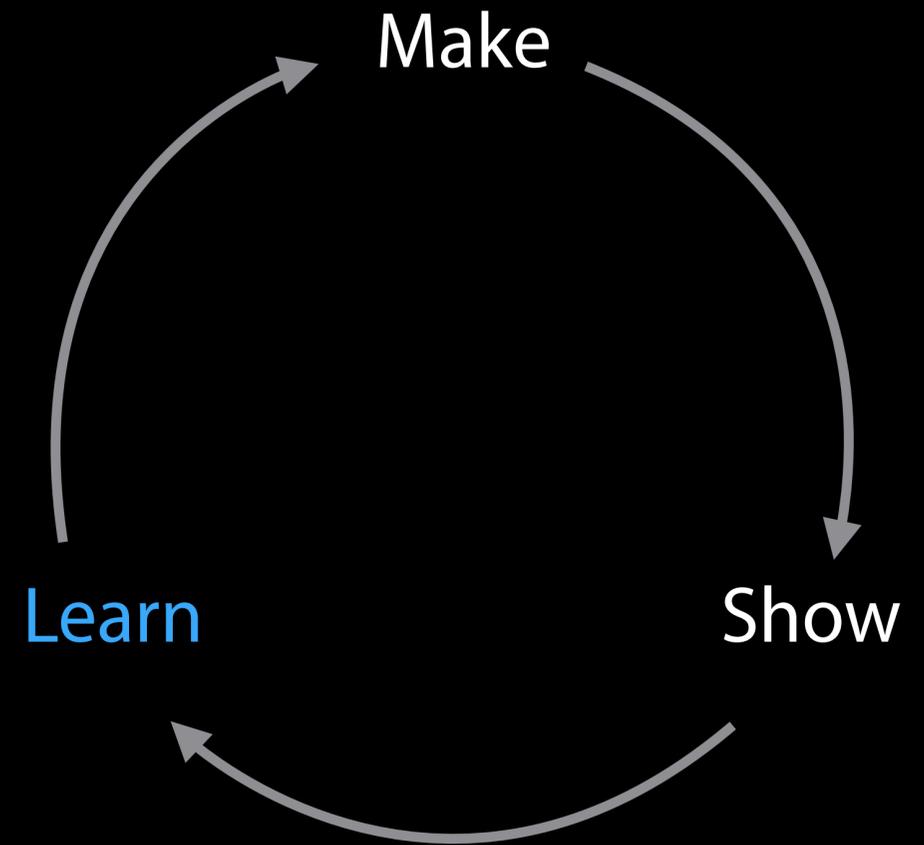


Learn from Feedback

What's working?

The LED on the toaster works well. The notification lets us know that there's a new Toastie wherever we are

What's not working?



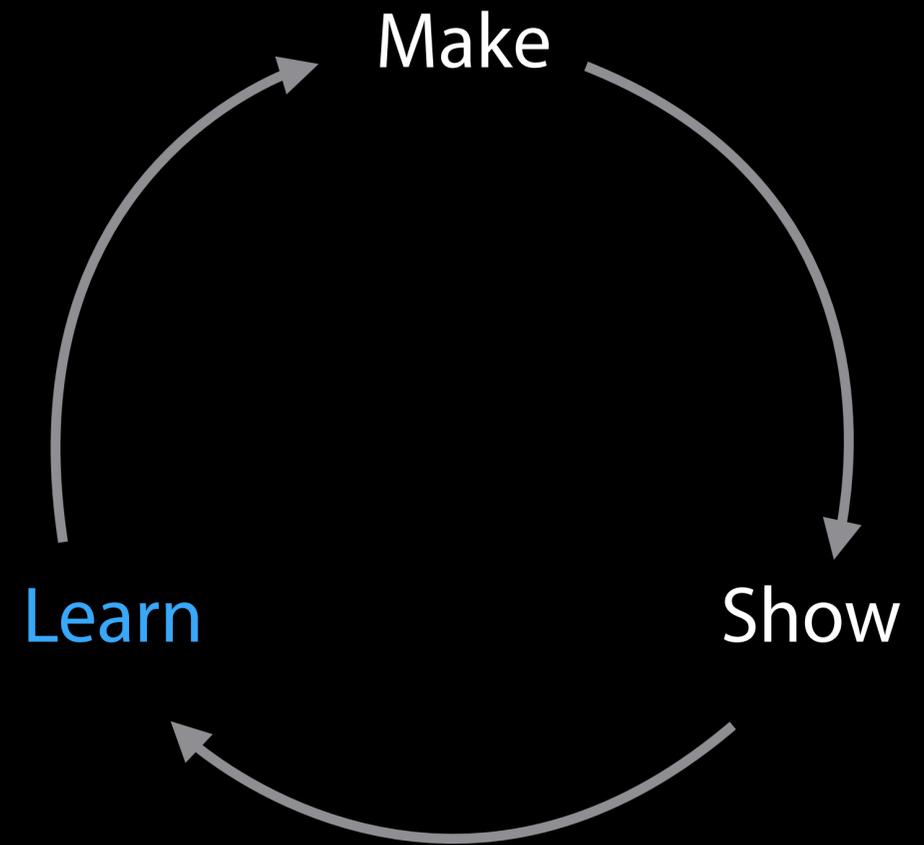
Learn from Feedback

What's working?

The LED on the toaster works well. The notification lets us know that there's a new Toastie wherever we are

What's not working?

What if you have more than one Toastie?



Learn from Feedback

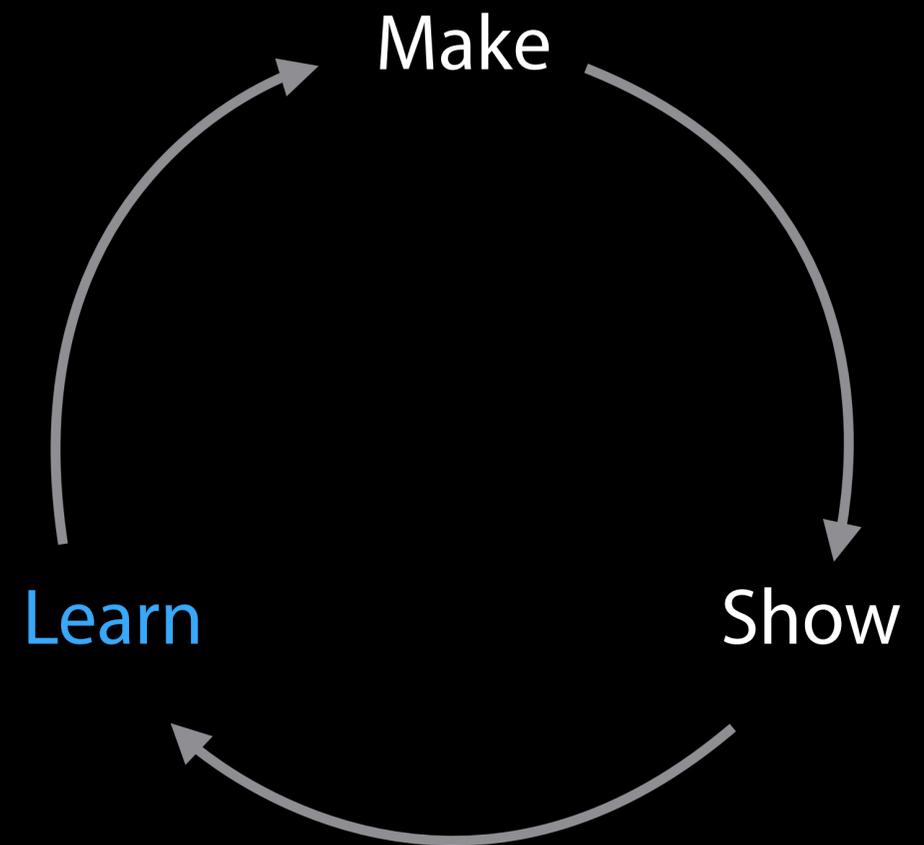
What's working?

The LED on the toaster works well. The notification lets us know that there's a new Toastie wherever we are

What's not working?

What if you have more than one Toastie?

What ideas does this give us?



Learn from Feedback

What's working?

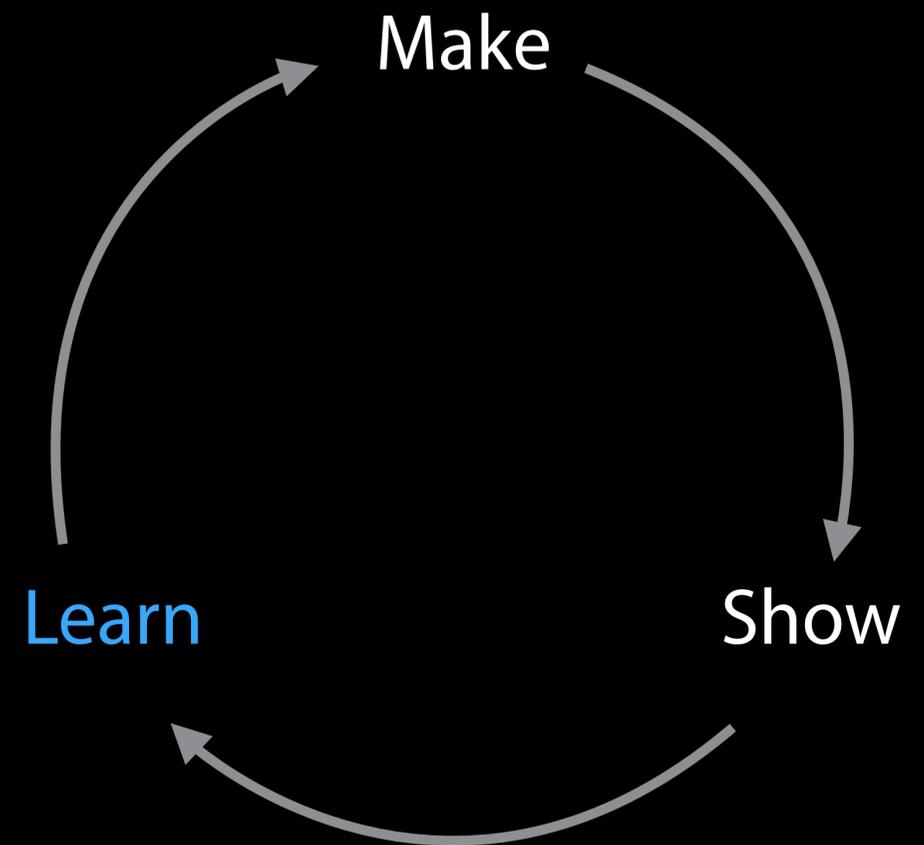
The LED on the toaster works well. The notification lets us know that there's a new Toastie wherever we are

What's not working?

What if you have more than one Toastie?

What ideas does this give us?

Upgrade the LED on the toaster. App should show a list of incoming Toasties



Make Fake Hardware and Software

Make Fake Hardware and Software

Fake hardware

Fake app

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Handle multiple new Toasties.
Toast the right one.

Browse Toasties in the inbox.

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Handle multiple new Toasties.
Toast the right one.

Browse Toasties in the inbox.

What can be faked?

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Handle multiple new Toasties.
Toast the right one.

Browse Toasties in the inbox.

What can be faked?

The connection between the app
and the toaster.
Anything related to toasting.

Sending data back and forth.
UI is still just pictures and
animations.

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Handle multiple new Toasties.
Toast the right one.

Browse Toasties in the inbox.

What can be faked?

The connection between the app
and the toaster.
Anything related to toasting.

Sending data back and forth.
UI is still just pictures and
animations.

Where will it be used?

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Handle multiple new Toasties.
Toast the right one.

Browse Toasties in the inbox.

What can be faked?

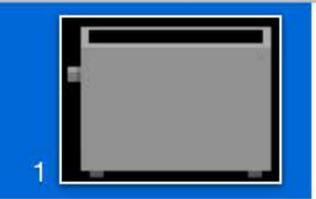
The connection between the app
and the toaster.
Anything related to toasting.

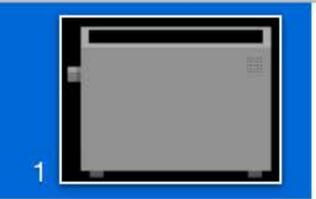
Sending data back and forth.
UI is still just pictures and
animations.

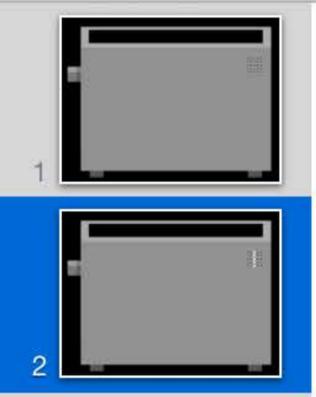
Where will it be used?

The kitchen.

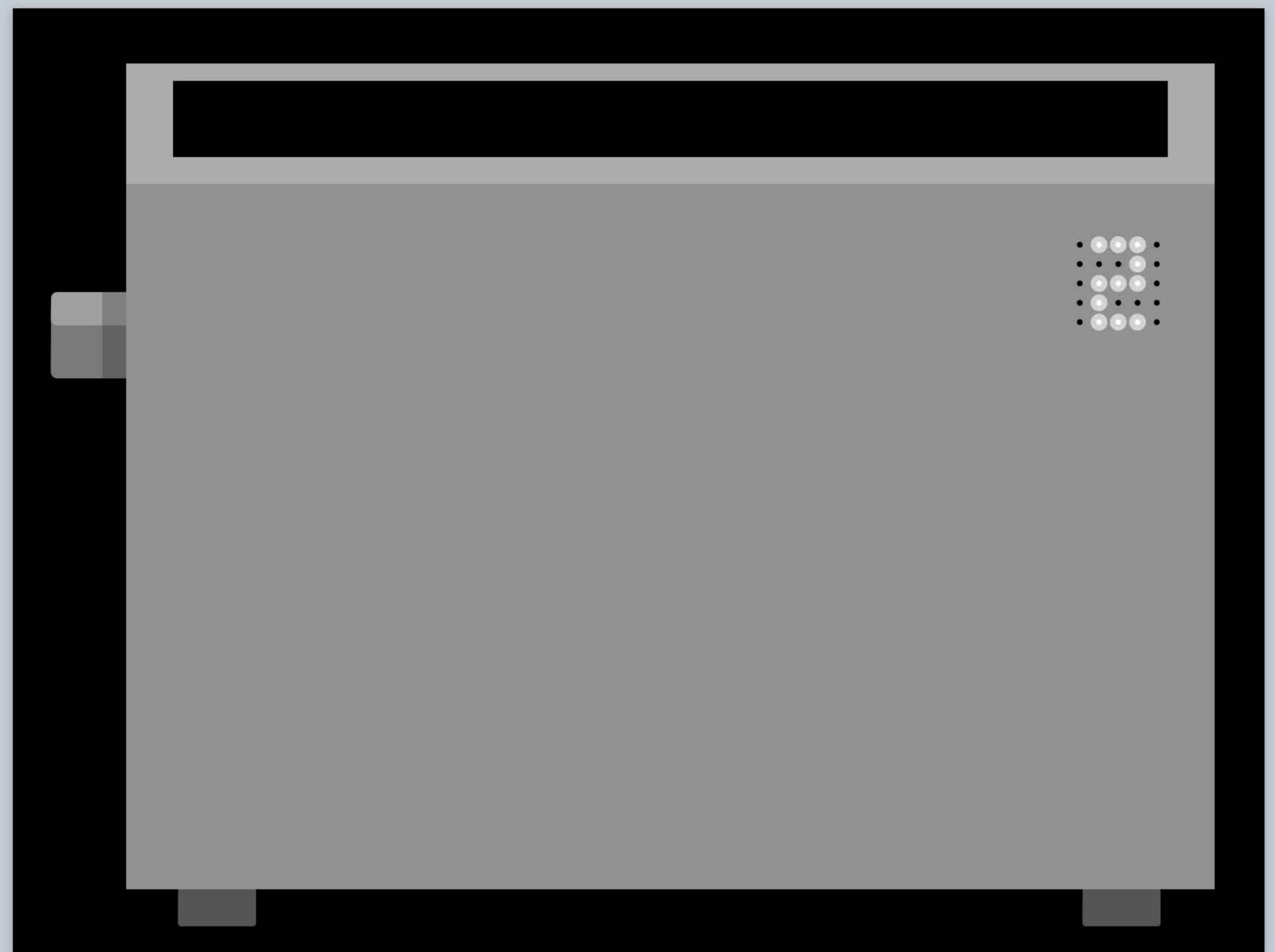
Anywhere.

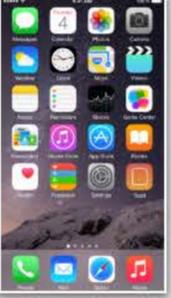
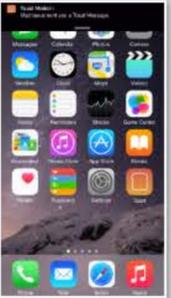
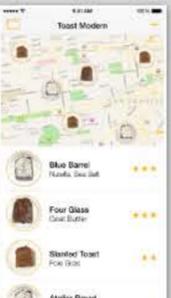
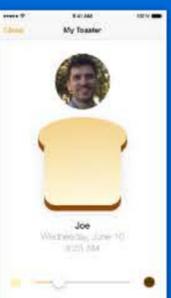






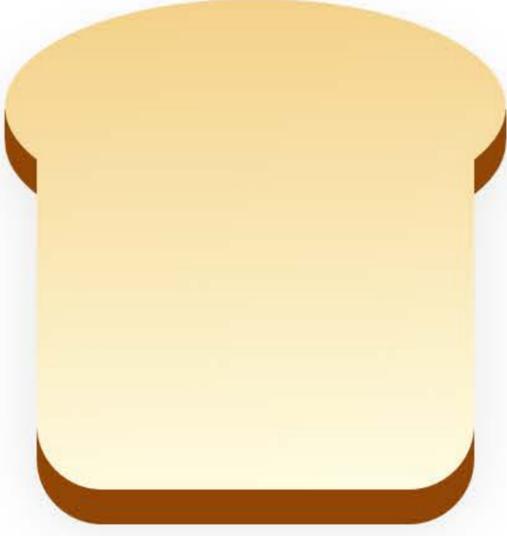
- 1
- 2
- 3



- 1 
- 2 
- 3 
- 4 

9:41 AM 100%

Close My Toaster



Joe
Wednesday, June 10
8:23 AM



- 1
- 2
- 3
- 4
- 5

The main interface displays three toast notifications, each represented by a toast icon with a number and a person's profile picture. The notifications are:

- Marek**: Tuesday, June 9, 6:11 AM. The toast icon contains the number 2.
- Rachel**: Tuesday, June 9, 7:06 AM. The toast icon contains the number 3.
- Joe**: Wednesday, June 10, 8:23 AM. The toast icon contains the number 4.

The background is a light blue gradient. At the bottom, there is a slider control with a yellow dot on the left, a white circle in the middle, and a dark brown dot on the right.

The inset view shows the 'My Toaster' app interface. At the top, it says 'Close' and 'My Toaster'. Below this is a circular profile picture of a man. Underneath the picture is a toast icon with the number 4. At the bottom of the inset, there is a slider control with a yellow dot on the left, a white circle in the middle, and a dark brown dot on the right.

- 1
- 2
- 3
- 4
- 5
- 6

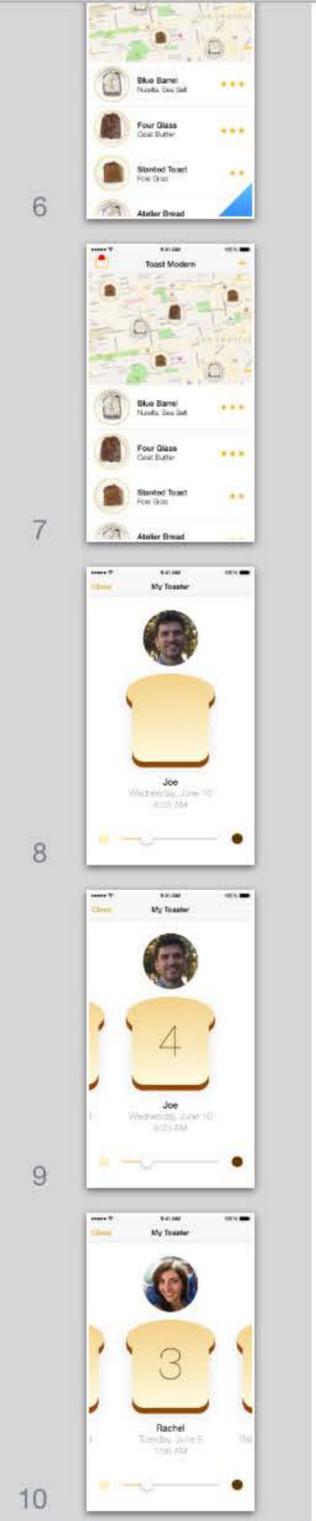
9:41 AM 100%

Close My Toaster

Marek
Tuesday, June 9
6:11 AM

Rachel
Tuesday, June 9
7:06 AM

Joe
Wednesday, June 10
8:23 AM



9:41 AM 100%

Close My Toaster

🕒 2:45



Rachel
Tuesday, June 9
7:06 AM







Marek
Tuesday, June 9
6:11 AM





Joe
Wednesday, June 10
8:23 AM

- 3
- 4
- 5
- 6
- 7
- 8

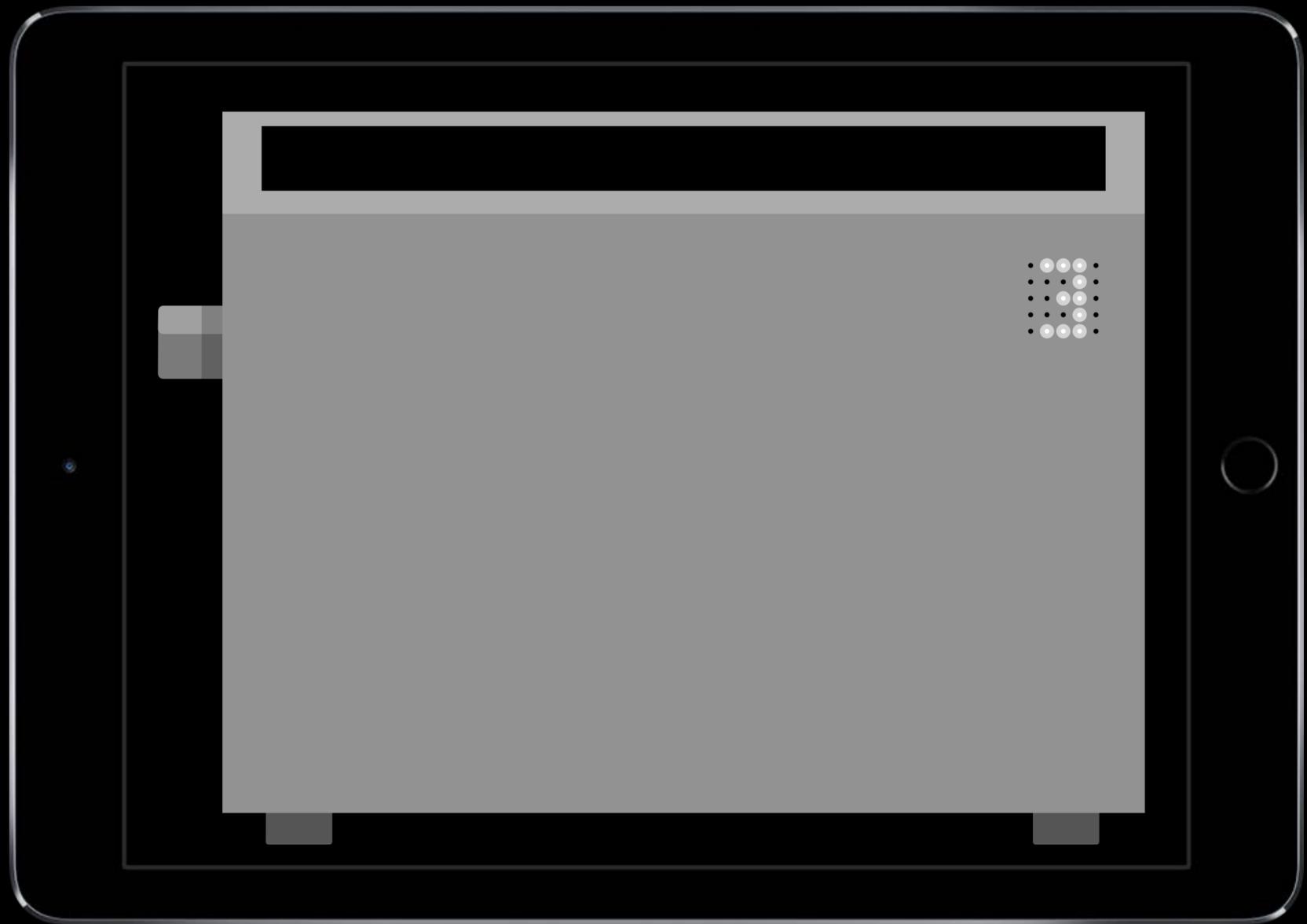
9:41 AM 100%

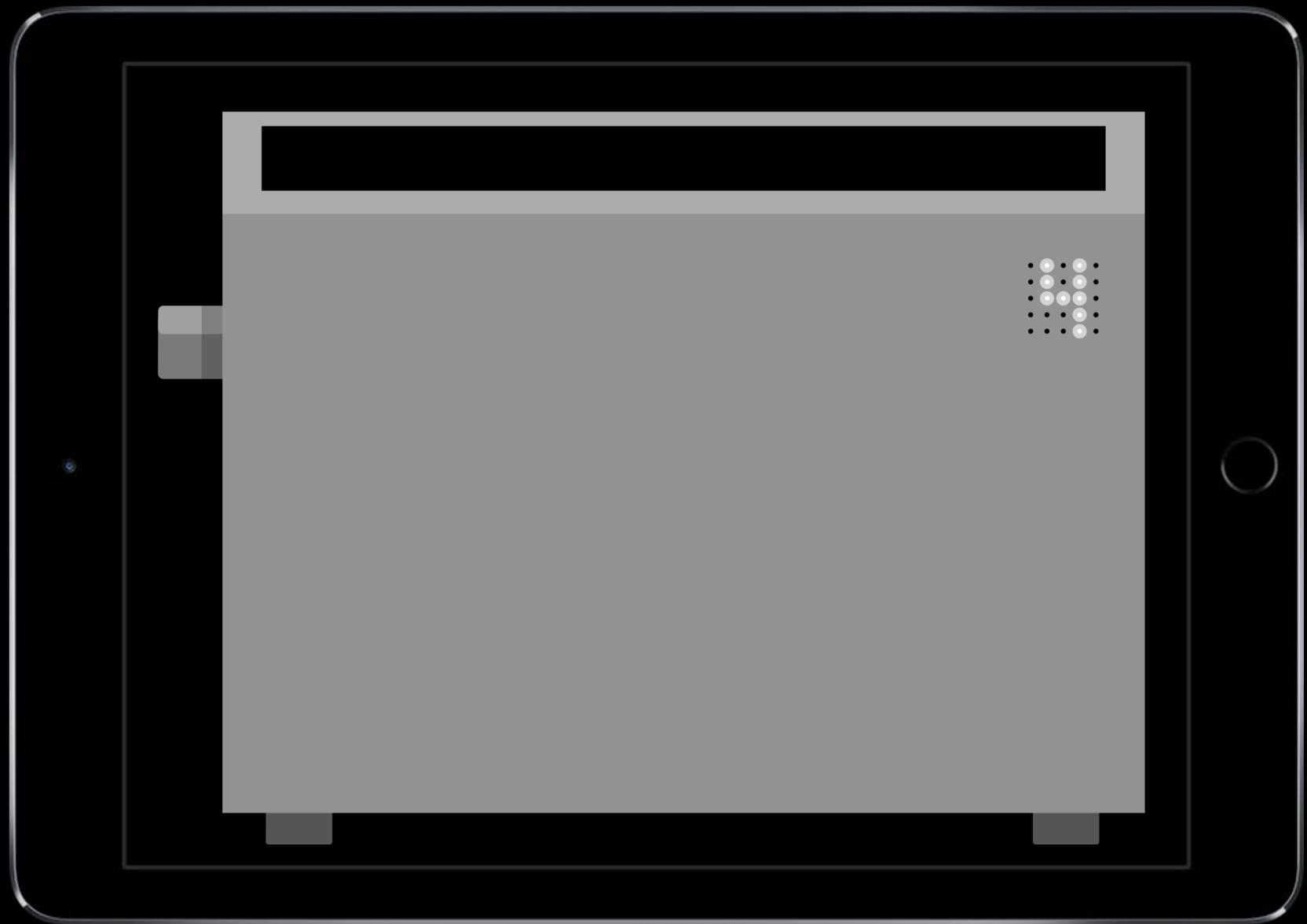
Close My Toaster

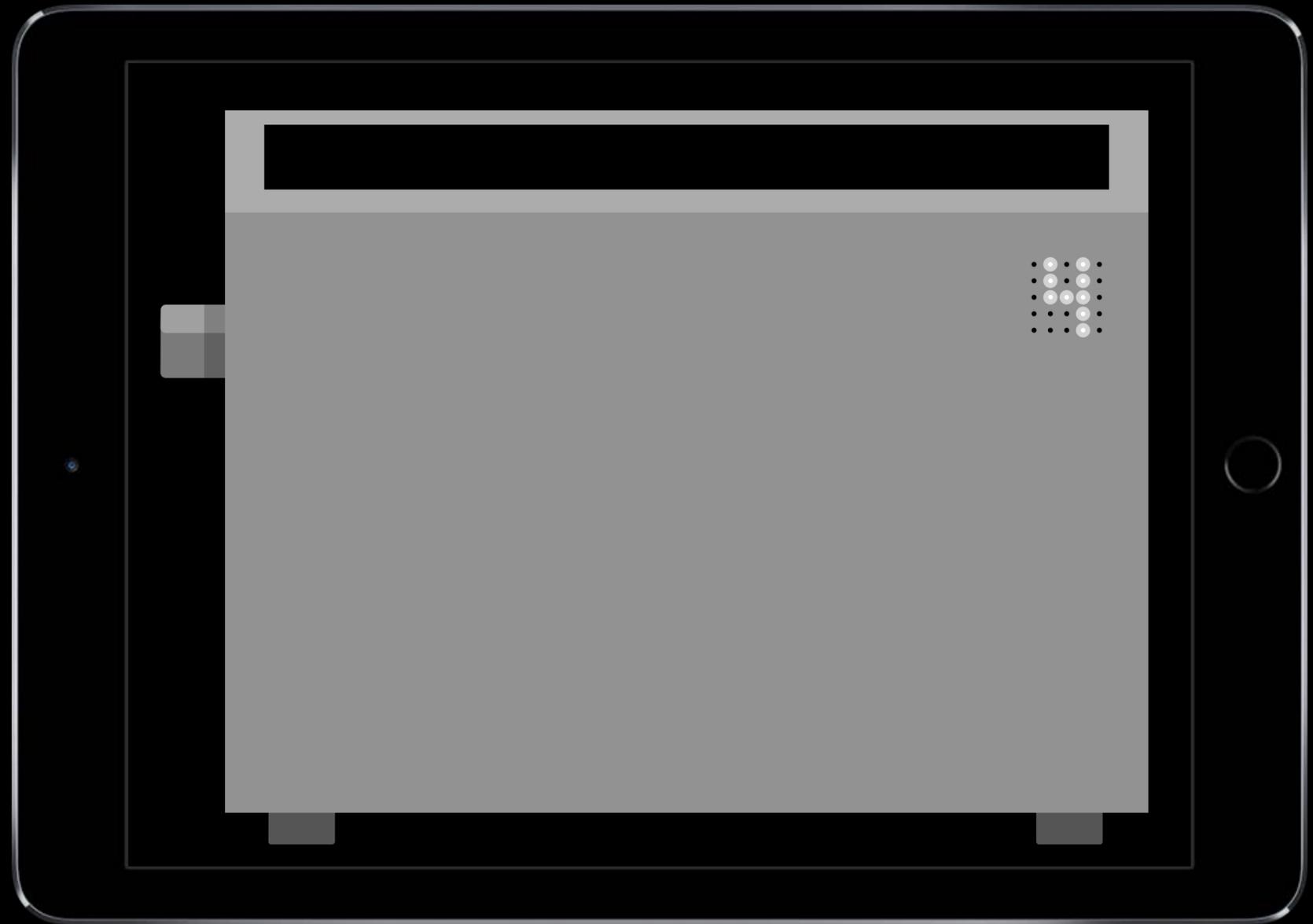
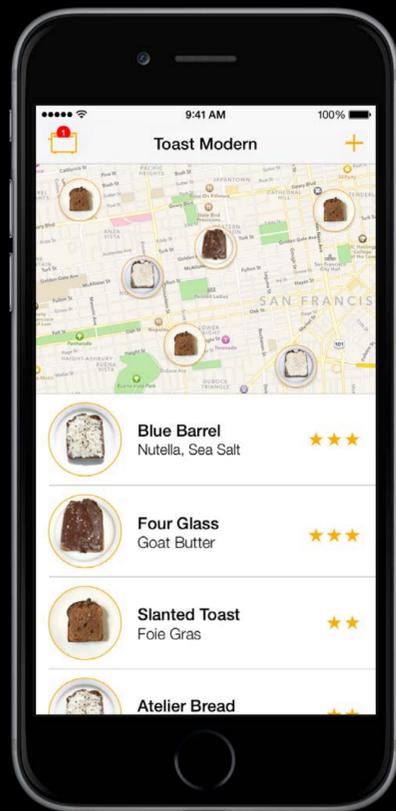
Rachel
Tuesday, June 9
7:06 AM

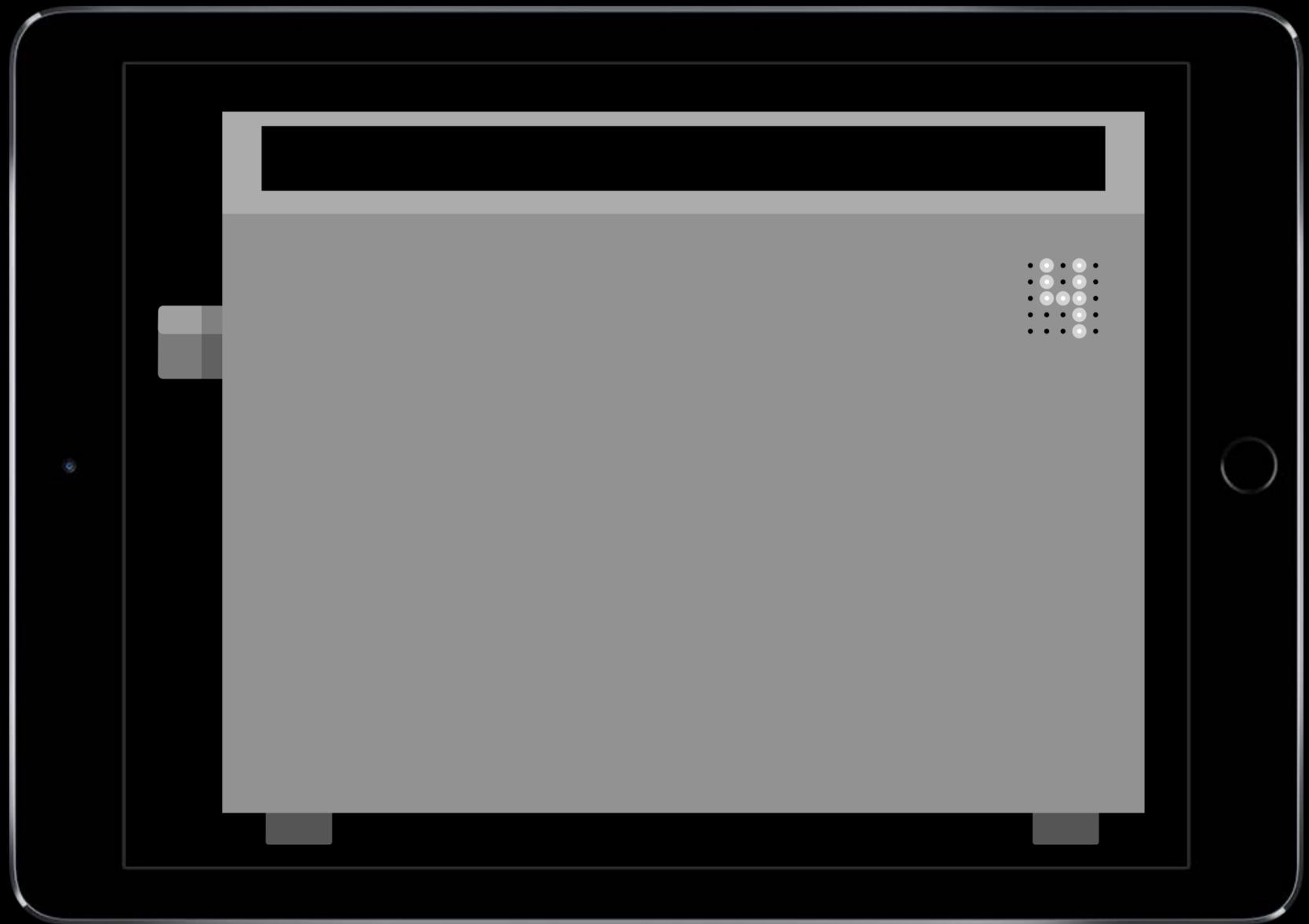
Marek
Tuesday, June 9
6:11 AM

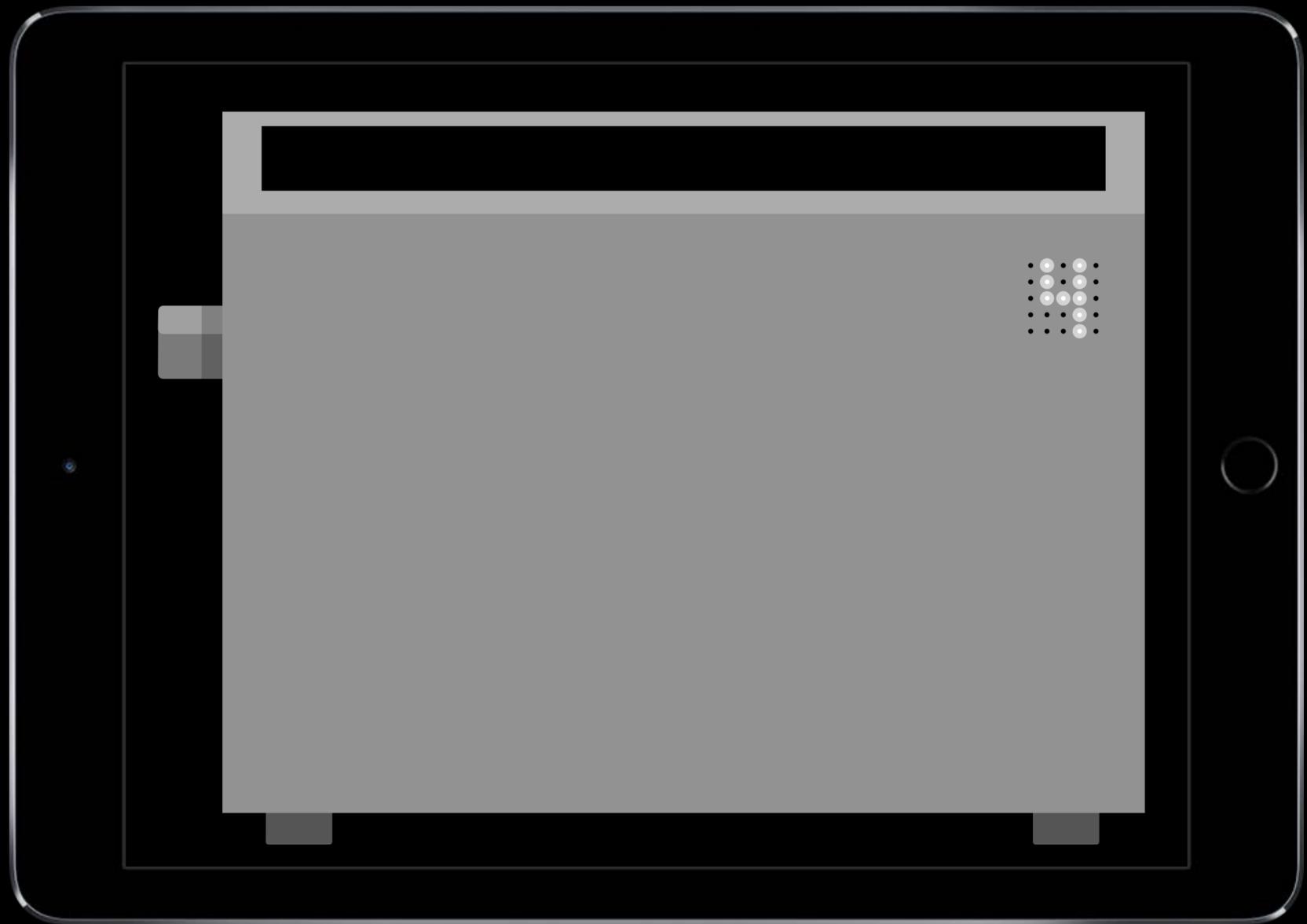
Joe
Wednesday, June 10
8:23 AM

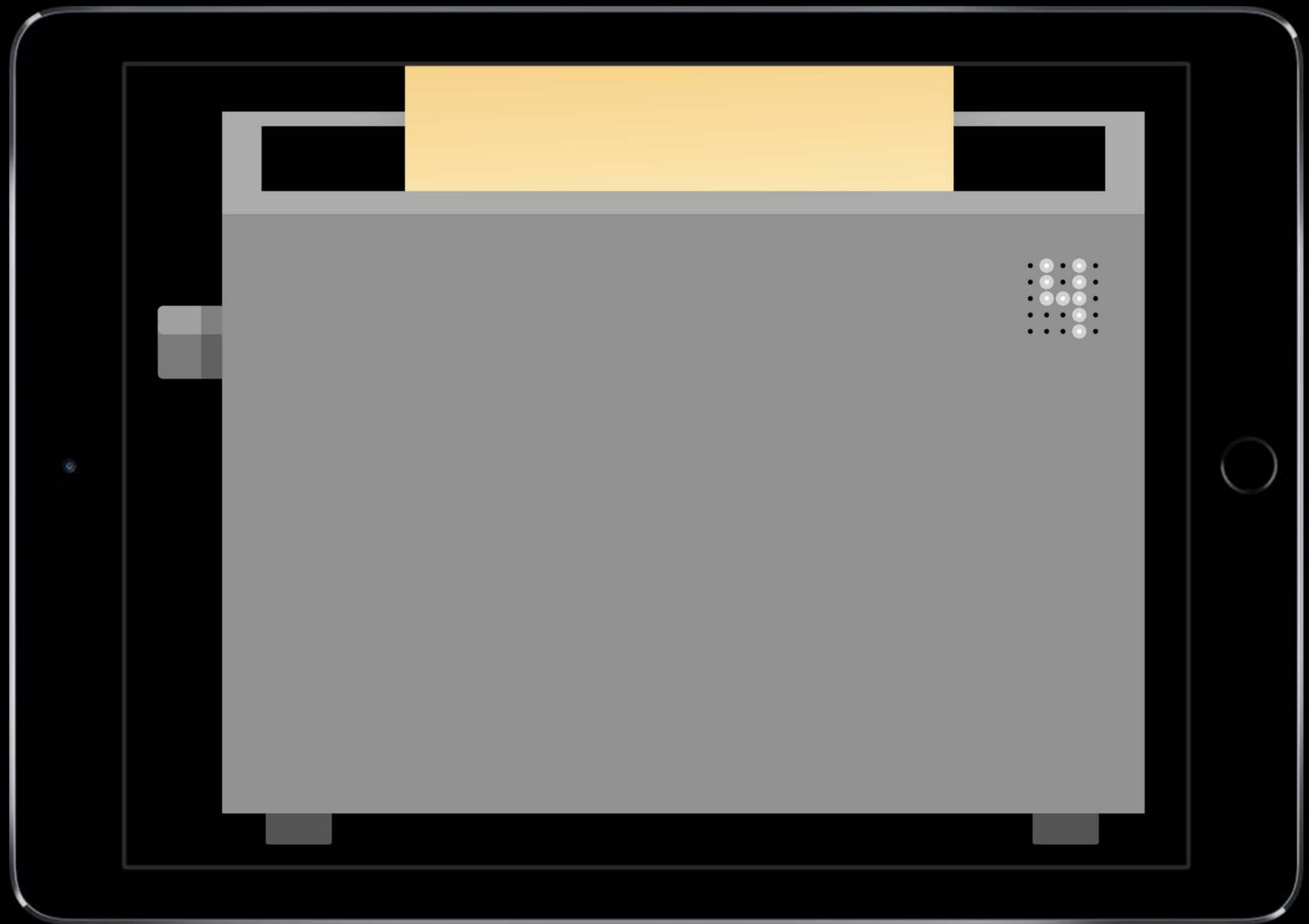


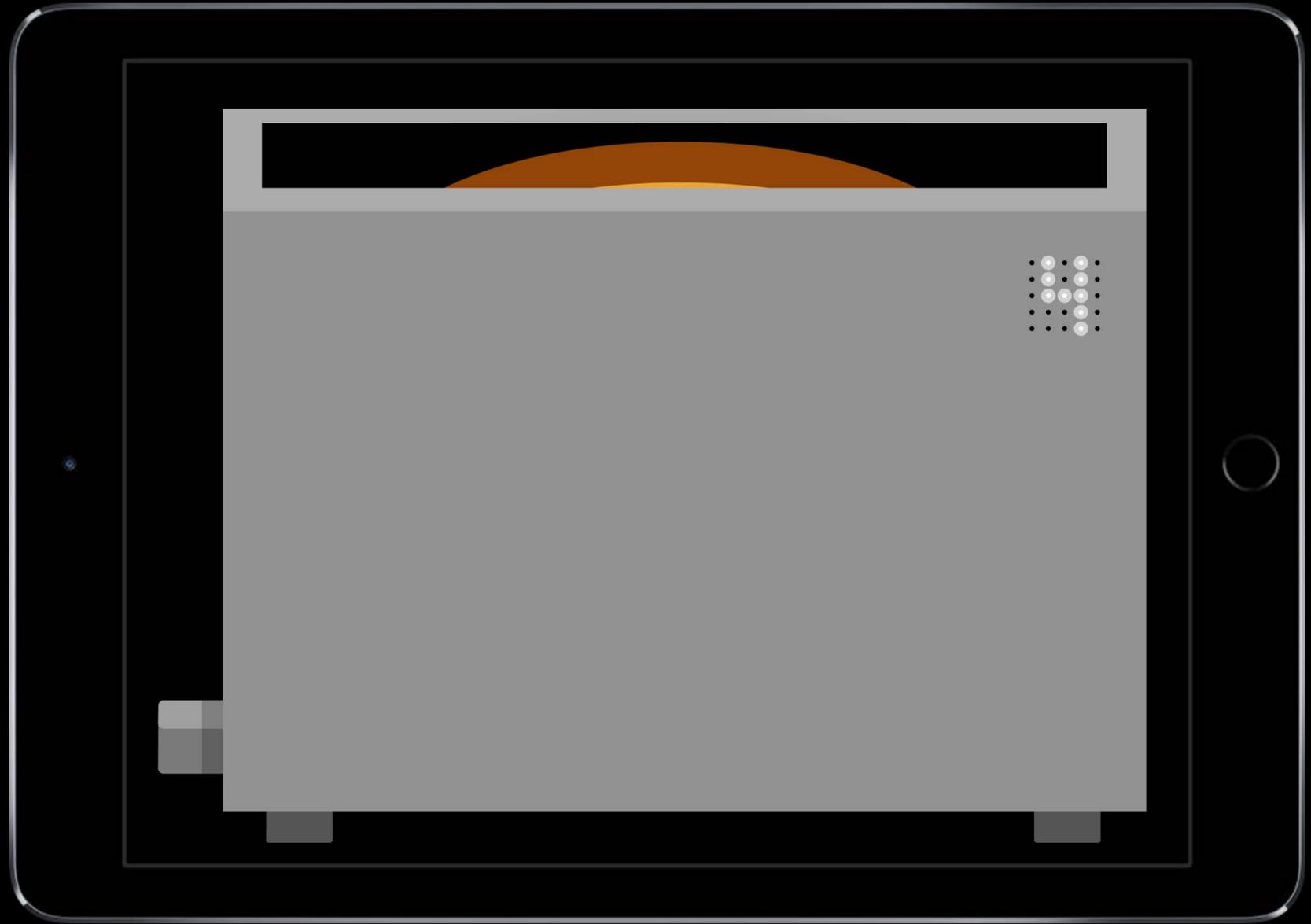


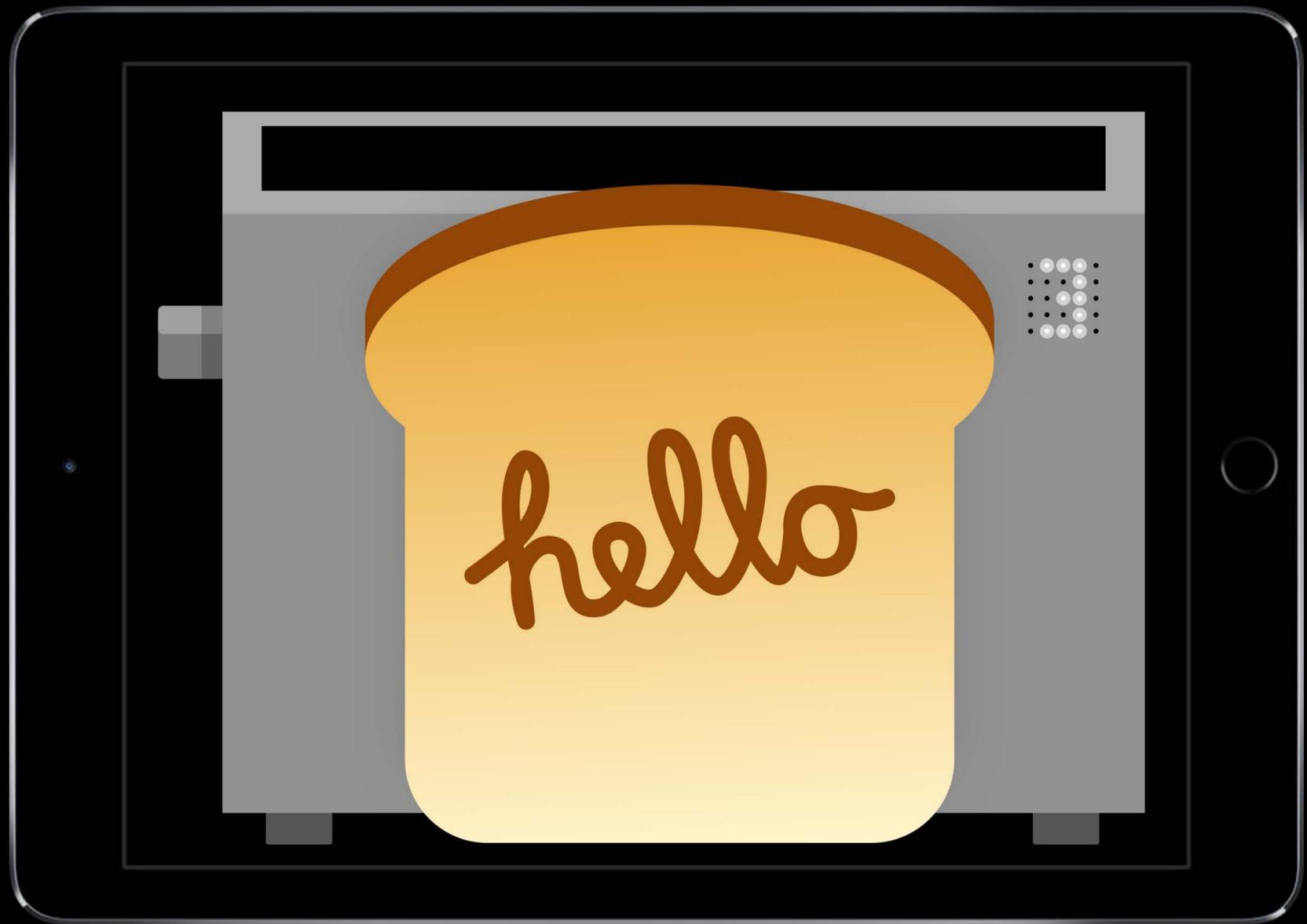




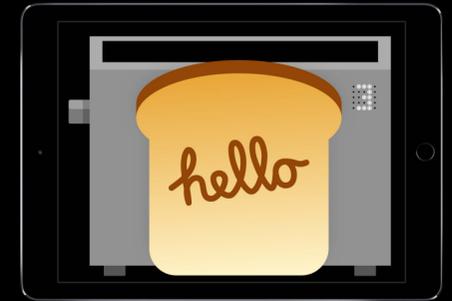






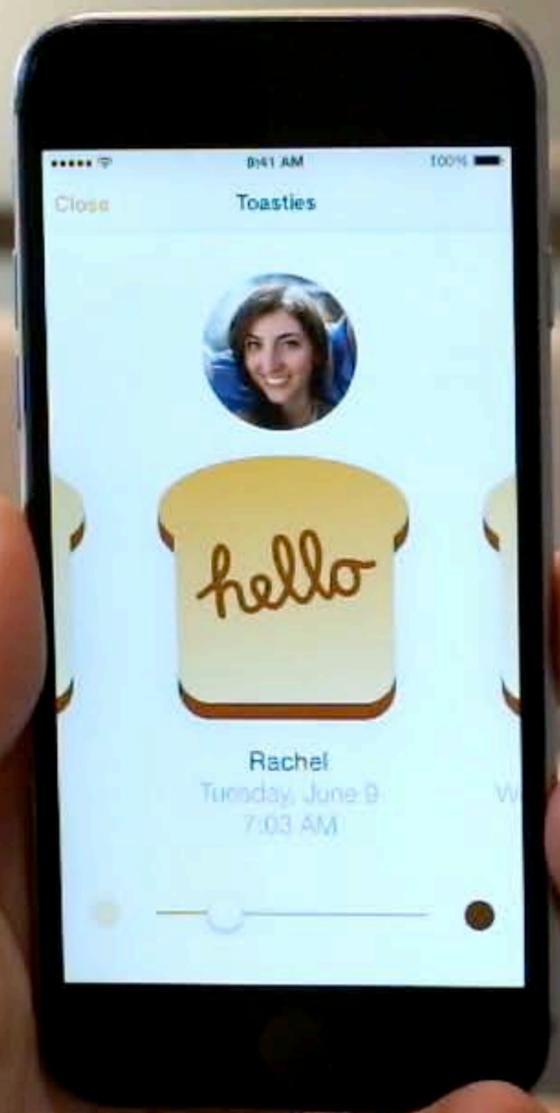
















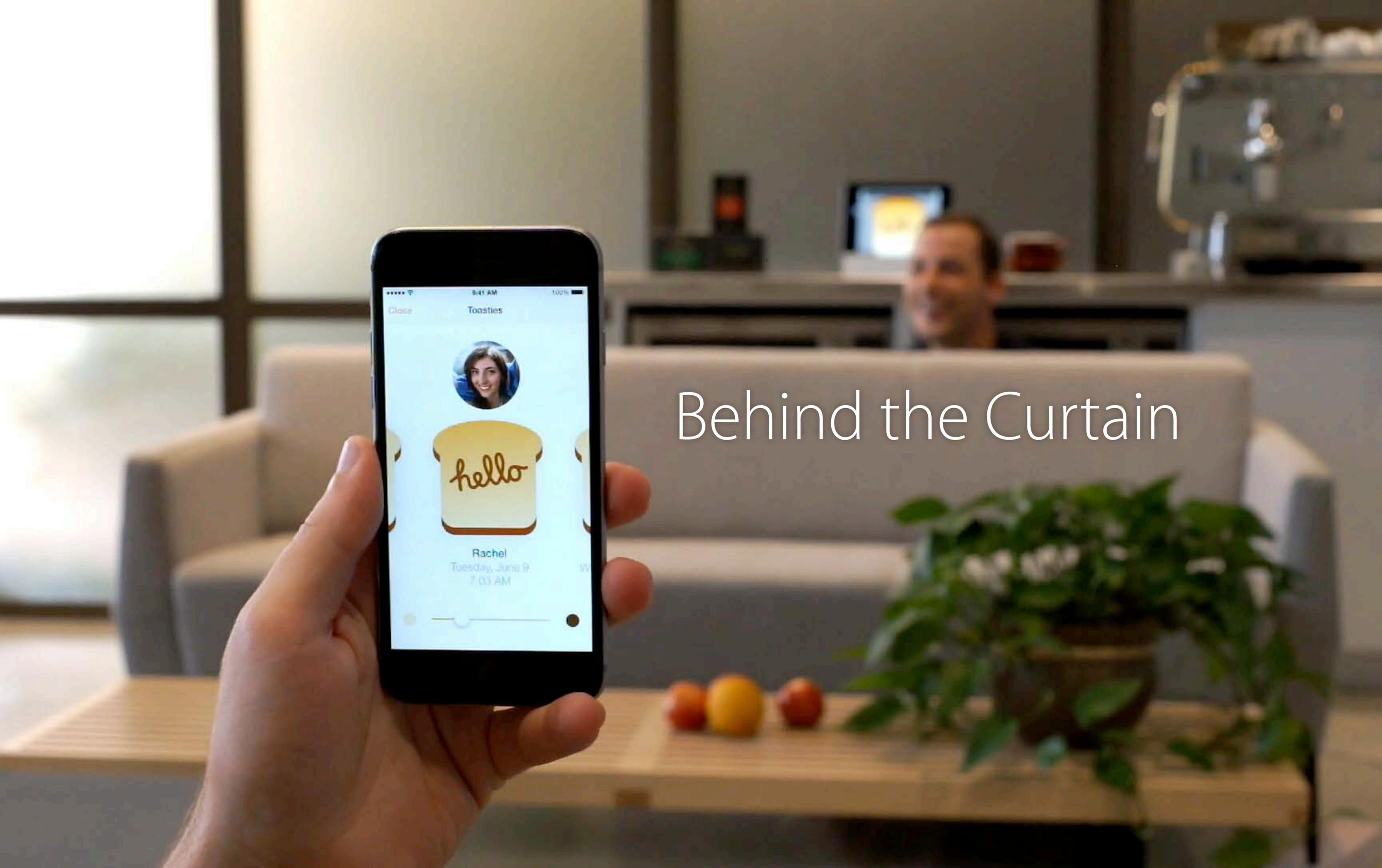
Behind the Curtain



Behind the Curtain



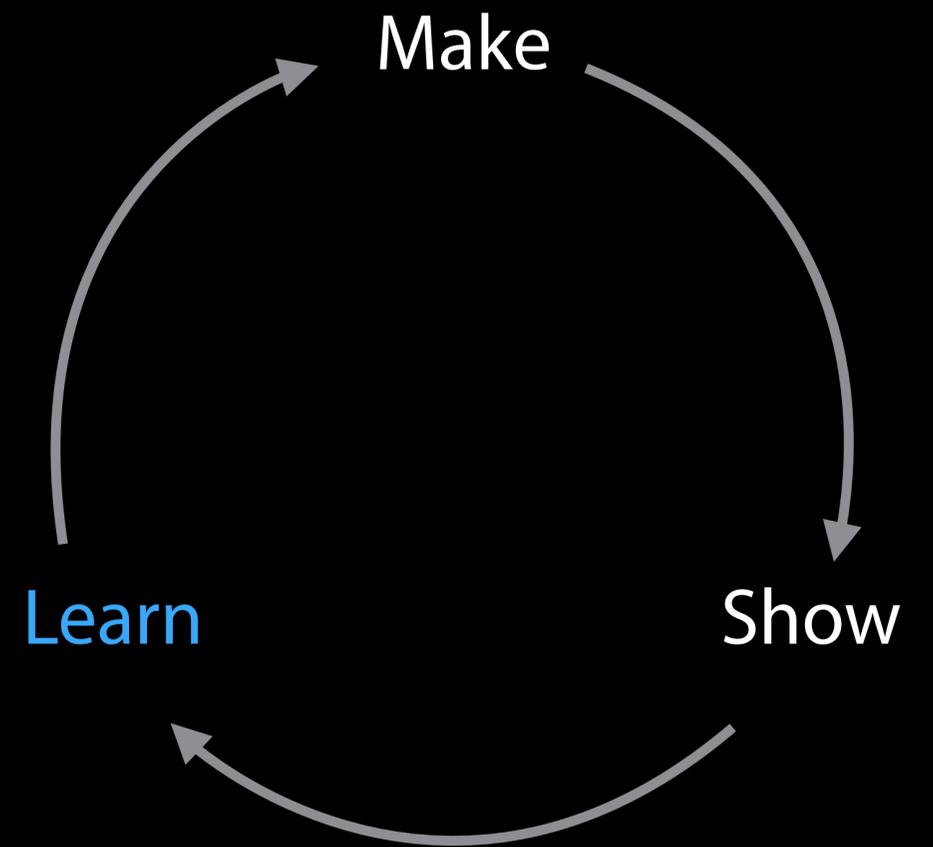
Behind the Curtain





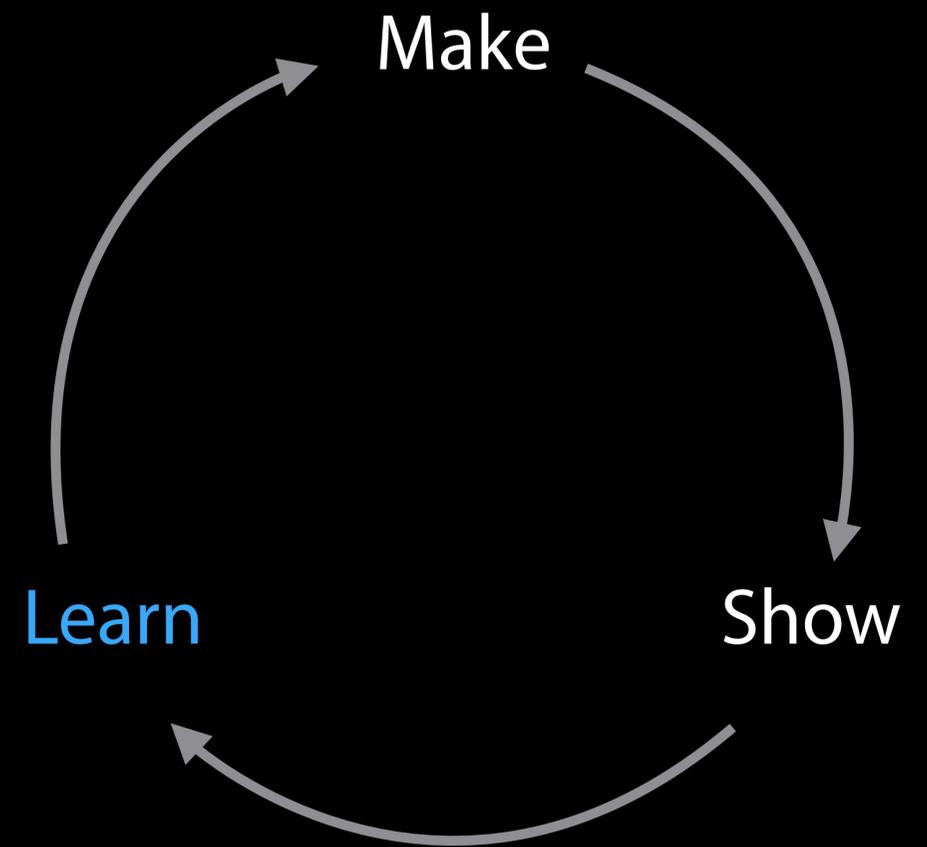
Behind the Curtain

Learn from Feedback



Learn from Feedback

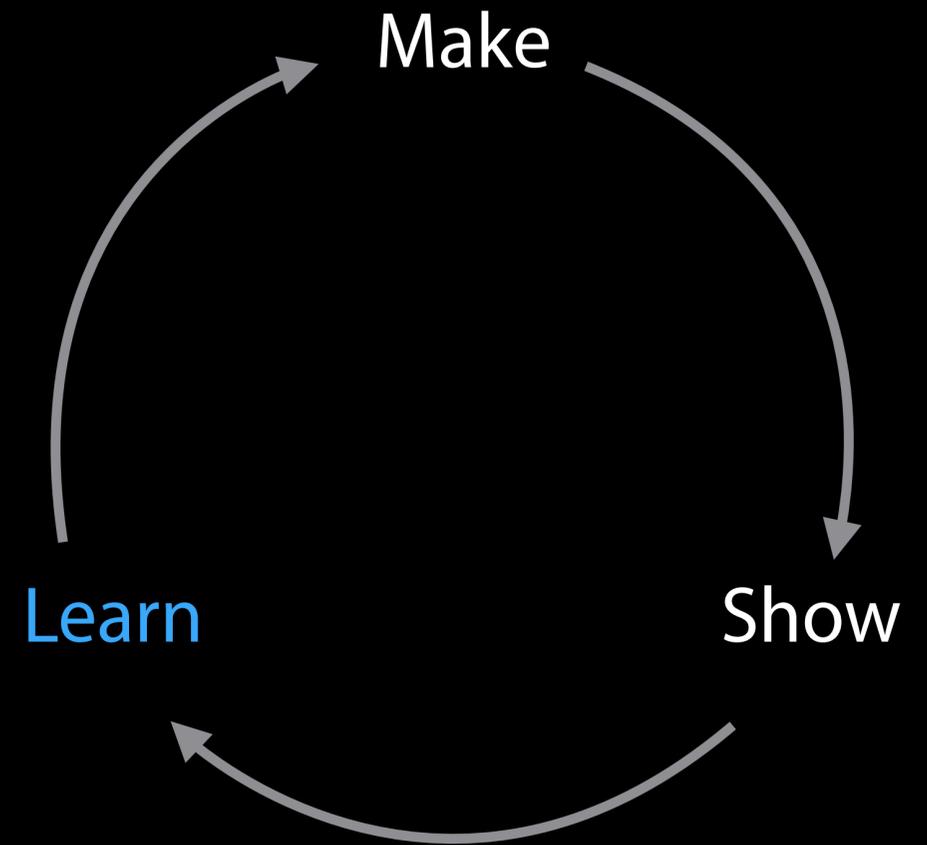
What's working?



Learn from Feedback

What's working?

Selecting from many Toasties and toasting the right one

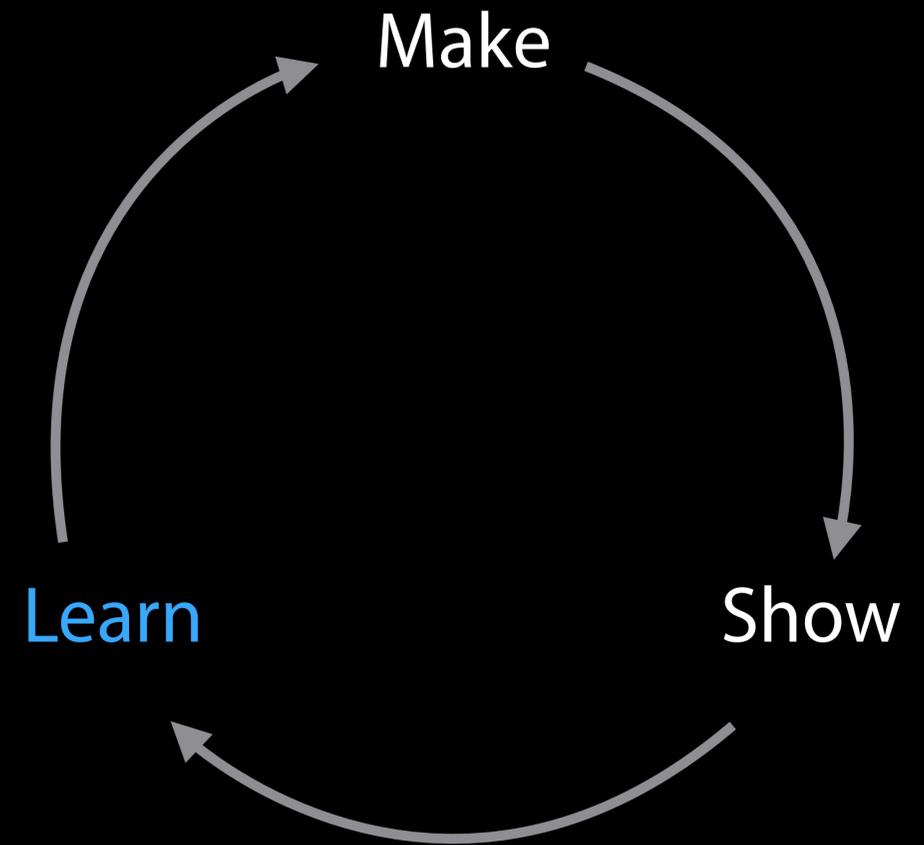


Learn from Feedback

What's working?

Selecting from many Toasties and toasting the right one

What's not working?



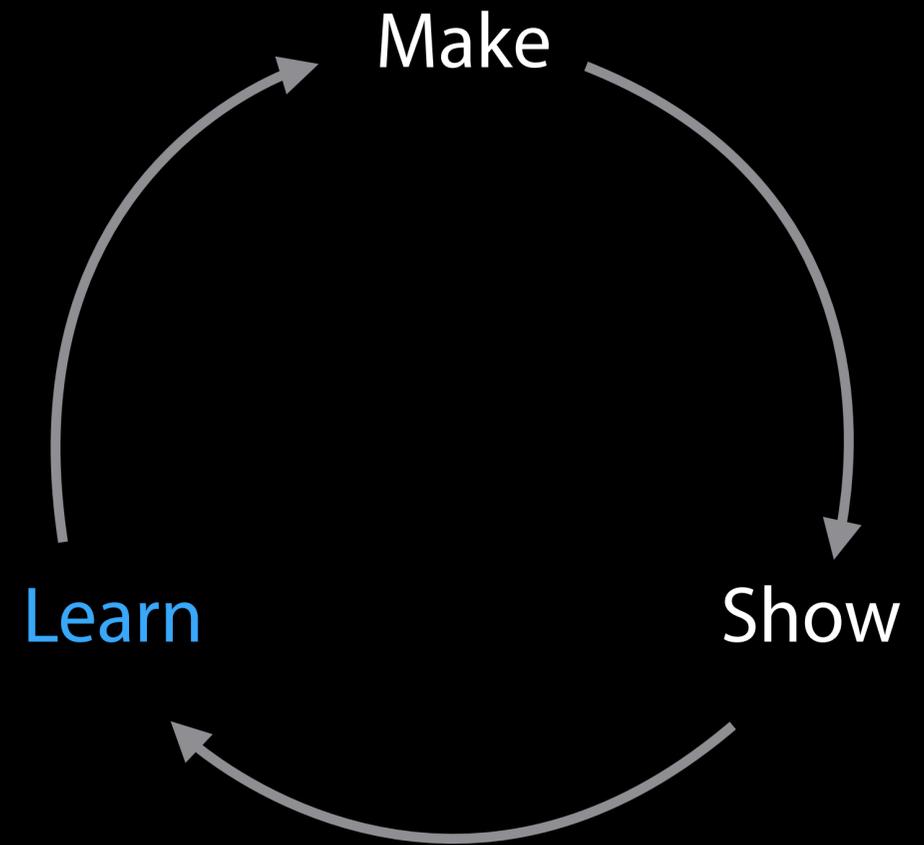
Learn from Feedback

What's working?

Selecting from many Toasties and toasting the right one

What's not working?

The connection between the devices doesn't feel responsive



Learn from Feedback

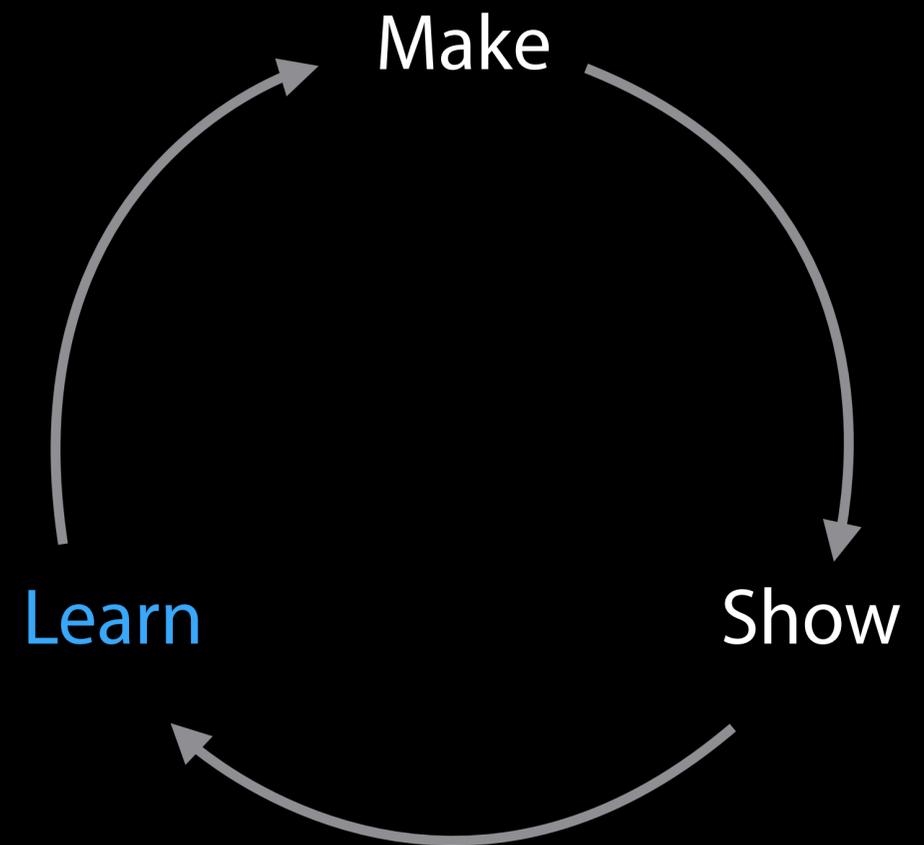
What's working?

Selecting from many Toasties and toasting the right one

What's not working?

The connection between the devices doesn't feel responsive

What ideas does this give us?



Learn from Feedback

What's working?

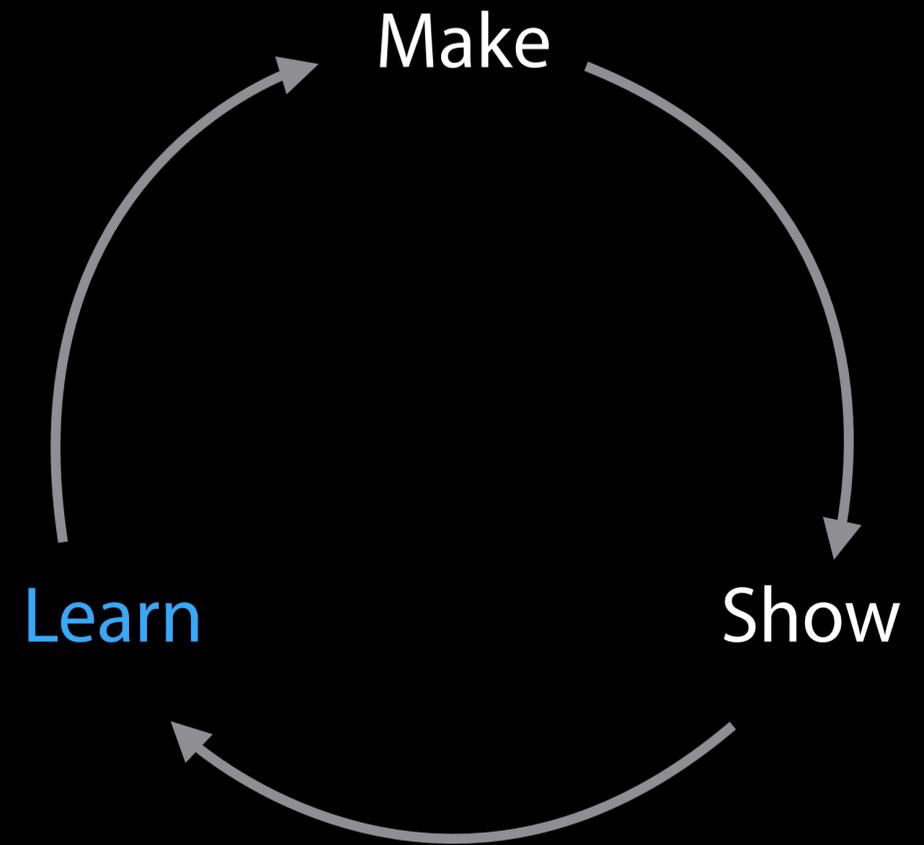
Selecting from many Toasties and toasting the right one

What's not working?

The connection between the devices doesn't feel responsive

What ideas does this give us?

The toaster should react in real-time to what happens in the app. And vice-versa



Make Fake Hardware and Software

Make Fake Hardware and Software

Fake hardware

Fake app

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Wire up the Digital Lever.
Connect the state of the toaster to
the iPhone app.

Browse Toasties inbox.
Pick Toasties to toast.

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Wire up the Digital Lever.
Connect the state of the toaster to
the iPhone app.

Browse Toasties inbox.
Pick Toasties to toast.

What can be faked?

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Wire up the Digital Lever.
Connect the state of the toaster to
the iPhone app.

Browse Toasties inbox.
Pick Toasties to toast.

What can be faked?

Servers.
Edge cases.
Turning bread into toast.

Message queue.
Sending the image data to the
toaster.

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Wire up the Digital Lever.
Connect the state of the toaster to
the iPhone app.

Browse Toasties inbox.
Pick Toasties to toast.

What can be faked?

Servers.
Edge cases.
Turning bread into toast.

Message queue.
Sending the image data to the
toaster.

Where will it be used?

Make Fake Hardware and Software

Fake hardware

Fake app

What needs to be more real?

Wire up the Digital Lever.
Connect the state of the toaster to
the iPhone app.

Browse Toasties inbox.
Pick Toasties to toast.

What can be faked?

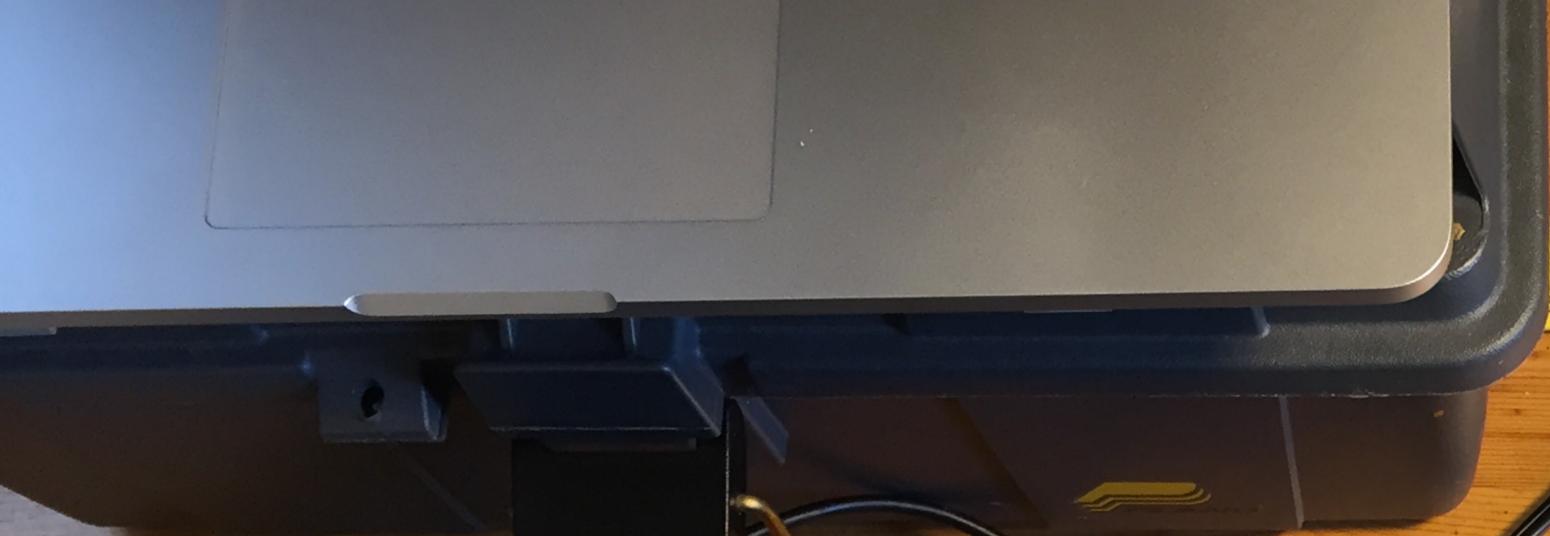
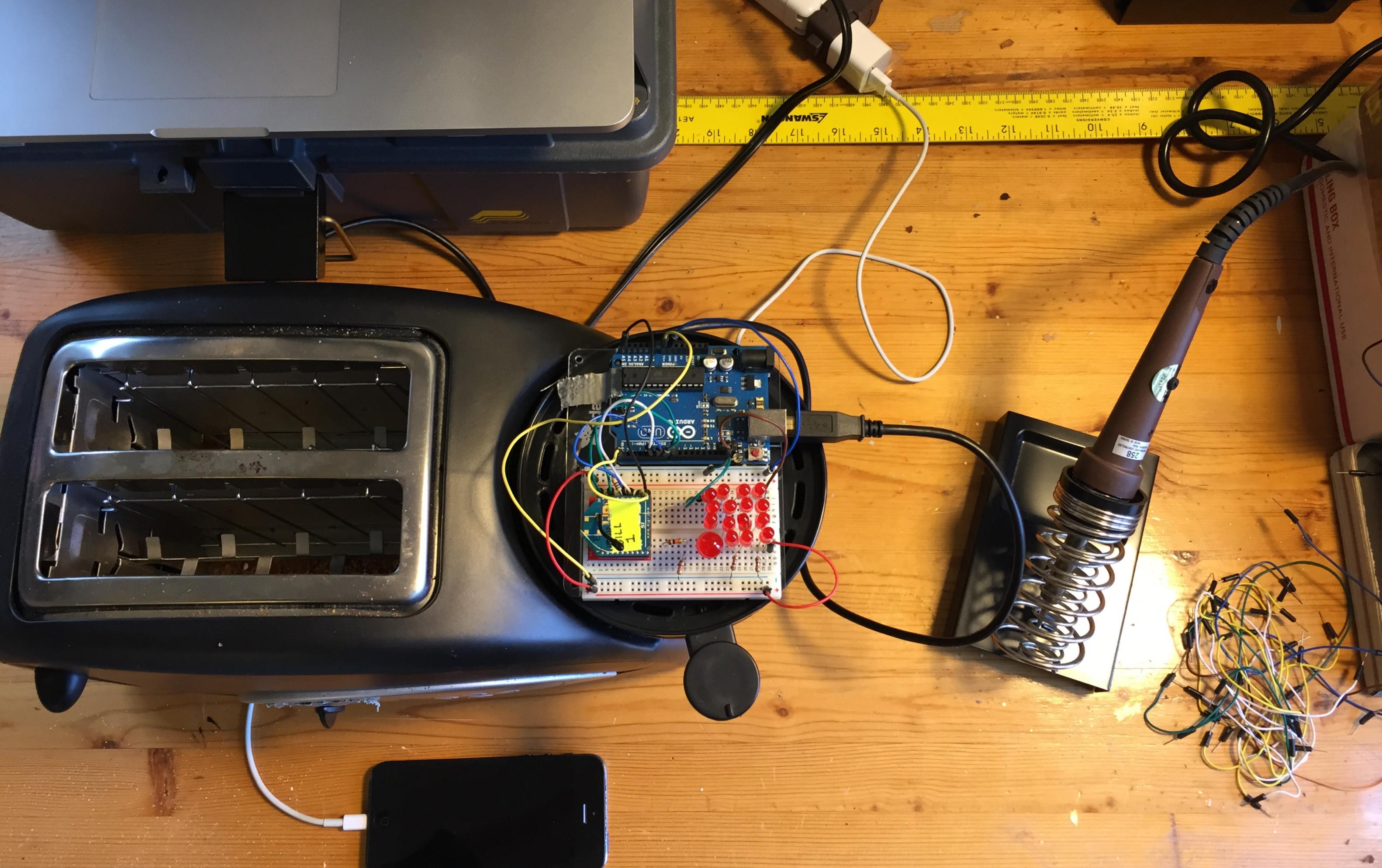
Servers.
Edge cases.
Turning bread into toast.

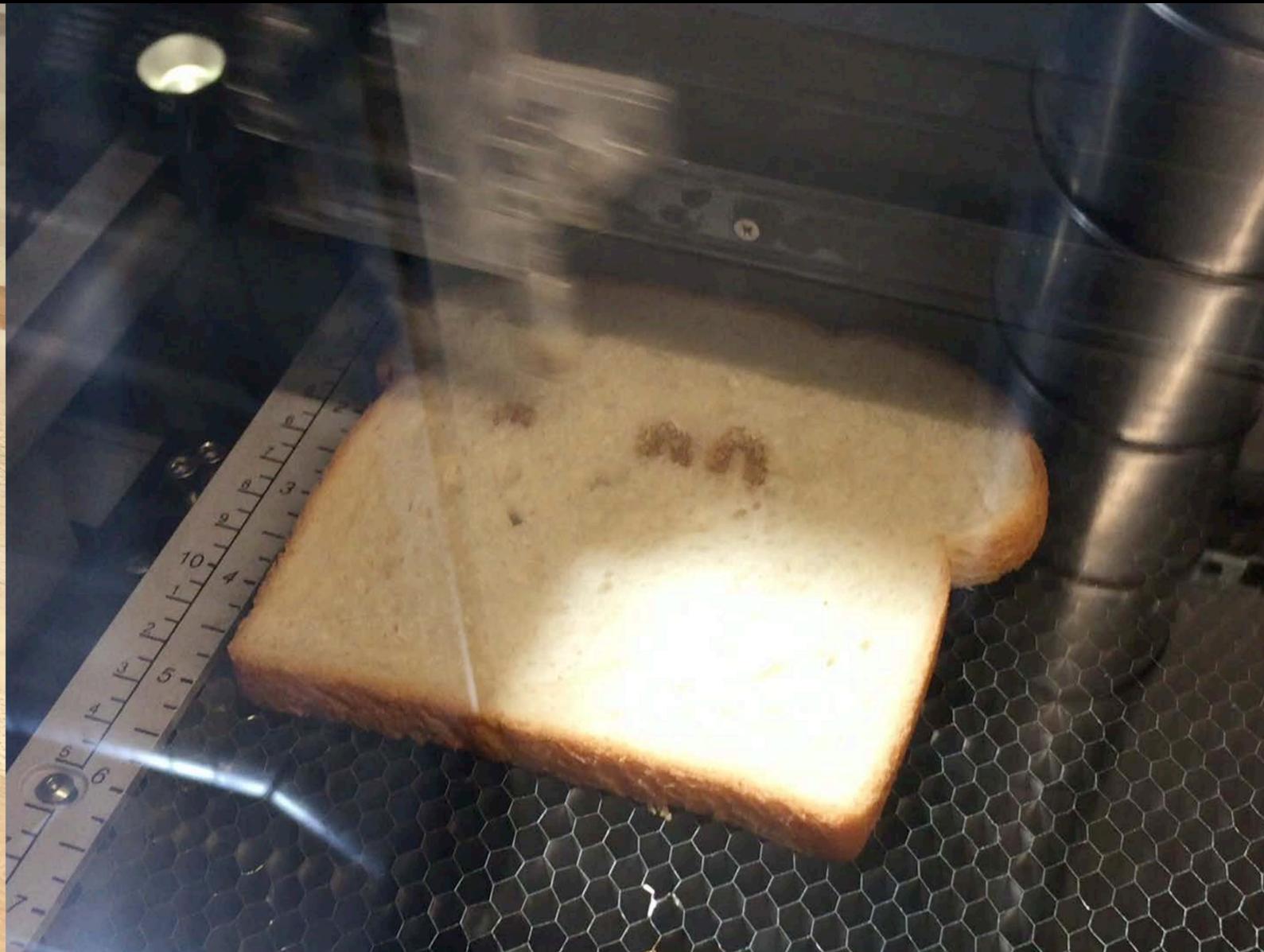
Message queue.
Sending the image data to the
toaster.

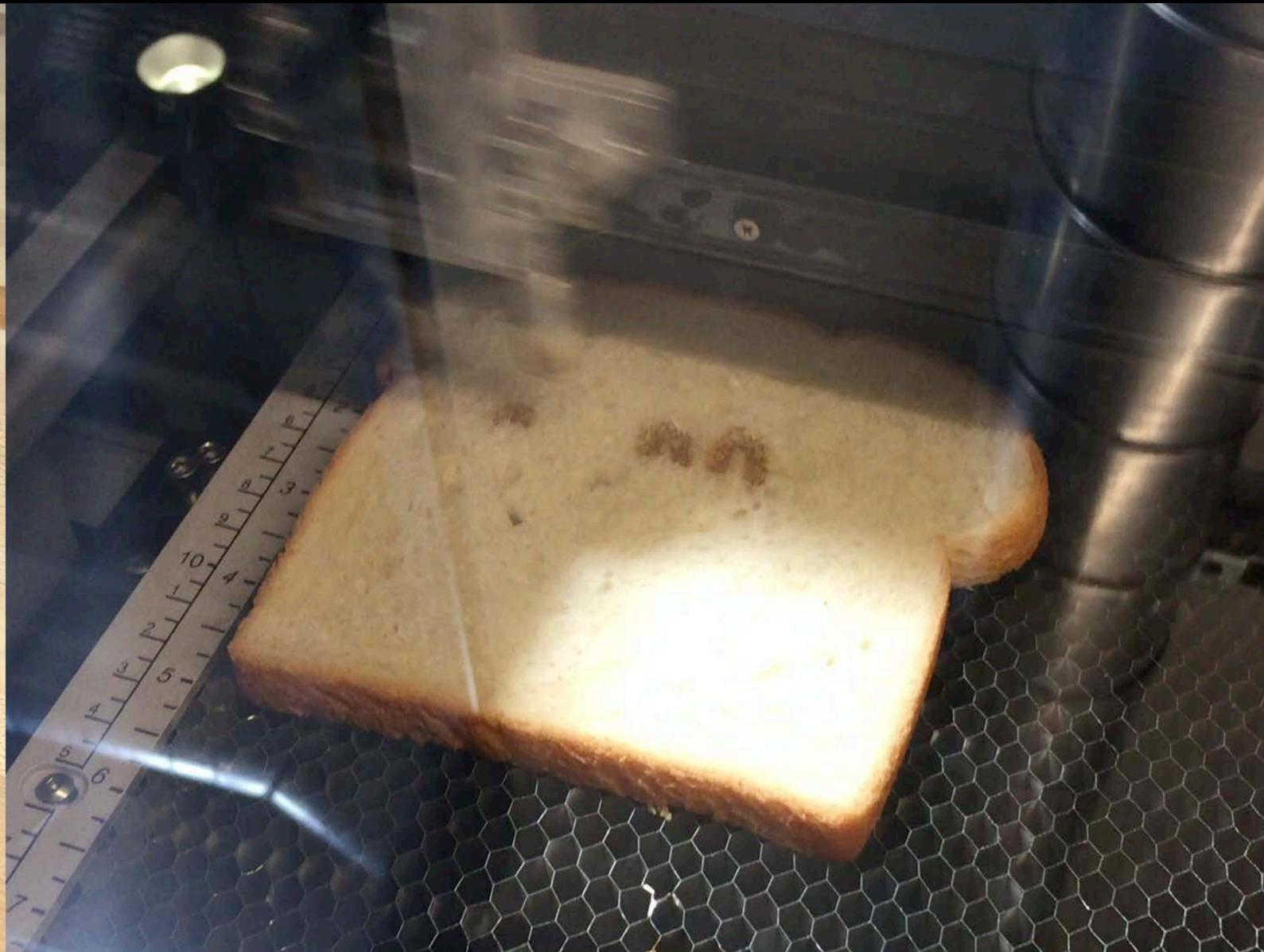
Where will it be used?

The kitchen.

Anywhere.











Keep It Lean

Interactive and Connected

What is a toaster?

Interactive and Connected

What is a toaster?

A connected state machine

Interactive and Connected

What is a toaster?

A connected state machine

- Physical state

Interactive and Connected

What is a toaster?

A connected state machine

- Physical state
- How it changes with user input

Interactive and Connected

Add some logic. But not too much.



Interactive and Connected

Toast a Toastie flow

Interactive and Connected

Toast a Toastie flow

1

Pick a Toastie
from the inbox

Interactive and Connected

Toast a Toastie flow

1

Pick a Toastie
from the inbox

2

Send the Toastie
to the toaster

Interactive and Connected

Toast a Toastie flow

1

Pick a Toastie
from the inbox

2

Send the Toastie
to the toaster

3

Toast the
selected Toastie

Interactive and Connected

Toast a Toastie flow

1

Pick a Toastie
from the inbox

2

Send the Toastie
to the toaster

3

Toast the
selected Toastie

4

Indicate that the
Toastie is toasted

How?

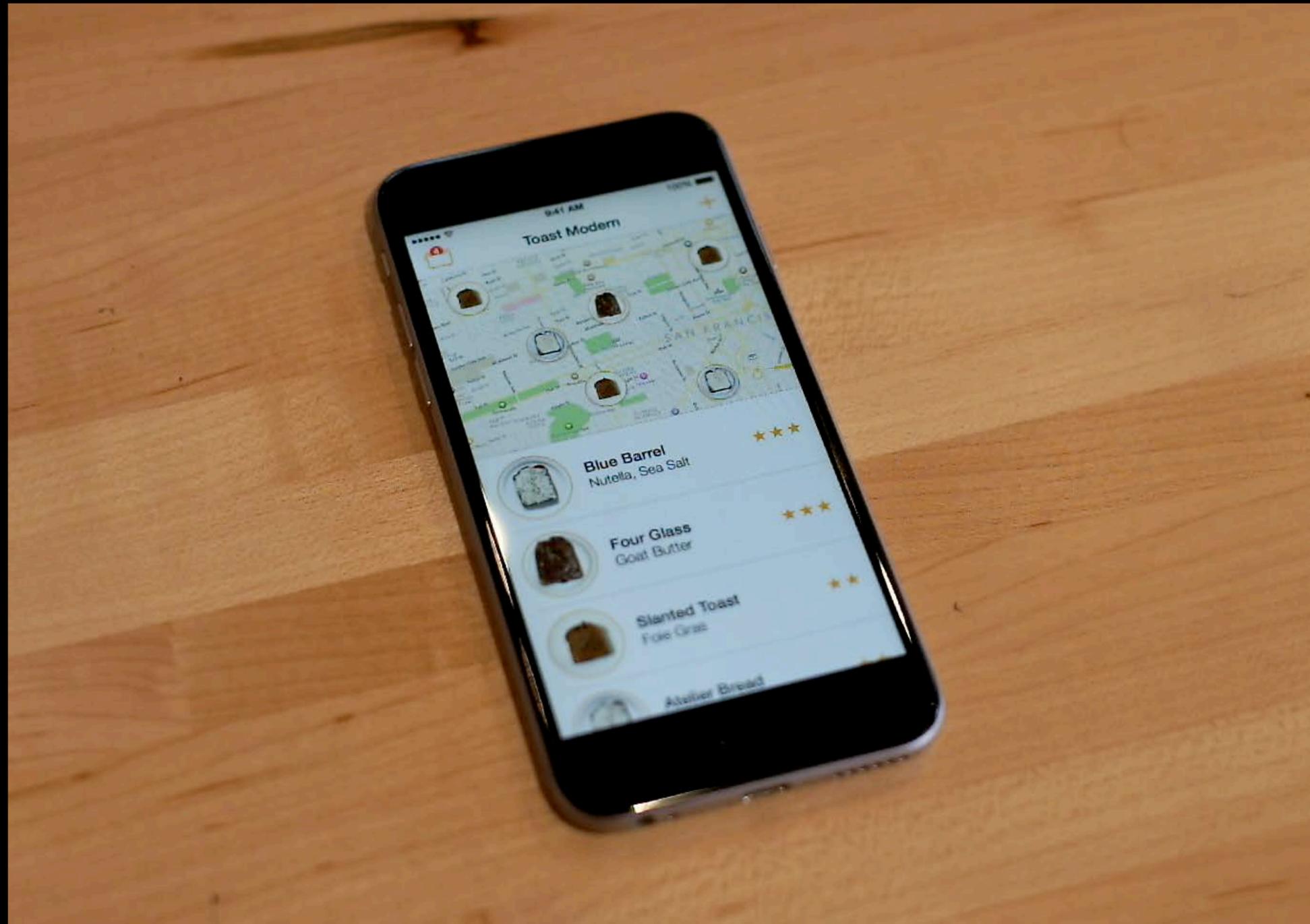
Pick a Toastie from the inbox



Make It

Pick a Toastie from the inbox

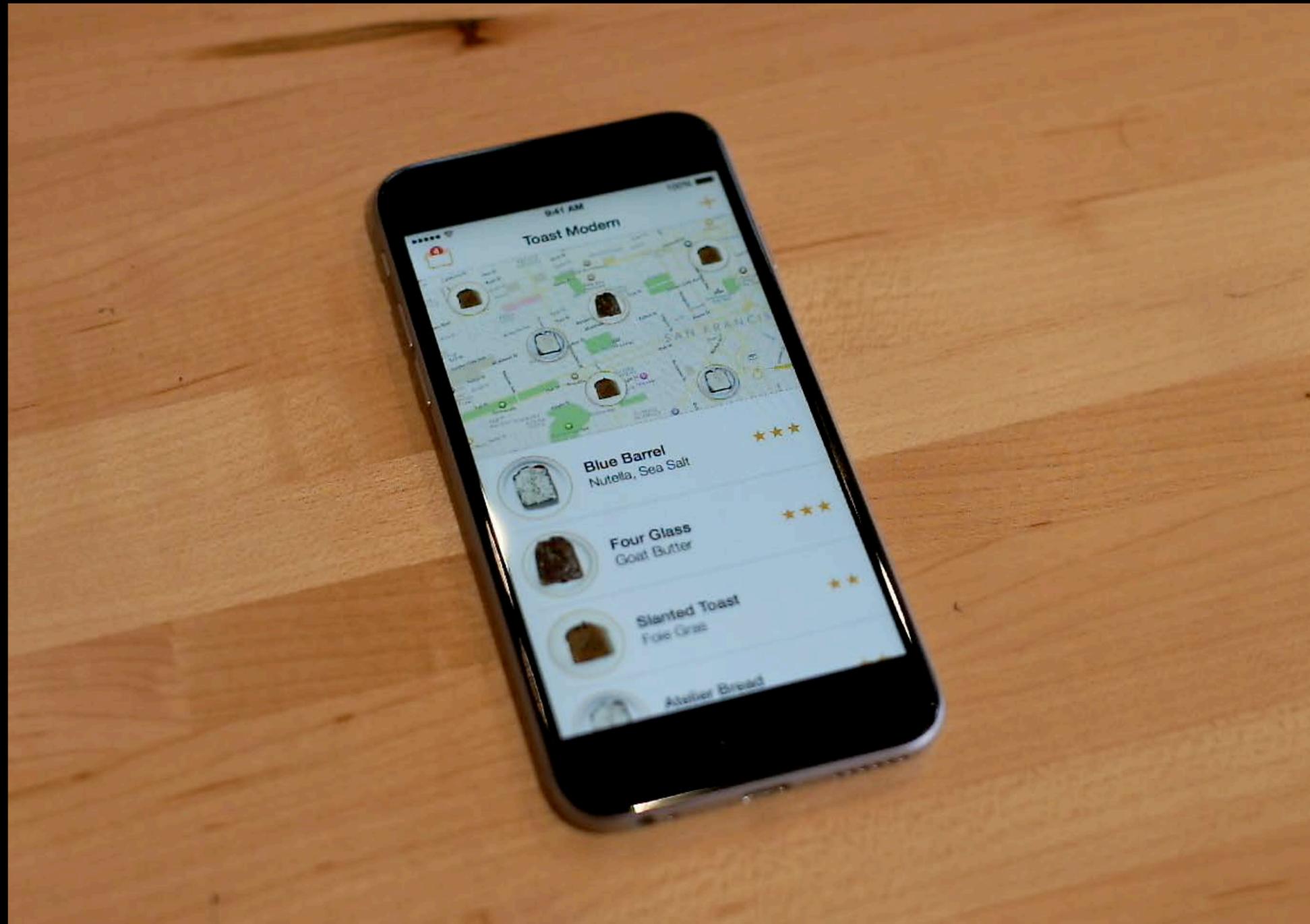
```
toastie_selected      1
toast_darkness        0.5
is_toasting           false
toast_time_remaining  0
```



Make It

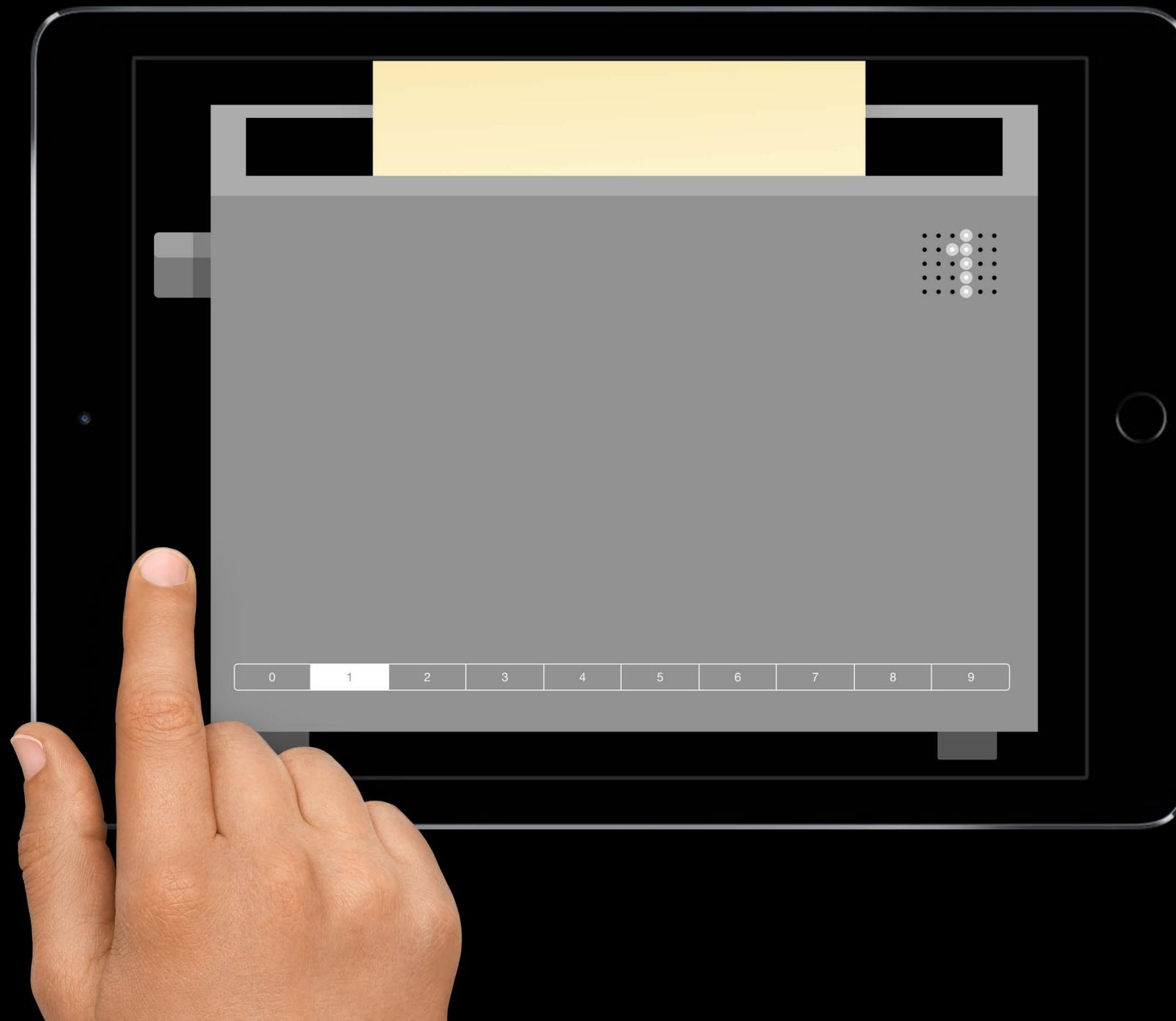
Pick a Toastie from the inbox

```
toastie_selected      1
toast_darkness       0.5
is_toasting          false
toast_time_remaining  0
```



How?

Toast a Toastie



Make It

Toast a Toastie

```
num_toasties      9
toastie_selected  0
lever_position    0.0
is_toasting      false
toast_time_remaining  0
```



Make It

Toast a Toastie

```
num_toasties      9
toastie_selected  0
lever_position    0.0
is_toasting      false
toast_time_remaining  0
```



Fake Hardware with Software

Fake Hardware with Software



Connected Fireplace and HomeKit

- Projector or TV to simulate controls

Fake Hardware with Software



Connected Fireplace and HomeKit

- Projector or TV to simulate controls



Transit App and Watch

- Interface on an iPod

Fake Hardware with Software



Connected Fireplace and HomeKit

- Projector or TV to simulate controls



Transit App and Watch

- Interface on an iPod



New Drone

- iPhone and paper prototype

How?

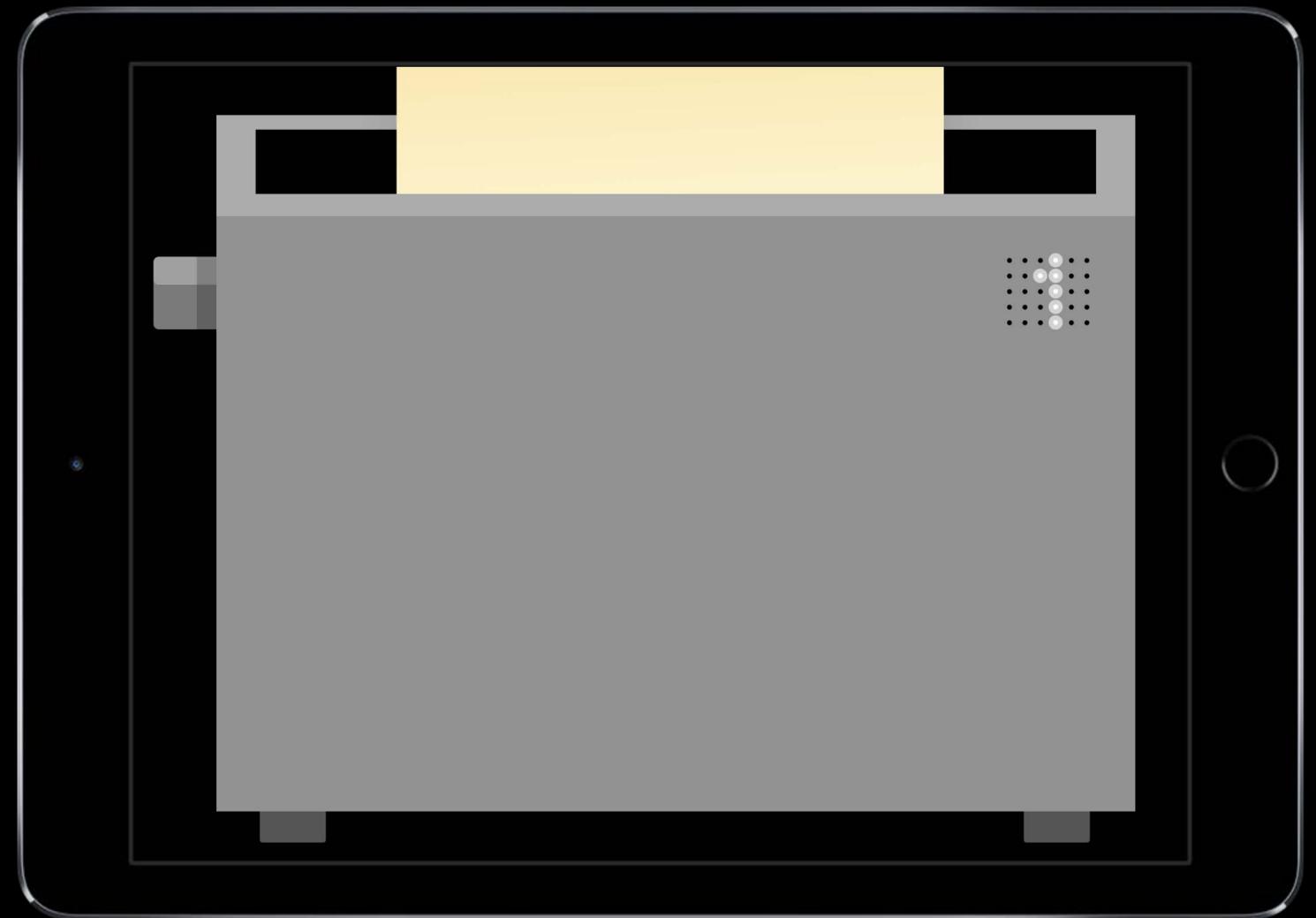
Connect the app and the device



pick a Toastie



physical state



Lightweight Networking

Lightweight Networking



HTML and JavaScript prototypes

- Web Sockets

Lightweight Networking



HTML and JavaScript prototypes

- Web Sockets



Streaming structured data

- OSC Library

Lightweight Networking



HTML and JavaScript prototypes

- Web Sockets



Streaming structured data

- OSC Library

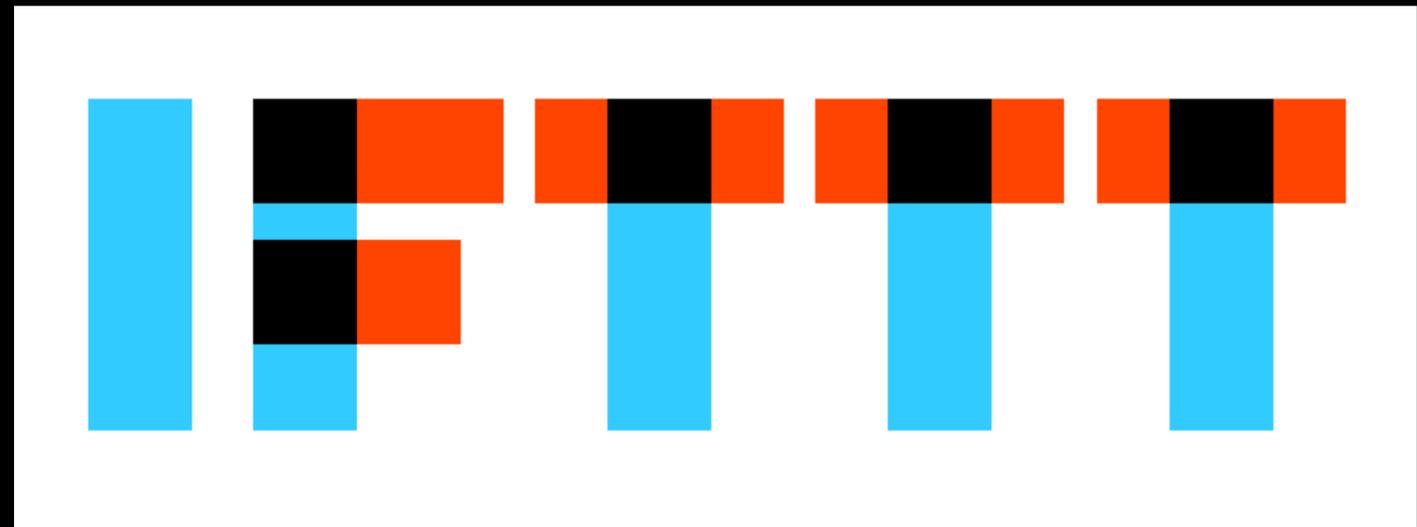
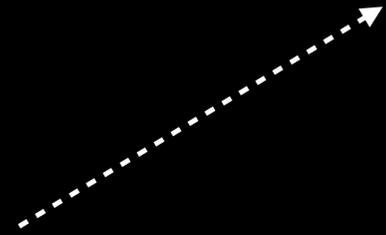
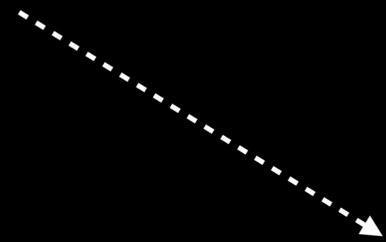


Simple state changes

- HTTP requests

Lightweight Networking

Think creatively!



Make It

Connect the app and the device

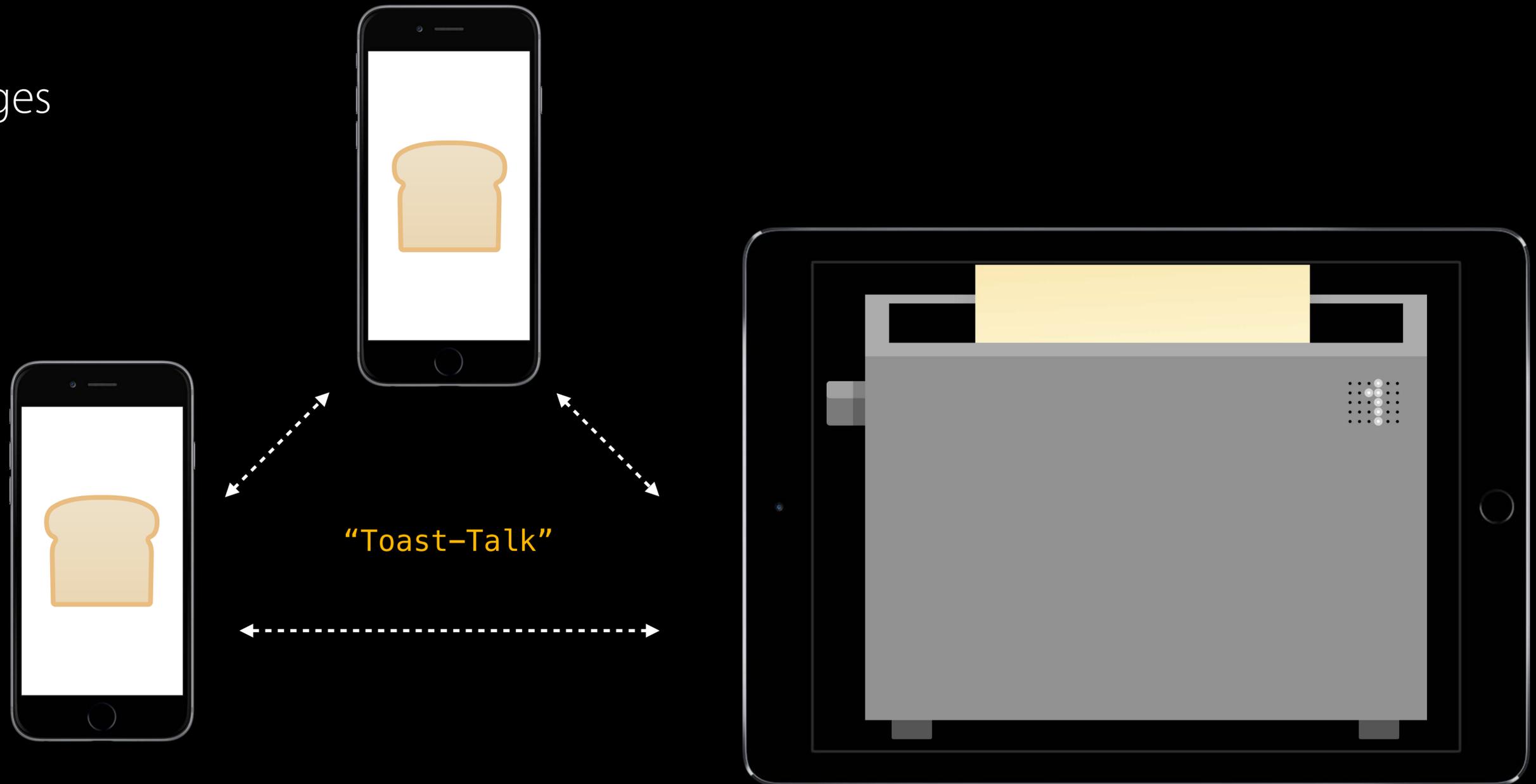
P2P connection using Multipeer Connectivity

Make It

Connect the app and the device

MCSession

- Pass messages



Make It

Connect the app and the device

```
override func viewDidLoad(){
    super.viewDidLoad()

    p2pPeerID = MCPeerID(displayName: UIDevice.currentDevice().name)
    p2pSession = MCSession(peer: p2pPeerID)
    p2pSession.delegate = self

    p2pAssistant = MCAdvertiserAssistant(serviceType: "Toast-Talk",
        discoveryInfo: nil, session: p2pSession)

    p2pAssistant.start()
}

func session(session: MCSession, didReceiveData data: NSData,
    fromPeer peerID: MCPeerID){

    // Pass it into the main thread as a string
    dispatch_async(dispatch_get_main_queue()){
        var message = NSString(data: data, encoding: NSUTF8StringEncoding)
        self.handleToastMessage(String(message!))
    }
}

// All other required delegate methods can be empty
```

Make It

Connect the app and the device

```
override func viewDidLoad(){
    super.viewDidLoad()

    p2pPeerID = MCPeerID(displayName: UIDevice.currentDevice().name)
    p2pSession = MCSession(peer: p2pPeerID)
    p2pSession.delegate = self

    p2pAssistant = MCAdvertiserAssistant(serviceType: "Toast-Talk",
        discoveryInfo: nil, session: p2pSession)

    p2pAssistant.start()
}

func session(session: MCSession, didReceiveData data: NSData,
    fromPeer peerID: MCPeerID){

    // Pass it into the main thread as a string
    dispatch_async(dispatch_get_main_queue()){
        var message = NSString(data: data, encoding: NSUTF8StringEncoding)
        self.handleToastMessage(String(message!))
    }
}

// All other required delegate methods can be empty
```

Make It

Connect the app and the device

```
override func viewDidLoad(){  
    super.viewDidLoad()
```

```
    p2pPeerID = MCPeerID(displayName: UIDevice.currentDevice().name)  
    p2pSession = MCSession(peer: p2pPeerID)  
    p2pSession.delegate = self  
  
    p2pAssistant = MCAvertiserAssistant(serviceType: "Toast-Talk",  
                                       discoveryInfo: nil, session: p2pSession)  
  
    p2pAssistant.start()
```

```
}  
  
func session(session: MCSession, didReceiveData data: NSData,  
            fromPeer peerID: MCPeerID){  
  
    // Pass it into the main thread as a string  
    dispatch_async(dispatch_get_main_queue()){  
        var message = NSString(data: data, encoding: NSUTF8StringEncoding)  
        self.handleToastMessage(String(message!))  
    }  
}
```

```
// All other required delegate methods can be empty
```

Make It

Connect the app and the device

```
override func viewDidLoad(){
    super.viewDidLoad()

    p2pPeerID = MCPeerID(displayName: UIDevice.currentDevice().name)
    p2pSession = MCSession(peer: p2pPeerID)
    p2pSession.delegate = self

    p2pAssistant = MCAdvertiserAssistant(serviceType: "Toast-Talk",
        discoveryInfo: nil, session: p2pSession)

    p2pAssistant.start()
}

func session(session: MCSession, didReceiveData data: NSData,
    fromPeer peerID: MCPeerID){

    // Pass it into the main thread as a string
    dispatch_async(dispatch_get_main_queue()){
        var message = NSString(data: data, encoding: NSUTF8StringEncoding)
        self.handleToastMessage(String(message!))
    }
}

// All other required delegate methods can be empty
```

Make It

Connect the app and the device

```
override func viewDidLoad(){
    super.viewDidLoad()

    p2pPeerID = MCPeerID(displayName: UIDevice.currentDevice().name)
    p2pSession = MCSession(peer: p2pPeerID)
    p2pSession.delegate = self

    p2pAssistant = MCAvertiserAssistant(serviceType: "Toast-Talk",
        discoveryInfo: nil, session: p2pSession)

    p2pAssistant.start()
}

func session(session: MCSession, didReceiveData data: NSData,
    fromPeer peerID: MCPeerID){
    // Pass it into the main thread as a string
    dispatch_async(dispatch_get_main_queue()){
        var message = NSString(data: data, encoding: NSUTF8StringEncoding)
        self.handleToastMessage(String(message!))
    }
}

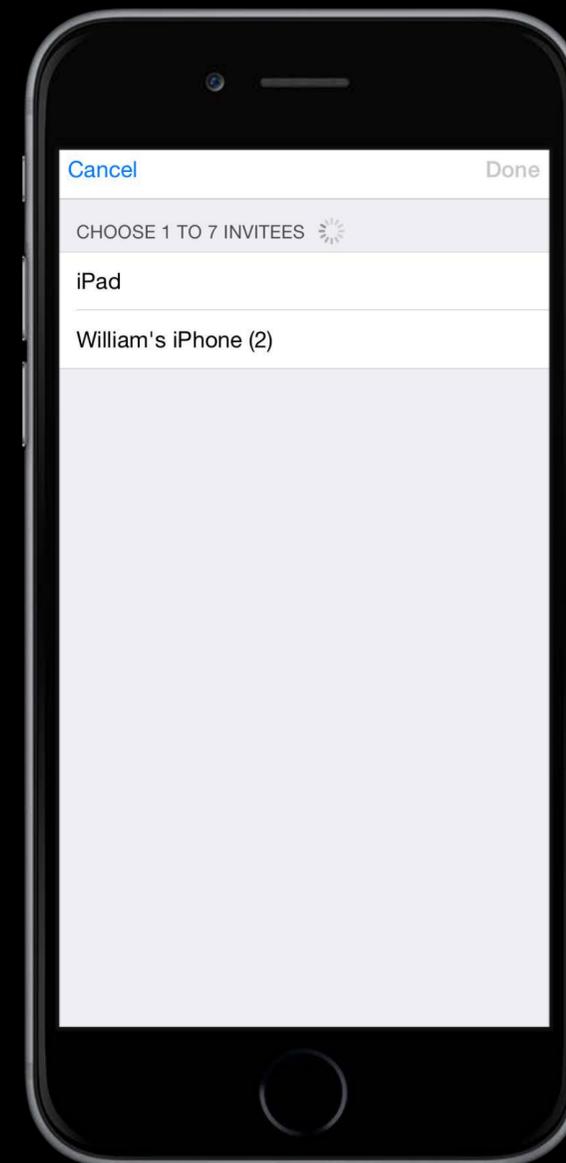
// All other required delegate methods can be empty
```

Make It

Connect the app and the device

MCBrowserViewController

- Choose devices



Make It

Connect the app and the device

```
override func viewDidLoad(){
    // Session stuff...

    p2pBrowser = MCBrowserViewController(serviceType: "Toast-Talk", session: p2pSession)
    p2pBrowser.delegate = self
}

@IBAction func buttonConnectPressed(sender : UIButton){
    presentViewController(p2pBrowser, animated: true, completion: nil)
}

func browserViewControllerDidFinish(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}

func browserViewControllerWasCancelled(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}
```

Make It

Connect the app and the device

```
override func viewDidLoad(){
    // Session stuff...

    p2pBrowser = MCBrowserViewController(serviceType: "Toast-Talk", session: p2pSession)
    p2pBrowser.delegate = self
}

@IBAction func buttonConnectPressed(sender : UIButton){
    presentViewController(p2pBrowser, animated: true, completion: nil)
}

func browserViewControllerDidFinish(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}

func browserViewControllerWasCancelled(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}
```

Make It

Connect the app and the device

```
override func viewDidLoad(){
    // Session stuff...

    p2pBrowser = MCBrowserViewController(serviceType: "Toast-Talk", session: p2pSession)
    p2pBrowser.delegate = self
}

@IBAction func buttonConnectPressed(sender : UIButton){
    presentViewController(p2pBrowser, animated: true, completion: nil)
}

func browserViewControllerDidFinish(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}

func browserViewControllerWasCancelled(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}
```

Make It

Connect the app and the device

```
override func viewDidLoad(){
    // Session stuff...

    p2pBrowser = MCBrowserViewController(serviceType: "Toast-Talk", session: p2pSession)
    p2pBrowser.delegate = self
}

@IBAction func buttonConnectPressed(sender : UIButton){
    presentViewController(p2pBrowser, animated: true, completion: nil)
}

func browserViewControllerDidFinish(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}

func browserViewControllerWasCancelled(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}
```

Make It

Connect the app and the device

```
override func viewDidLoad(){
    // Session stuff...

    p2pBrowser = MCBrowserViewController(serviceType: "Toast-Talk", session: p2pSession)
    p2pBrowser.delegate = self
}

@IBAction func buttonConnectPressed(sender : UIButton){
    presentViewController(p2pBrowser, animated: true, completion: nil)
}

func browserViewControllerDidFinish(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}

func browserViewControllerWasCancelled(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}
```

Make It

Connect the app and the device

```
override func viewDidLoad(){
    // Session stuff...

    p2pBrowser = MCBrowserViewController(serviceType: "Toast-Talk", session: p2pSession)
    p2pBrowser.delegate = self
}

@IBAction func buttonConnectPressed(sender : UIButton){
    presentViewController(p2pBrowser, animated: true, completion: nil)
}

func browserViewControllerDidFinish(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}

func browserViewControllerWasCancelled(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}
```

Make It

Connect the app and the device

```
override func viewDidLoad(){
    // Session stuff...

    p2pBrowser = MCBrowserViewController(serviceType: "Toast-Talk", session: p2pSession)
    p2pBrowser.delegate = self
}

@IBAction func buttonConnectPressed(sender : UIButton){
    presentViewController(p2pBrowser, animated: true, completion: nil)
}
```

```
func browserViewControllerDidFinish(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}

func browserViewControllerWasCancelled(browserViewController: MCBrowserViewController!){
    dismissViewControllerAnimated(true, completion: nil)
}
```

Make It

Connect the app and the device

What messages?

Make It

Connect the app and the device

What messages?

App proto

`toastie_number`

`toast_darkness`

Make It

Connect the app and the device

What messages?

App proto

`toastie_number`

`toast_darkness`

Toaster

`lever_position`

`toasting_began`

`toasting_complete`

Make It

Connect the app and the device

```
func broadcastMessage(message: String){
    p2pSession.sendData( message.dataUsingEncoding(NSUTF8StringEncoding,
        allowLossyConversion: false )!,
        toPeers: p2pSession.connectedPeers,
        withMode: MCSessionSendDataMode.Reliable,
        error: nil)
}

// Example of sending a message from the Toaster

func sendLeverPosition(){
    broadcastMessage("lever_position^\(self.sliderLever!.value)")
}
```

Make It

Connect the app and the device

```
func broadcastMessage(message: String){
    p2pSession.sendData( message.dataUsingEncoding(NSUTF8StringEncoding,
        allowLossyConversion: false )!,
        toPeers: p2pSession.connectedPeers,
        withMode: MCSessionSendDataMode.Reliable,
        error: nil)
}

// Example of sending a message from the Toaster

func sendLeverPosition(){
    broadcastMessage("lever_position^\(self.sliderLever!.value)")
}
```

Make It

Connect the app and the device

```
func broadcastMessage(message: String){  
    p2pSession.sendData( message.dataUsingEncoding(NSUTF8StringEncoding,  
        allowLossyConversion: false )!,  
        toPeers: p2pSession.connectedPeers,  
        withMode: MCSessionSendDataMode.Reliable,  
        error: nil)  
}
```

```
// Example of sending a message from the Toaster
```

```
func sendLeverPosition(){  
    broadcastMessage("lever_position^\(self.sliderLever!.value)")  
}
```

Make It

Connect the app and the device

```
func broadcastMessage(message: String){
    p2pSession.sendData( message.dataUsingEncoding(NSUTF8StringEncoding,
        allowLossyConversion: false )!,
        toPeers: p2pSession.connectedPeers,
        withMode: MCSessionSendDataMode.Reliable,
        error: nil)
}

// Example of sending a message from the Toaster

func sendLeverPosition(){
    broadcastMessage("lever_position^\(self.sliderLever!.value)")
}
```

Make It

Connect the app and the device

```
func broadcastMessage(message: String){
    p2pSession.sendData( message.dataUsingEncoding(NSUTF8StringEncoding,
        allowLossyConversion: false )!,
        toPeers: p2pSession.connectedPeers,
        withMode: MCSessionSendDataMode.Reliable,
        error: nil)
}

// Example of sending a message from the Toaster

func sendLeverPosition(){
    broadcastMessage("lever_position^\(self.sliderLever!.value)")
}
```

Make It

Connect the app and the device

```
func handleToastMessage(message : String){
    let messageWithParams : [String] = message.componentsSeparatedByString("^")
    let messageName = messageWithParams[0]

    switch messageName{
    case "lever_position":
        let position = messageWithParams[1]
        updateRemoteLeverPosition((position as NSString).doubleValue)

        // Handle the other messages here...

    default:
        break
    }
}
```

Make It

Connect the app and the device

```
func handleToastMessage(message : String){
    let messageWithParams : [String] = message.componentsSeparatedByString("^")
    let messageName = messageWithParams[0]

    switch messageName{
    case "lever_position":
        let position = messageWithParams[1]
        updateRemoteLeverPosition((position as NSString).doubleValue)

        // Handle the other messages here...

    default:
        break
    }
}
```

Make It

Connect the app and the device

```
func handleToastMessage(message : String){
    let messageWithParams : [String] = message.componentsSeparatedByString("^")
    let messageName = messageWithParams[0]

    switch messageName{
    case "lever_position":
        let position = messageWithParams[1]
        updateRemoteLeverPosition((position as NSString).doubleValue)

        // Handle the other messages here...

    default:
        break
    }
}
```

Make It

Connect the app and the device

```
func handleToastMessage(message : String){
    let messageWithParams : [String] = message.componentsSeparatedByString("^")
    let messageName = messageWithParams[0]

    switch messageName{
    case "lever_position":
        let position = messageWithParams[1]
        updateRemoteLeverPosition((position as NSString).doubleValue)

        // Handle the other messages here...

    default:
        break
    }
}
```

Make It

Connect the app and the device

```
func handleToastMessage(message : String){
    let messageWithParams : [String] = message.componentsSeparatedByString("^")
    let messageName = messageWithParams[0]

    switch messageName{
    case "lever_position":
        let position = messageWithParams[1]
        updateRemoteLeverPosition((position as NSString).doubleValue)

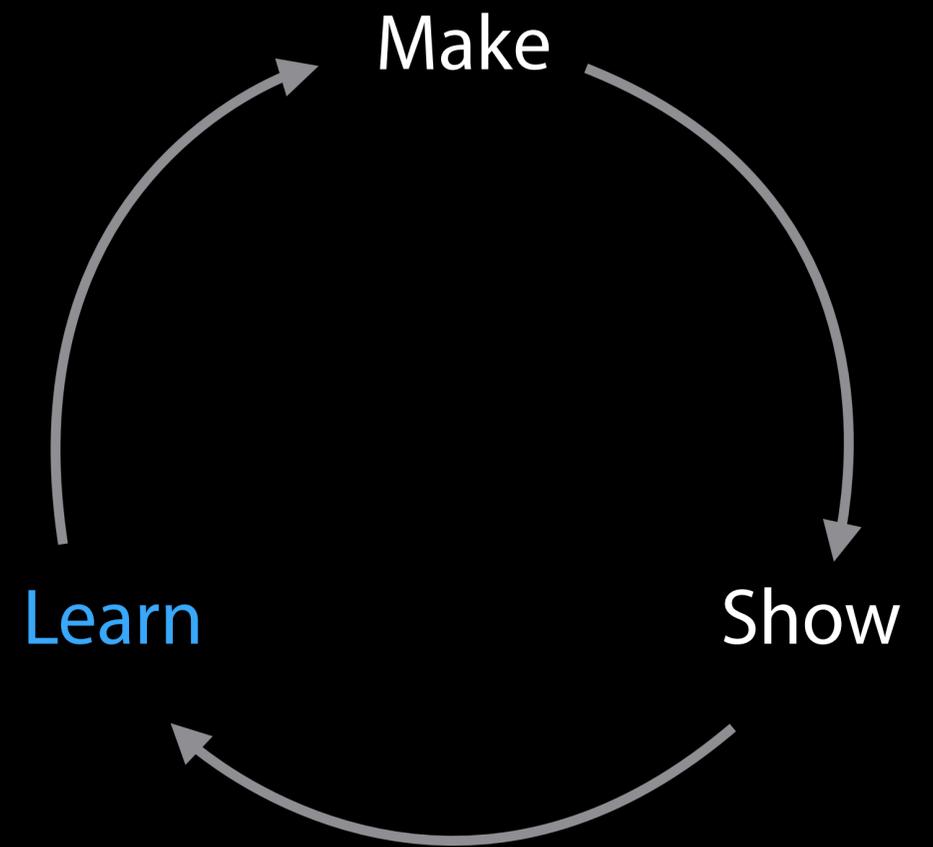
        // Handle the other messages here...

    default:
        break
    }
}
```



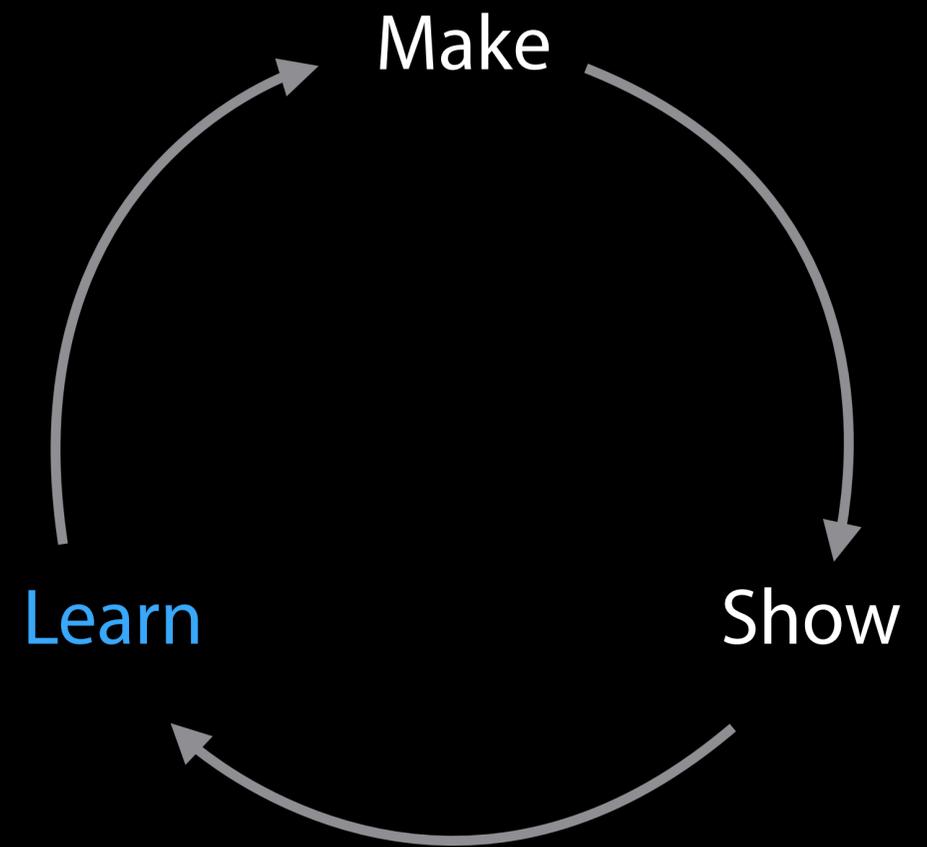


Learn from Feedback



Learn from Feedback

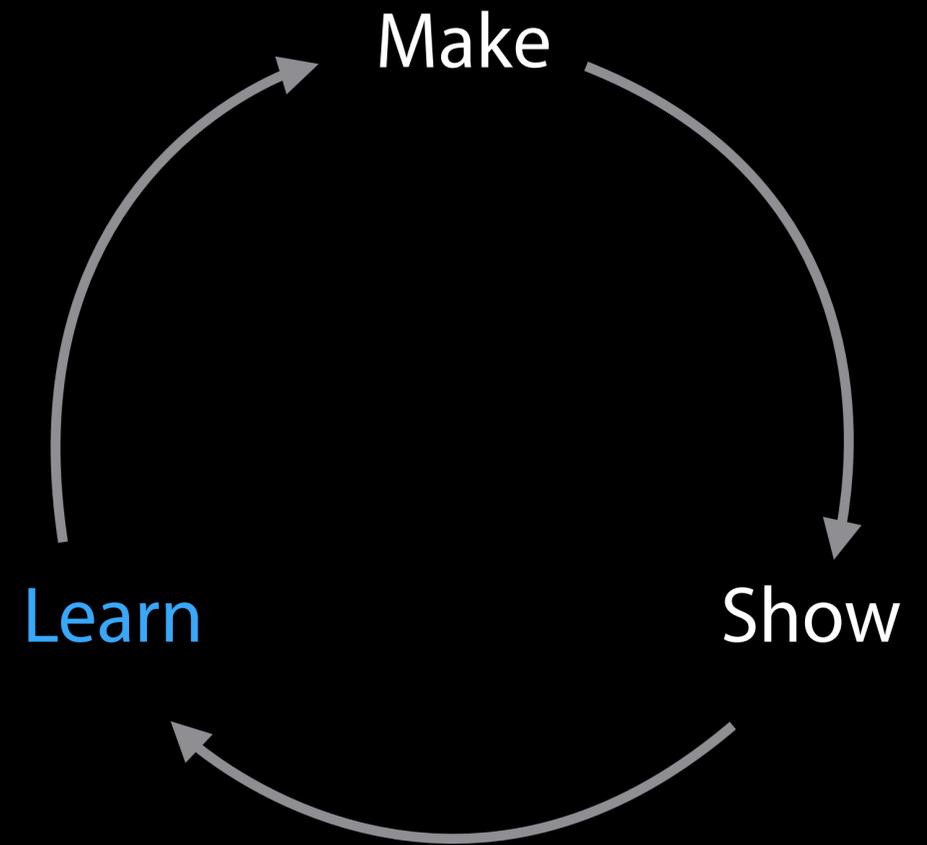
What's working?



Learn from Feedback

What's working?

Picking Toasties from the App, toasting them on the device

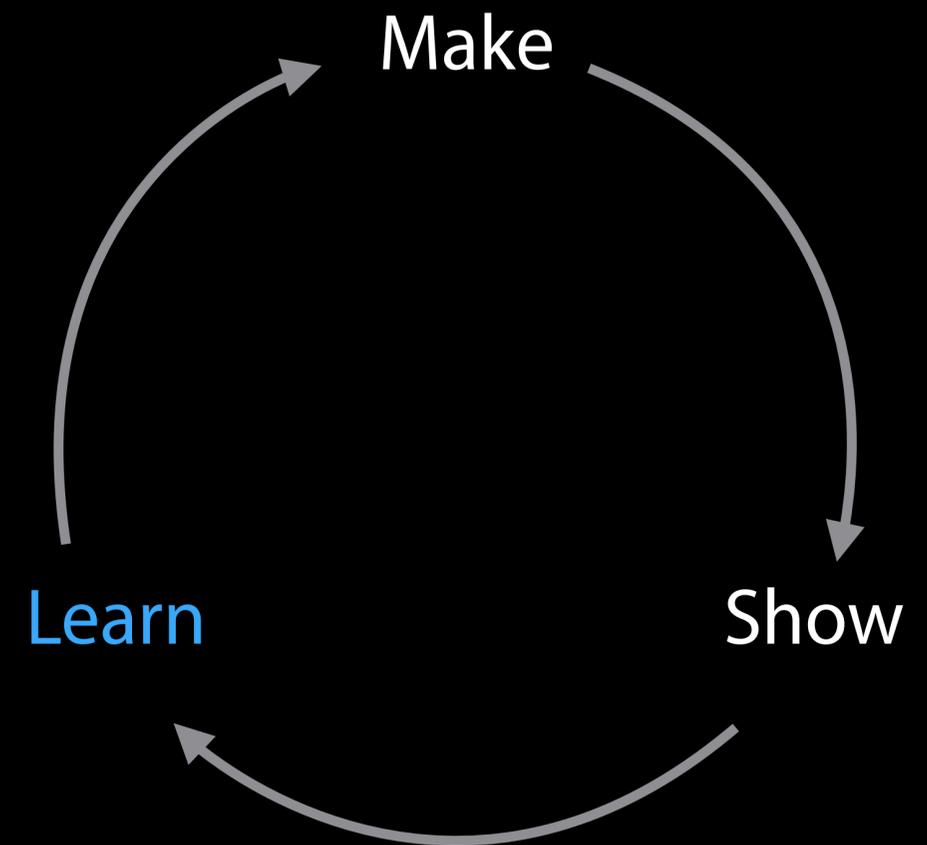


Learn from Feedback

What's working?

Picking Toasties from the App, toasting them on the device

What's not working?



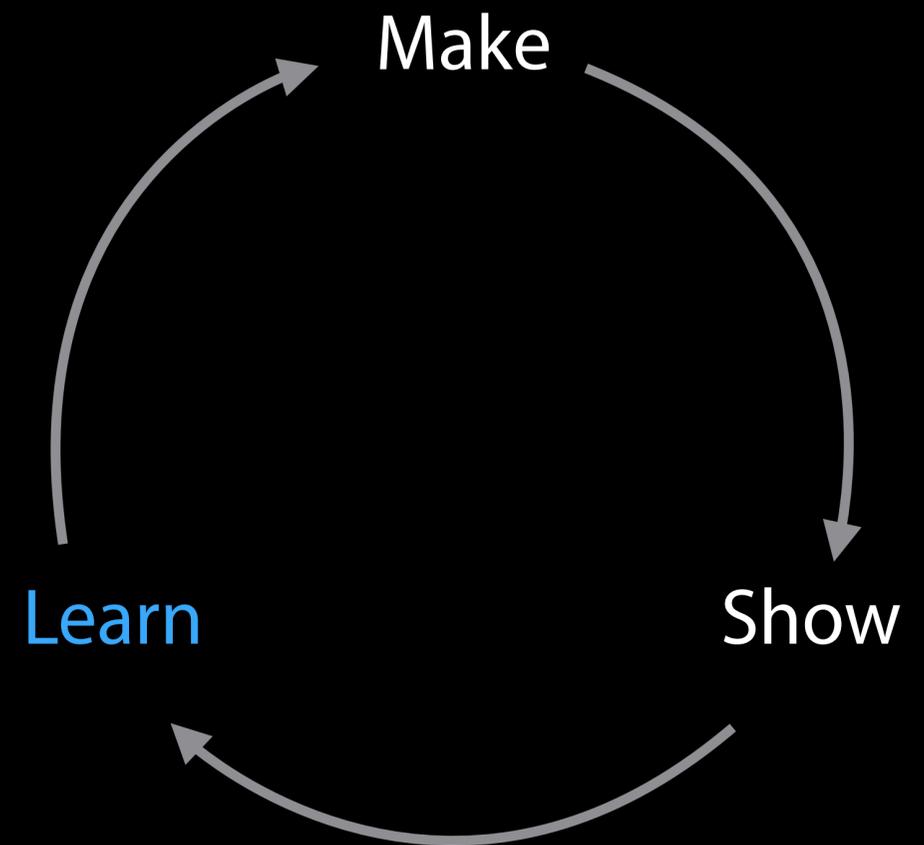
Learn from Feedback

What's working?

Picking Toasties from the App, toasting them on the device

What's not working?

The Toaster doesn't give us any visual feedback when we adjust the color



Learn from Feedback

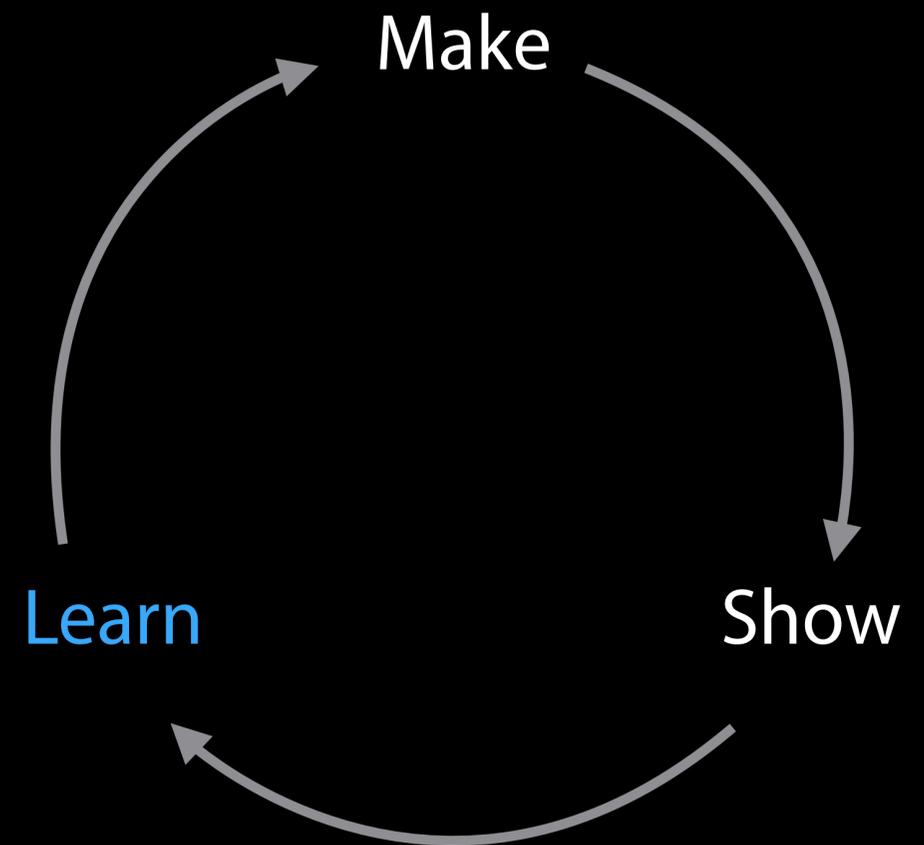
What's working?

Picking Toasties from the App, toasting them on the device

What's not working?

The Toaster doesn't give us any visual feedback when we adjust the color

What ideas does this give us?



Learn from Feedback

What's working?

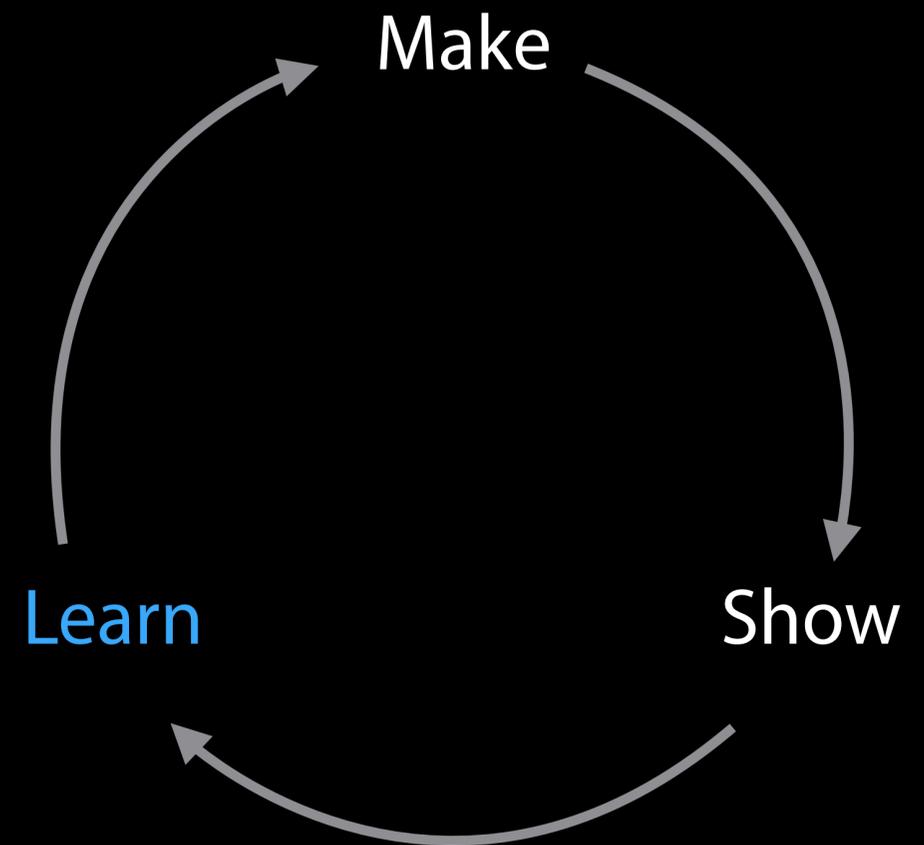
Picking Toasties from the App, toasting them on the device

What's not working?

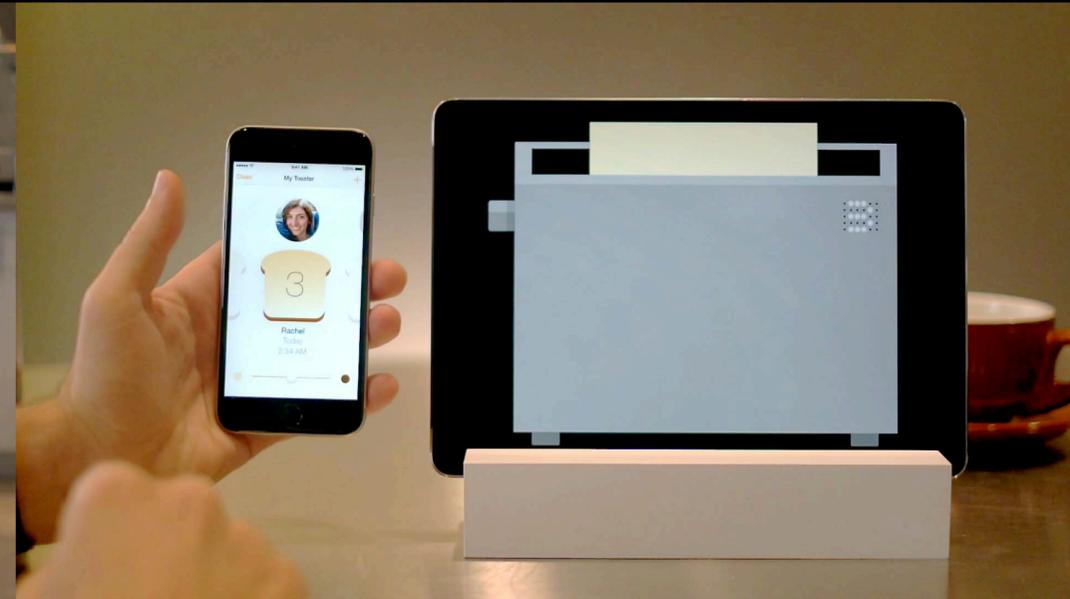
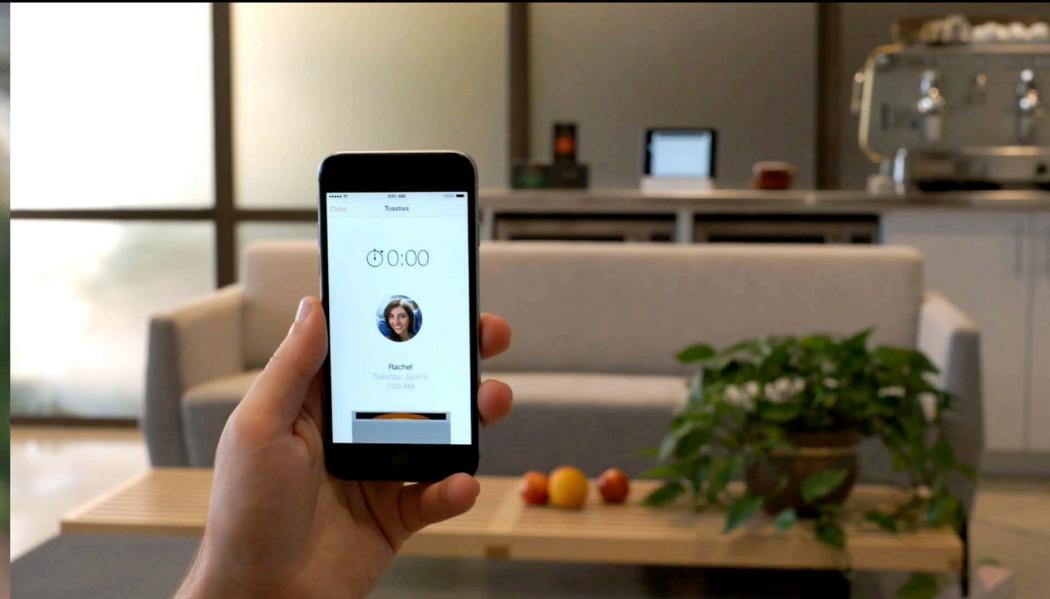
The Toaster doesn't give us any visual feedback when we adjust the color

What ideas does this give us?

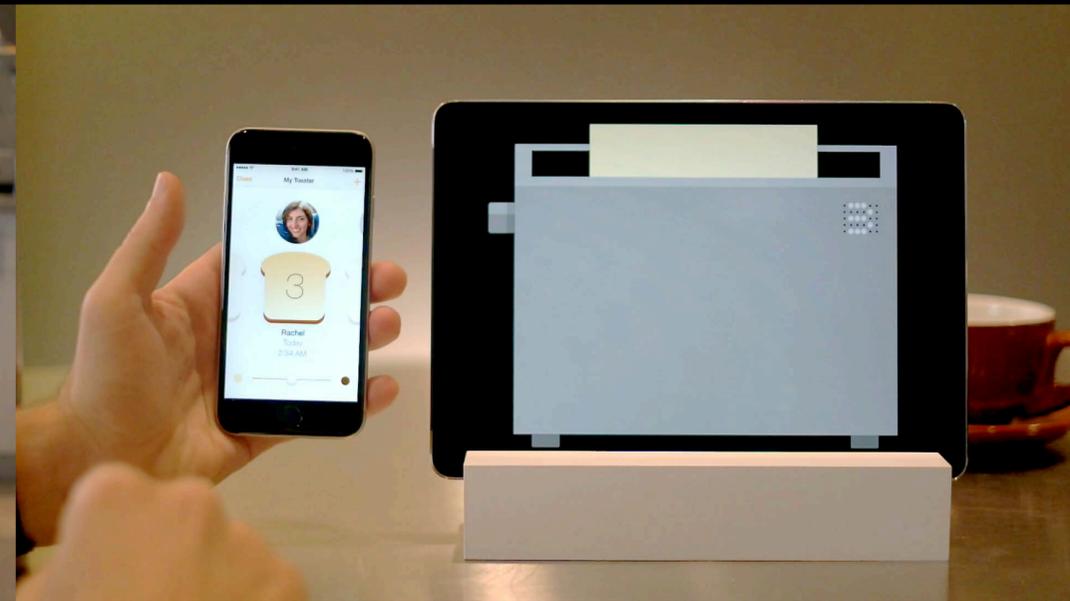
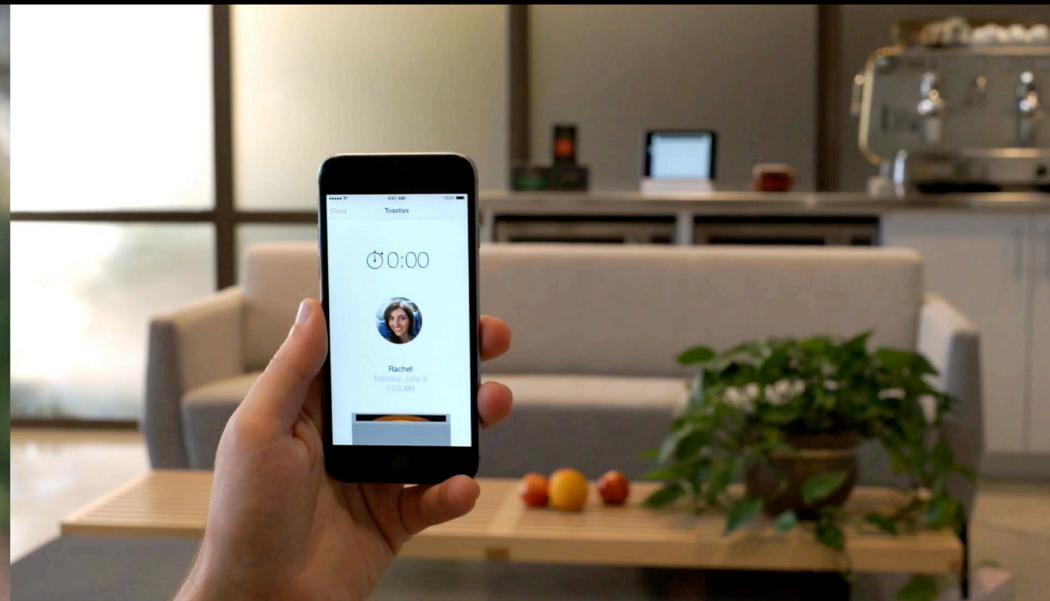
Adjust the color of the LED display to reflect the color setting



Recap

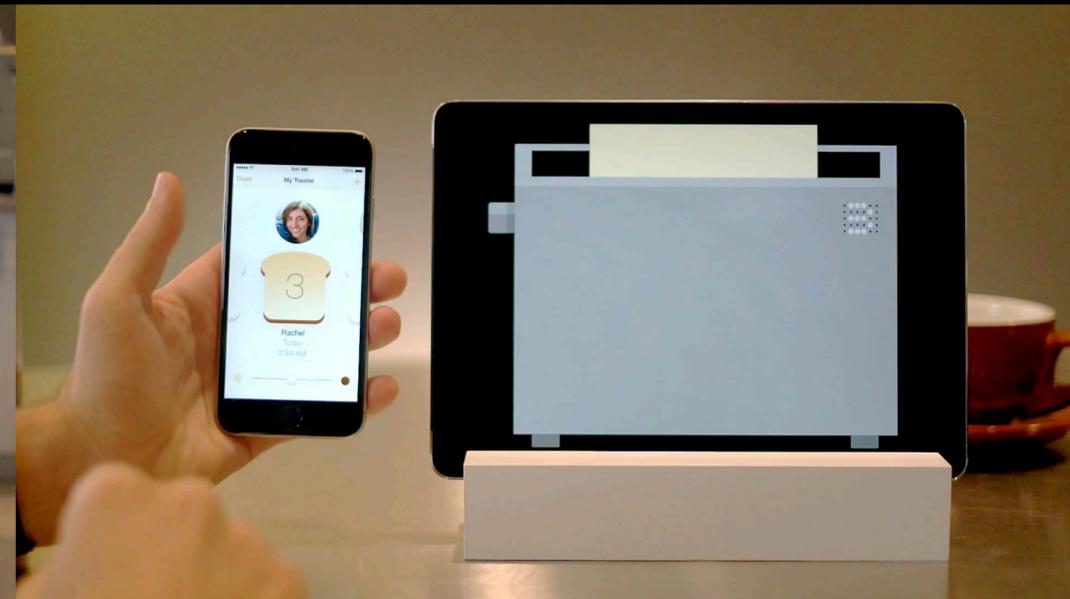
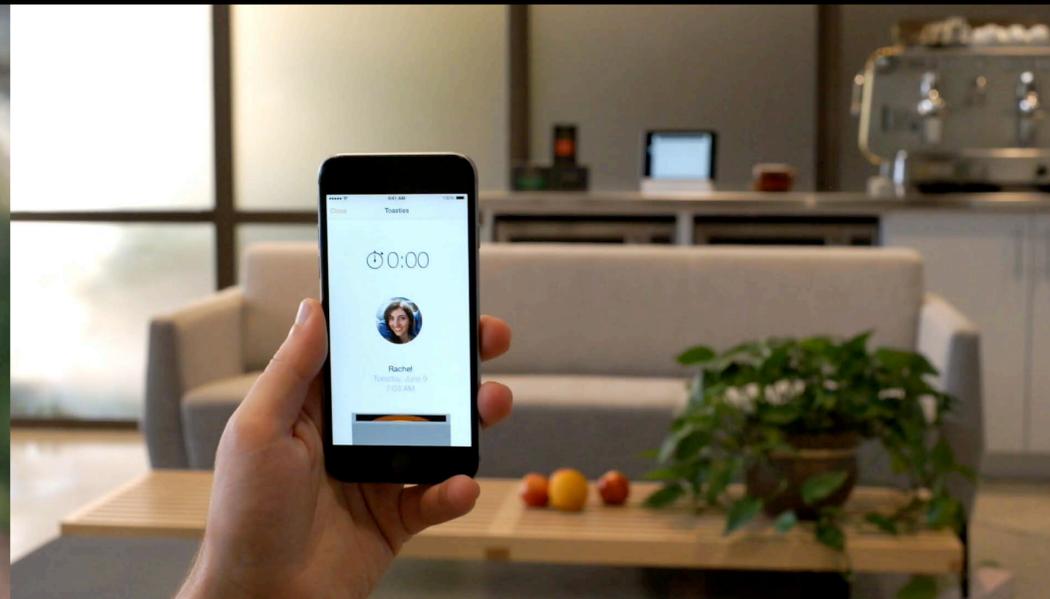


Recap



Pictures and Animation

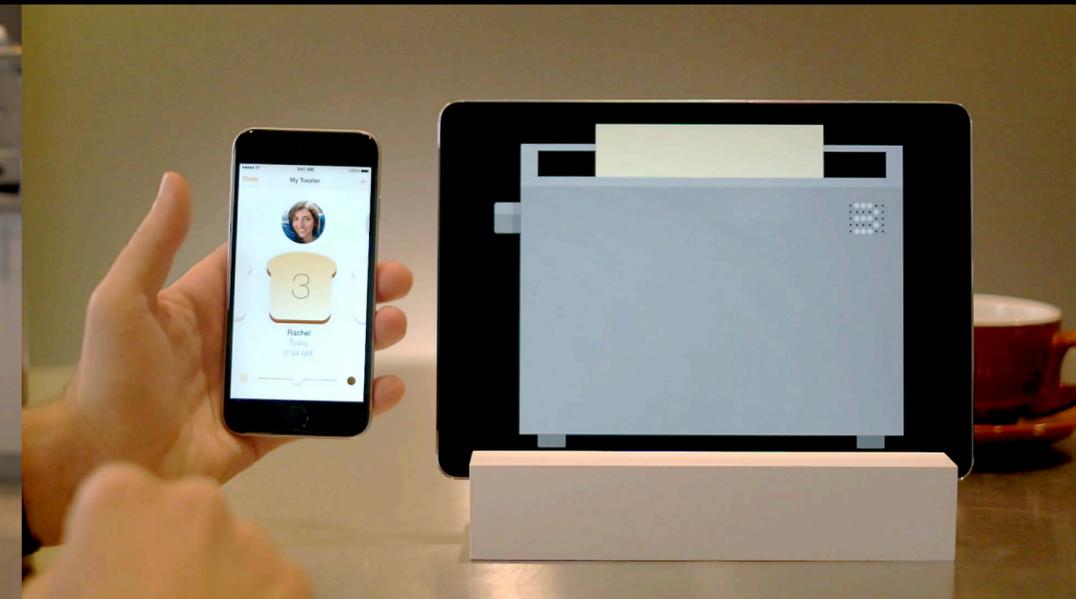
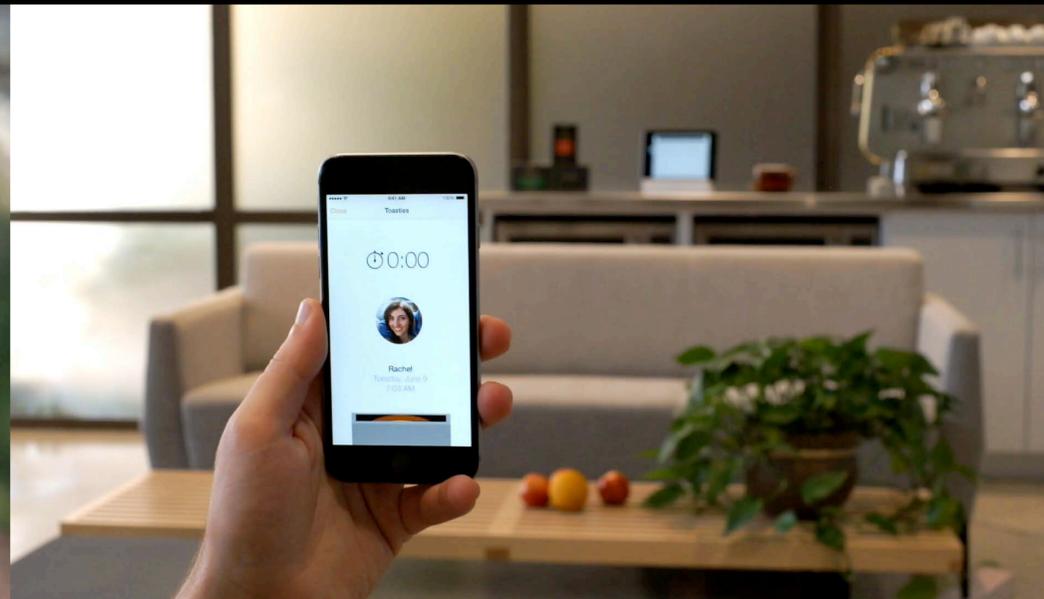
Recap



Pictures and Animation

Try different inputs and outputs really fast

Recap

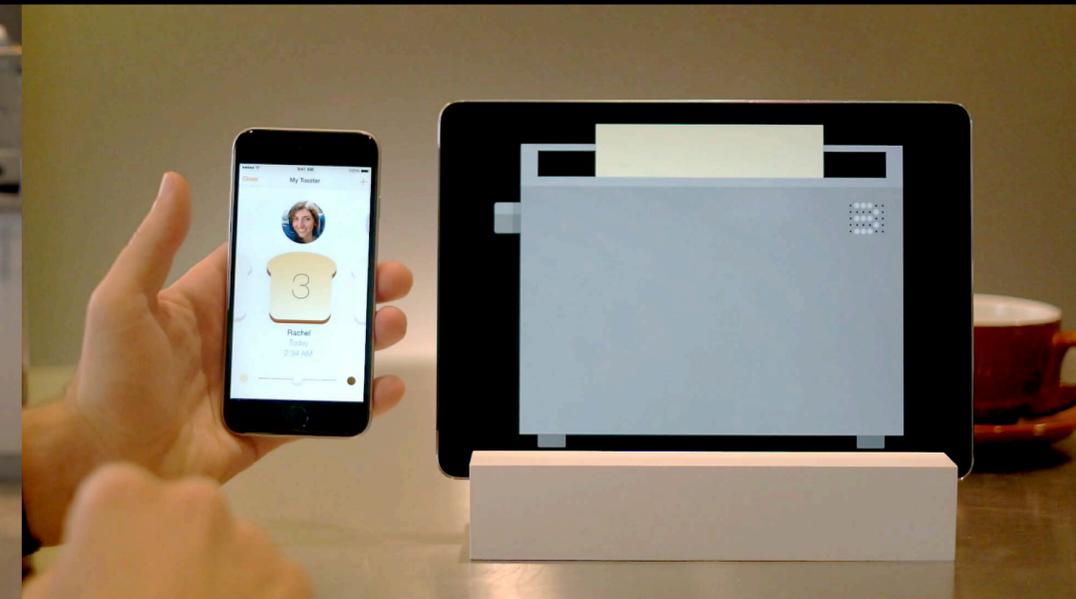
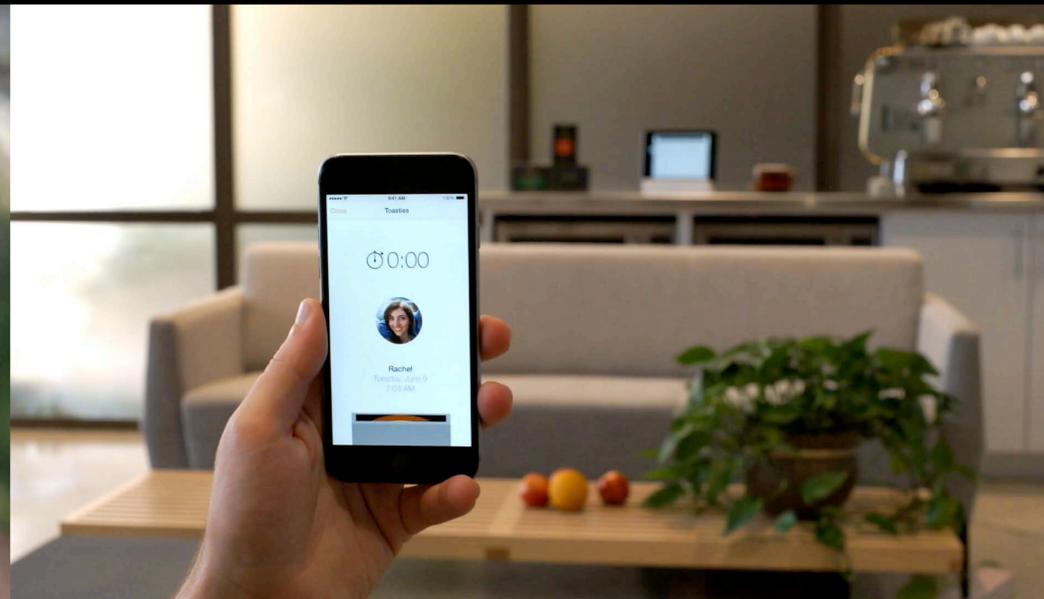


Pictures and Animation

Try different inputs and
outputs really fast

Behind the Curtain

Recap



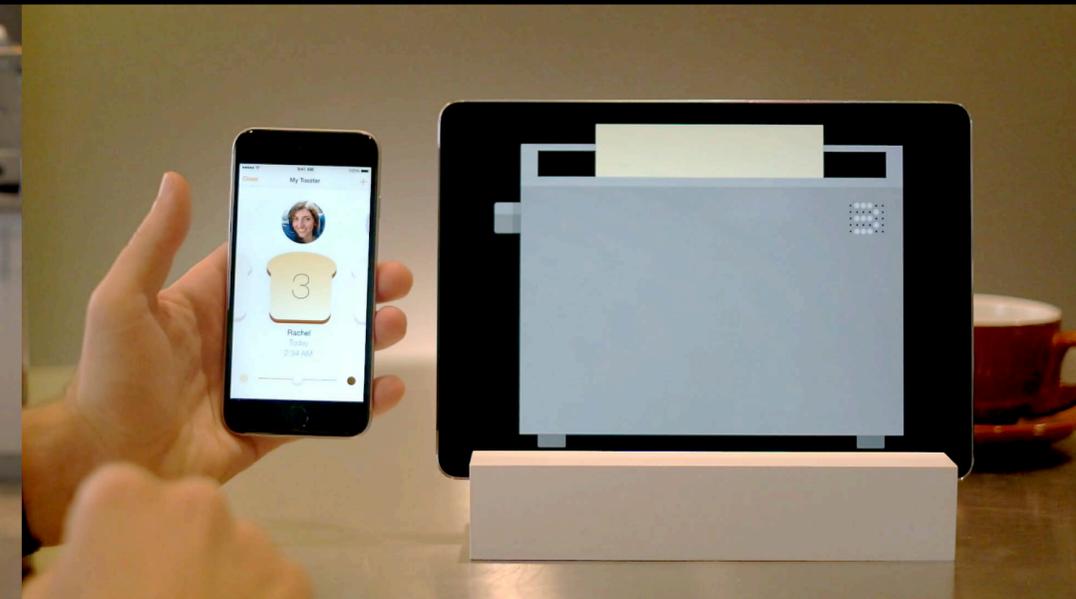
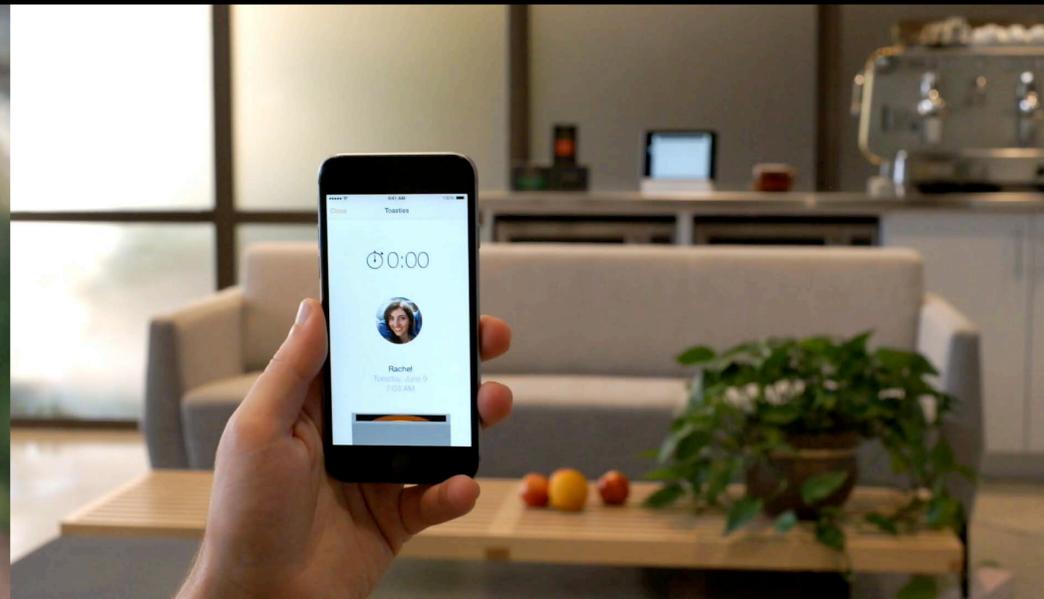
Pictures and Animation

Try different inputs and outputs really fast

Behind the Curtain

See how the app and device worked together

Recap



Pictures and Animation

Try different inputs and outputs really fast

Behind the Curtain

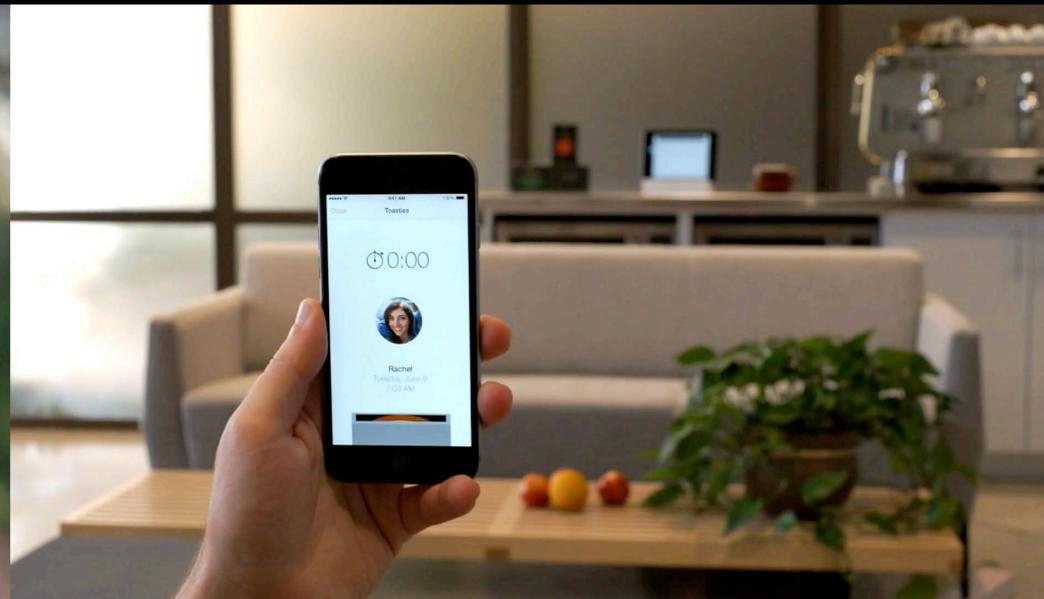
See how the app and device worked together

Interactive and Connected

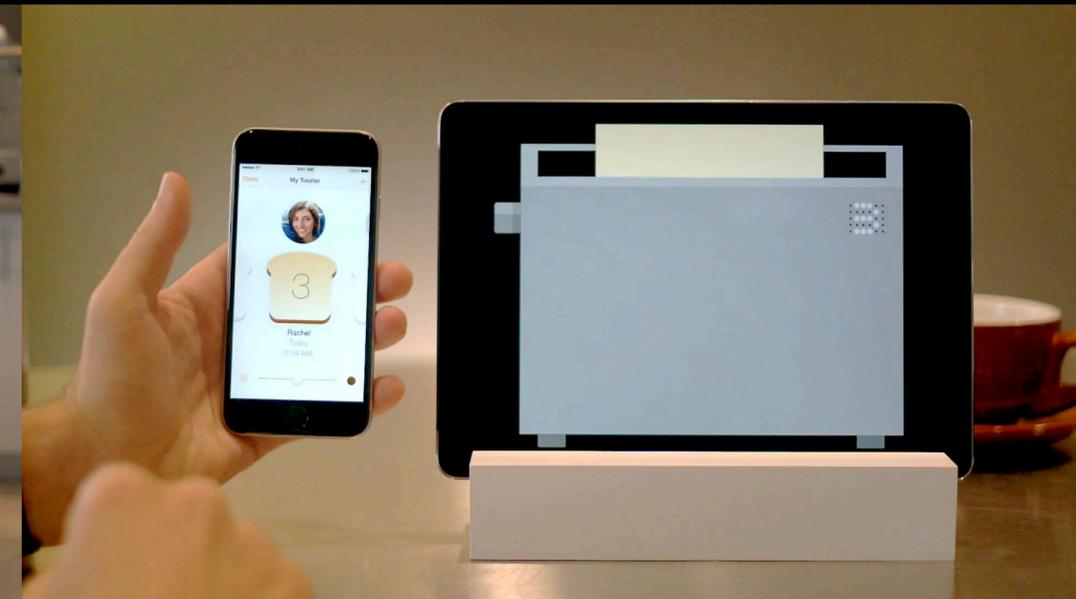
Recap



Pictures and Animation
Try different inputs and
outputs really fast



Behind the Curtain
See how the app and device
worked together



Interactive and Connected
Try the end to end experience

What We Learned About Our Toaster

What We Learned About Our Toaster

We don't need a darkness setting on the toaster

What We Learned About Our Toaster

We don't need a darkness setting on the toaster

We need a number display on the toaster

What We Learned About Our Toaster

We don't need a darkness setting on the toaster

We need a number display on the toaster

We need a sensor on the lever to communicate to the app

What We Learned About Our Toaster

We don't need a darkness setting on the toaster

We need a number display on the toaster

We need a sensor on the lever to communicate to the app

We figured all this out, really fast!

What We Learned About Our Toaster

We don't need a darkness setting on the toaster

We need a number display on the toaster

We need a sensor on the lever to communicate to the app

We figured all this out, really fast!

Look at all the time and money we saved!

What We Learned About Our Toaster

We don't need a darkness setting on the toaster

We need a number display on the toaster

We need a sensor on the lever to communicate to the app

We figured all this out, really fast!

Look at all the time and money we saved!

We got some pretty cool ideas!

Why

How

Why

Test ideas

Save time and money building
the right things

How

Why

Test ideas

Save time and money building
the right things

Get new ideas

Make the experience of your
product even better

How

Why

Test ideas

Save time and money building
the right things

Get new ideas

Make the experience of your
product even better

How

Make fake apps

Why

Test ideas

Save time and money building
the right things

Get new ideas

Make the experience of your
product even better

How

Make fake apps

Show people

Why

Test ideas

Save time and money building
the right things

Get new ideas

Make the experience of your
product even better

How

Make fake apps

Show people

Learn from their feedback

Why

Test ideas

Save time and money building
the right things

Get new ideas

Make the experience of your
product even better

How

Make fake apps

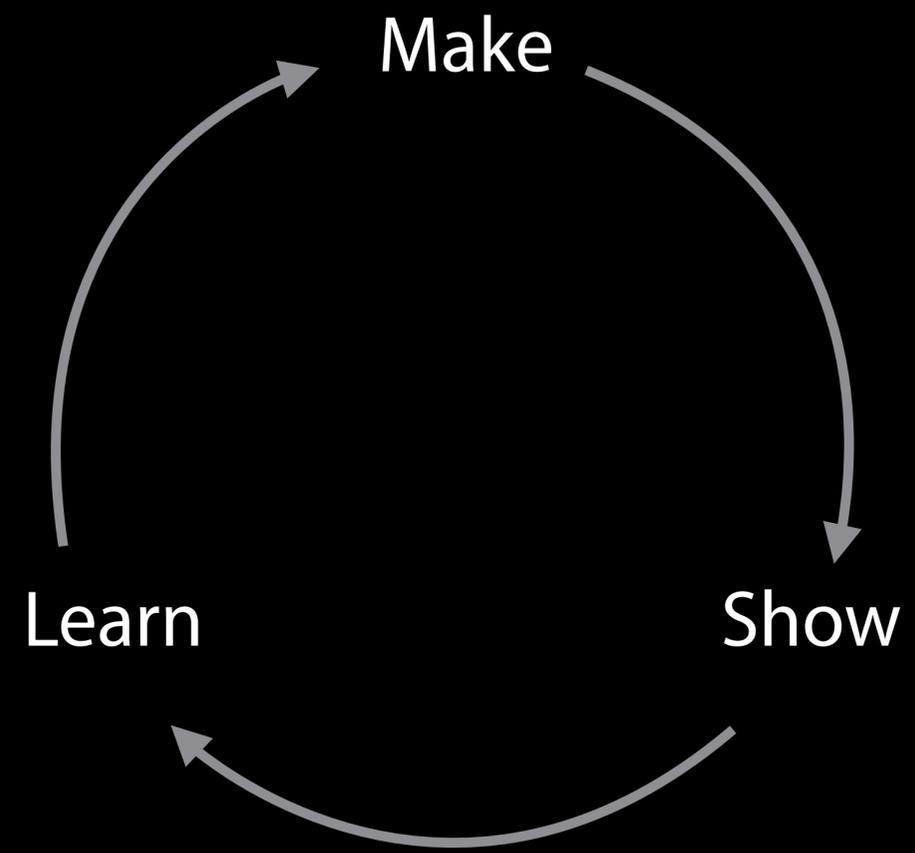
Show people

Learn from their feedback

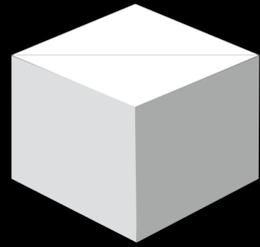
Make

Learn

Show



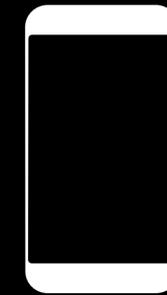
What We Want You to Do



A thing

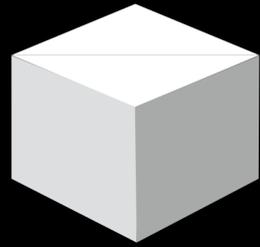


connected to



an app

What We Want You to Do

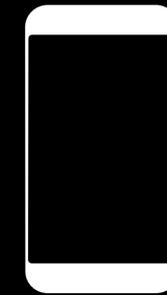


A thing

Fake it on a screen

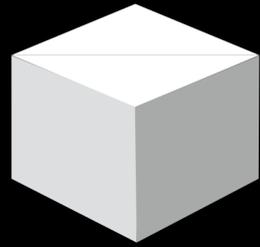


connected to



an app

What We Want You to Do



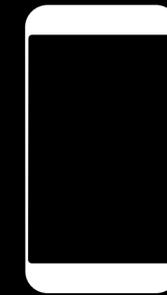
A thing

Fake it on a screen

In context

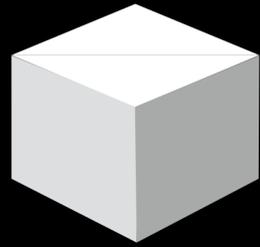


connected to



an app

What We Want You to Do



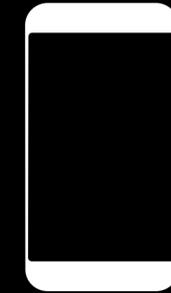
A thing

Fake it on a screen

In context



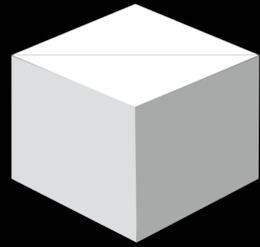
connected to



an app

Fake it with pictures

What We Want You to Do



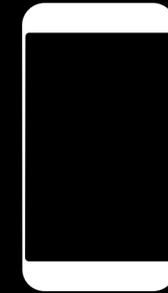
A thing

Fake it on a screen

In context



connected to

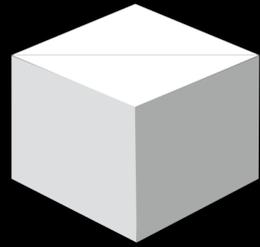


an app

Fake it with pictures

In context

What We Want You to Do



A thing

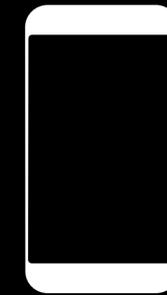
Fake it on a screen

In context



connected to

Tap through both

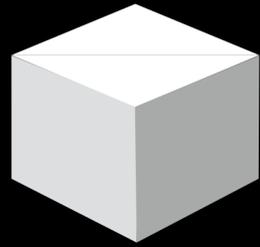


an app

Fake it with pictures

In context

What We Want You to Do



A thing

Fake it on a screen

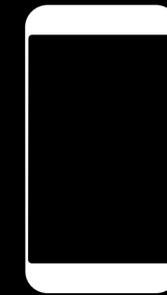
In context



connected to

Tap through both

Behind the curtain

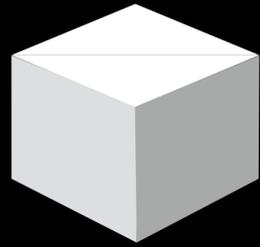


an app

Fake it with pictures

In context

What We Want You to Do



A thing

Fake it on a screen

In context

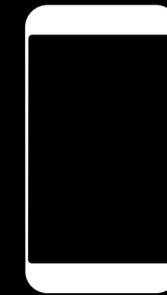


connected to

Tap through both

Behind the curtain

Lightweight networking

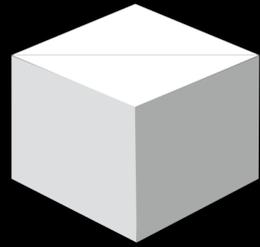


an app

Fake it with pictures

In context

What We Want You to Do



A thing

Fake it on a screen

In context

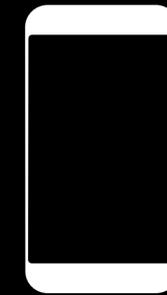


connected to

Tap through both

Behind the curtain

Lightweight networking



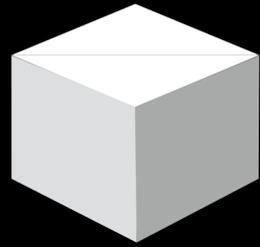
an app

Fake it with pictures

In context

Don't have the device?

What We Want You to Do



A thing

Fake it on a screen

In context

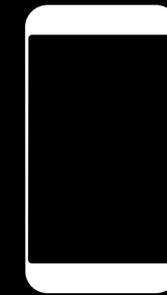


connected to

Tap through both

Behind the curtain

Lightweight networking



an app

Fake it with pictures

In context

Don't have the device? [Don't wait, fake it on a screen, in context](#)

More Information

Documentation

Swift Language Documentation
developer.apple.com/swift

Designing Great Apps

developer.apple.com/design

developer.apple.com/watchkit

WWDC2014

developer.apple.com/videos/wwdc/2014/#223

General Inquiries

Rachel Roth, User Experience Evangelist
rroth@apple.com

Related Sessions

Designing for Apple Watch

Presidio

Wednesday 4:30PM

Watch Design Tips and Tricks

Presidio

Friday 3:30PM

Related Sessions

Designing for Apple Watch

Presidio

Wednesday 4:30PM

Designing with Animation

Presidio

Thursday 3:30PM

Watch Design Tips and Tricks

Presidio

Friday 3:30PM

Labs

Prototyping Lab

Frameworks Lab E Wednesday 3:30PM

User Interface Design Lab

User Interface
Design Lab Wednesday 9:00AM



WWDC 15

 WWDC 15