What's new in iOS13

RYAN DAVIS

Melbourne Xamarin Meetup

2019 11 20

(Some of) What's new in iOS13

RYAN DAVIS

Melbourne Xamarin Meetup

2019 11 20

(More of) What's new in iOS13

RYAN DAVIS

Melbourne Xamarin Meetup

2019 11 20

(More of) What's new in iOS13

for Xamarin developers

RYAN DAVIS

Melbourne Xamarin Meetup

2019 11 20

Homework slide

(More of) What's new in iOS13

for Xamarin developers

RYAN DAVIS

Melbourne Xamarin Meetup

2019 11 20

whoami

- Ryan Davis
- Professional ~~Mobile~~ LINQPad Developer

 ryandavis.io  rdavis_au

 rdavisau

- **essential-interfaces** – use DI/mocking with Xamarin.Essentials
- **lightswitch** – toggle app dark/light mode from the IDE
- **dumpeditable-linqpad** – extensible inline object editor for LINQPad
- **jsondatacontext-linqpad** – json data context driver for LINQPad
- **sockets-for-pcl, sockethelpers –** socket comms in a PCL
  (today you should use netstandard sockets why are you all still installing this)

to cover

- what is iOS13
- what's new
- demos, samples
- resources

~~finally we get dark mode~~ iOS 13

- 13th public release of iOS, shipped alongside Xcode 11 and macOS 15
- Release mid-September, with several updates already
- Lots of built in OS improvements – performance, security, core app updates
- Lots of new features and frameworks for developers
- **Dark mode**

what we get to work with

New | Improved

iOS 13 includes a huge number of new frameworks and improvements to existing ones

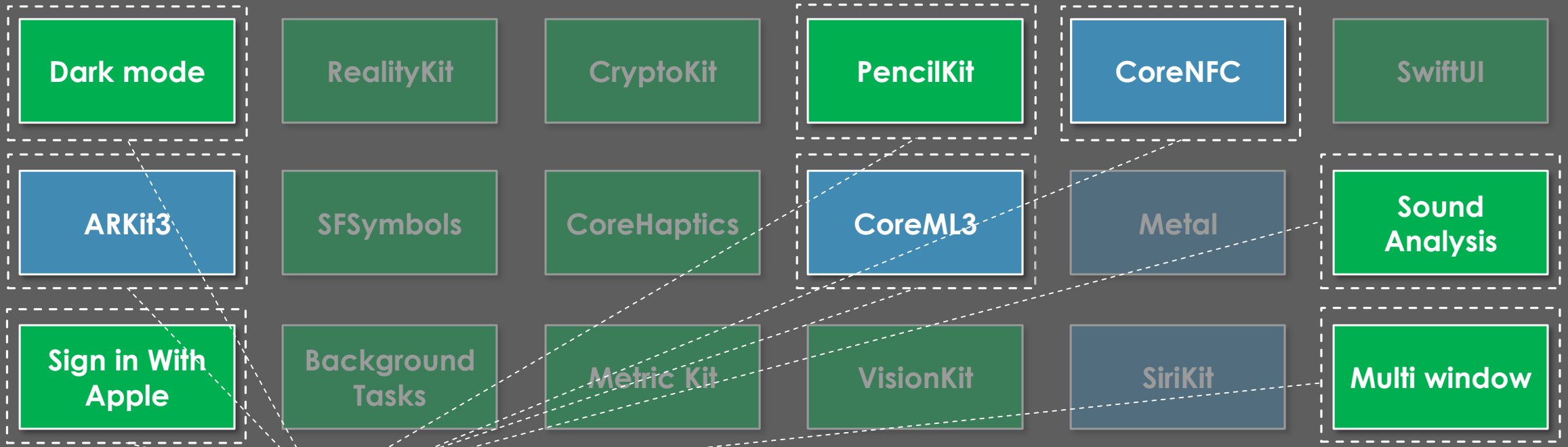
Dark mode	RealityKit	CryptoKit	PencilKit	CoreNFC	SwiftUI
ARKit3	SFSymbols	CoreHaptics	CoreML3	Metal	Sound Analysis
Sign in With Apple	Background Tasks	Metric Kit	VisionKit	SiriKit	Multi window

Maybe we just look at these today

what we get to work with

New | Improved

iOS 13 includes a huge number of new frameworks and improvements to existing ones



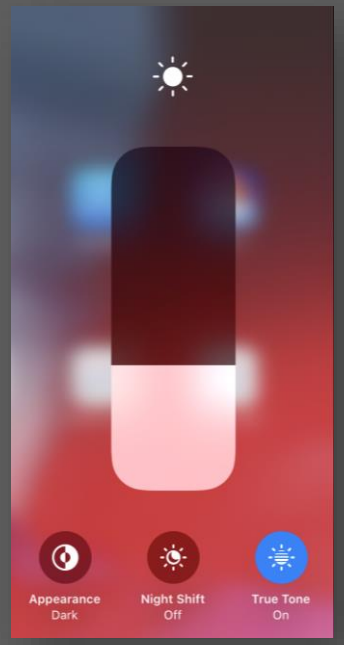
Maybe we just look at these today

-= some of what's new in ios13 =-

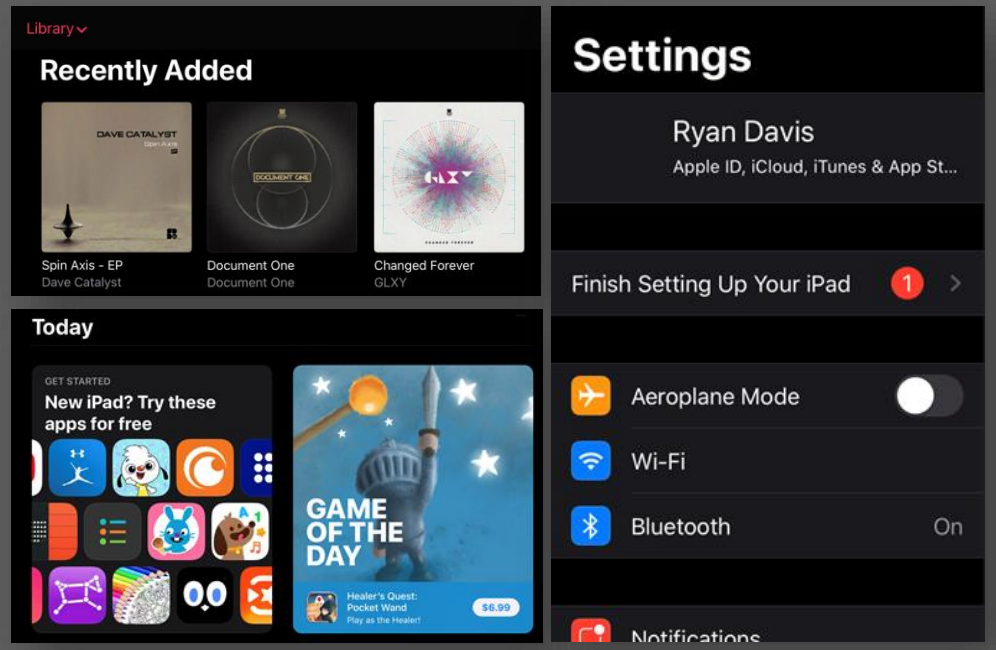
dark mode

~~leave now if you aren't using dark mode~~

iOS 13 includes a system-wide dark/light theme setting



Schedule-based or manually toggled



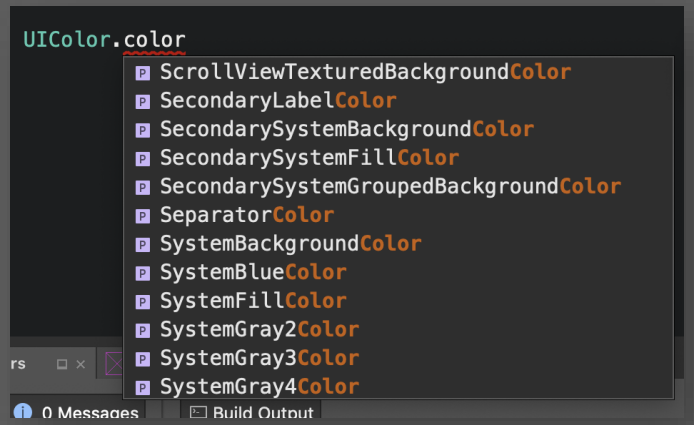
Observed by all built-in apps

Observed by your app?

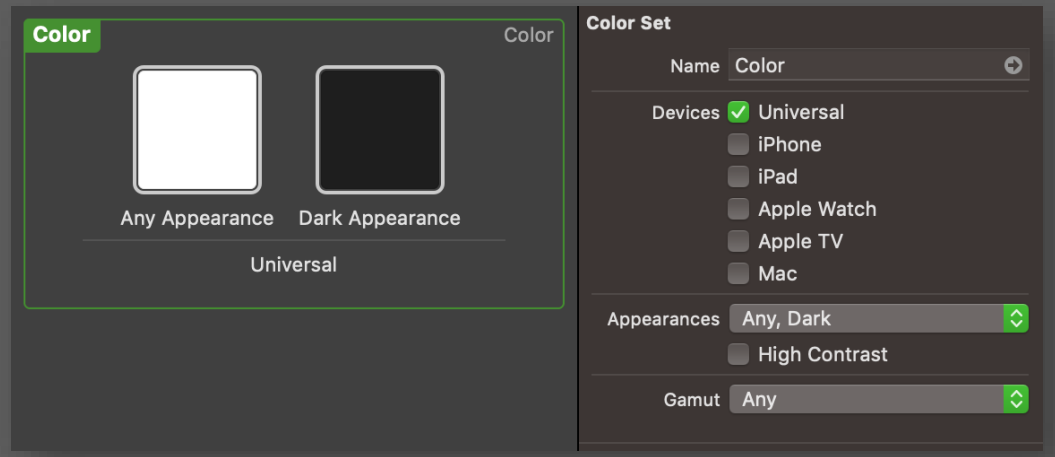
Once your app targets iOS13, it will also be influenced by the selection.

dynamic everything

- iOS adds functionality to colours and images to support dark mode
- Dynamic elements will automatically change to match the user's theme
- A new set of dynamic system colours have been added to UIColor
- Asset catalog includes support for defining theme-aware colours and images



Dynamic colours on UIColor are easily identified by the 'Color' suffix



Custom dynamic colours and images can be defined at build time in your asset catalog

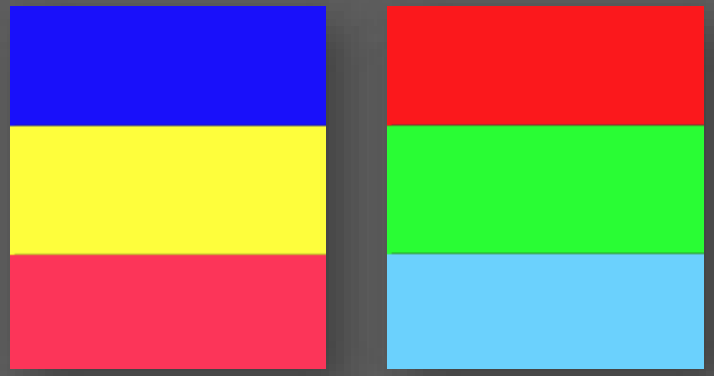
some assembly required

Homework slide

- Existing apps are likely to require some changes to support dark mode nicely
- Any use of non-dynamic colour will need to be considered
- You can programmatically define dynamic colours and images, which may simplify conversion
- You can perform arbitrary work on any ViewController in response to a theme change by overriding **TraitCollectionDidChange** and checking for the new theme
- If your app is not ready, you can target iOS13 but opt out of theming via Info.plist

demo – dark mode

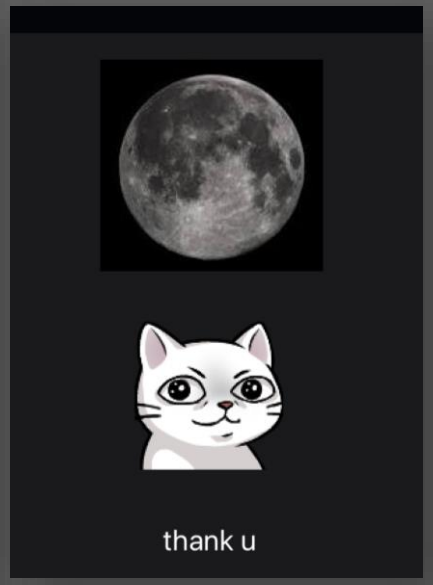
Compare light and dark system colours



```
public UIColor[] Colours
  => new []
  {
      Dynamic(UIColor.Blue, UIColor.Red),
      Dynamic(UIColor.Yellow, UIColor.Green),
      Dynamic(UIColor.SystemPinkColor, UIColor.SystemTealColor),
  };
```

```
UIColor Dynamic(UIColor light, UIColor dark)
  => new UIColor(t =>
      t.UserInterfaceStyle.HasFlag(LightFlag)
      ? light
      : dark);
```

Create theme-aware colours programmatically




```
public override void TraitCollectionDidChange(UITraitCollection prev)
{
    base.TraitCollectionDidChange(prev);

    if (TraitCollection.HasDifferentColorAppearanceComparedTo(prev))
        UpdateUIForThemeChange();
}
```

Perform arbitrary work in response to theme change

-= some of what's new in ios13 =-

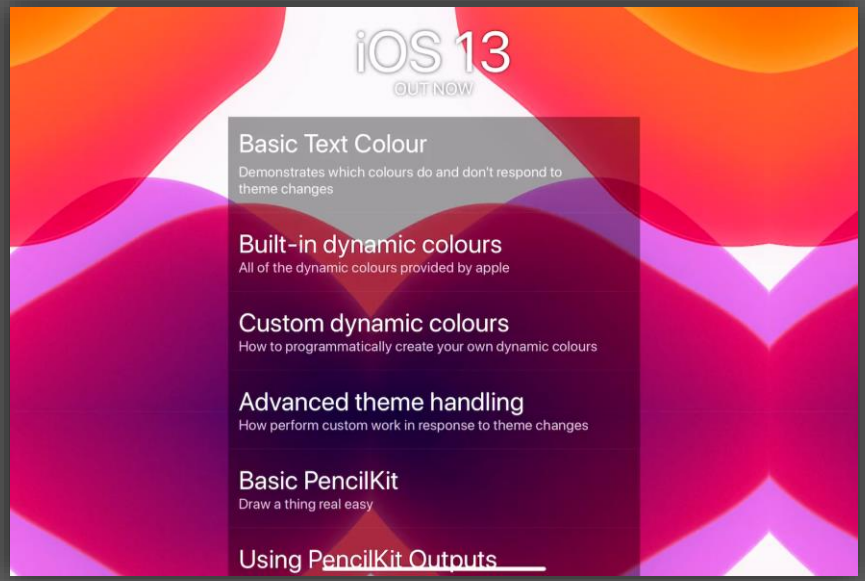
multi-window apps

your app can opt into multi-window support

Old World

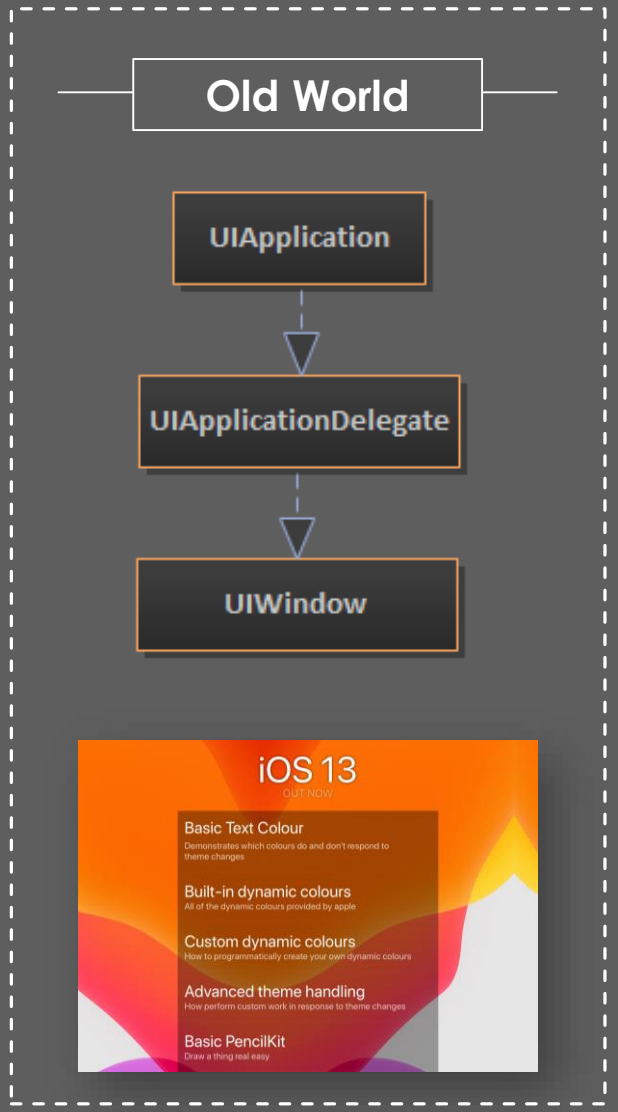
one x UIApplication
one x UIApplicationDelegate
one x UIWindow

New World



one x UIApplication
one x UIApplicationDelegate
many x UISceneDelegate
many x UISceneDelegate
many x UIWindow

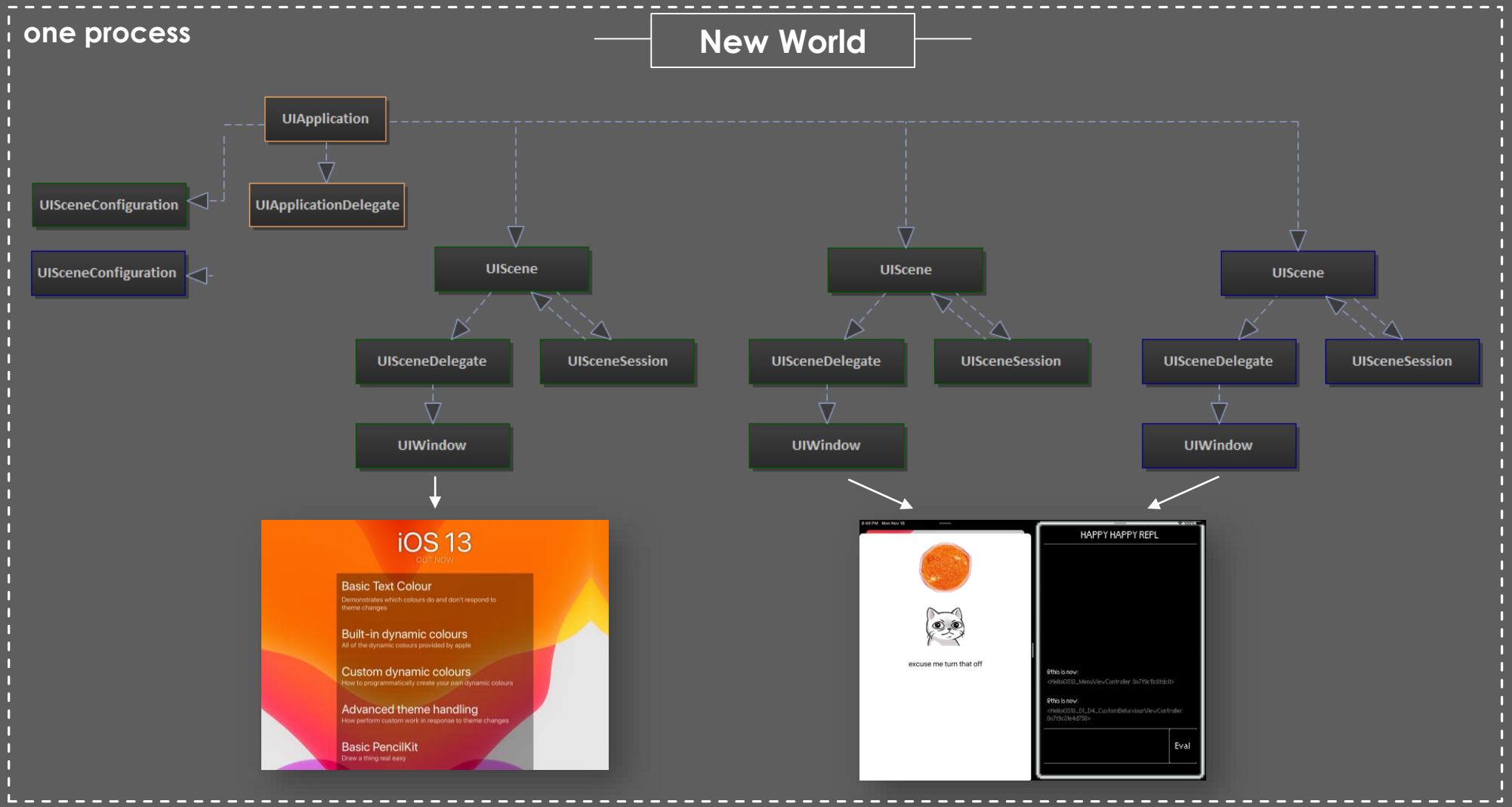
fond memories of a simpler time



one process

mo windows mo problems

Homework slide



app for process, scene for lifecycle

App/AppDelegate

- One per app
- Process-level concerns
- Initialization, DI, Xamarin.Forms.Init

Scene/SceneDelegate

- (Potentially) Many per app
- UI Lifecycle
- State Restoration

```
application:willEnterForeground
application:didEnterBackground
application:willResignActive
application:didBecomeActive
```

```
scene:willEnterForeground
scene:didEnterBackground
scene:willResignActive
scene:didBecomeActive
```

Many AppDelegate methods move to the SceneDelegate in a multi-window setup

Except on devices running iOS <13 ha ha ha

opting in is dangerously easy…

..compared to all the things you'll need to think about once it's enabled

```
<key>UIApplicationSceneManifest</key>
<dict>
    <key>UIApplicationSupportsMultipleScenes</key>
    <true/>
    <key>UISceneConfigurations</key>
    <dict>
        <key>UIWindowSceneSessionRoleApplication</key>
        <array>
            <dict>
                <key>UISceneConfigurationName</key>
                <string>Default Configuration</string>
                <key>UISceneDelegateClassName</key>
                <string>SceneDelegate</string>
            </dict>
        </array>
    </dict>
</dict>
```

Tell iOS you support multiple scenes.
This opts you in to Scene style application lifecycle

Define one or more scene configurations.
Scene configurations can be used to provide specialized behaviour for new windows, or to influence how iOS launches your app from shortcuts etc.

Just having one scene configuration is valid.

there are many ways to create a new window

From the application launcher

there are many ways to create a new window

Programmatically

```
var demo = Demos[indexPath.Row];
var userInfo = new[]
    {
       "demo",
       demo.Type.AssemblyQualifiedName
    }
    .ToNSDictionary();

var userActivity =
    new NSUserActivity("com.r2.HelloiOS13.openDetail")
    {
        UserInfo = userInfo
    };
```

Prepare an NSUserActivity describing intent
This will be provided to the UISceneDelegate to configure the window

```
UIApplication.SharedApplication.RequestSceneSessionActivation(
    sceneSession: SceneDelegate.DetailSceneSession,
    userActivity: userActivity,
    options: null,
    errorHandler: null
);
```

Request Scene Activation.
If you provide an existing scene session, it will be reused. Otherwise, a new one will be created

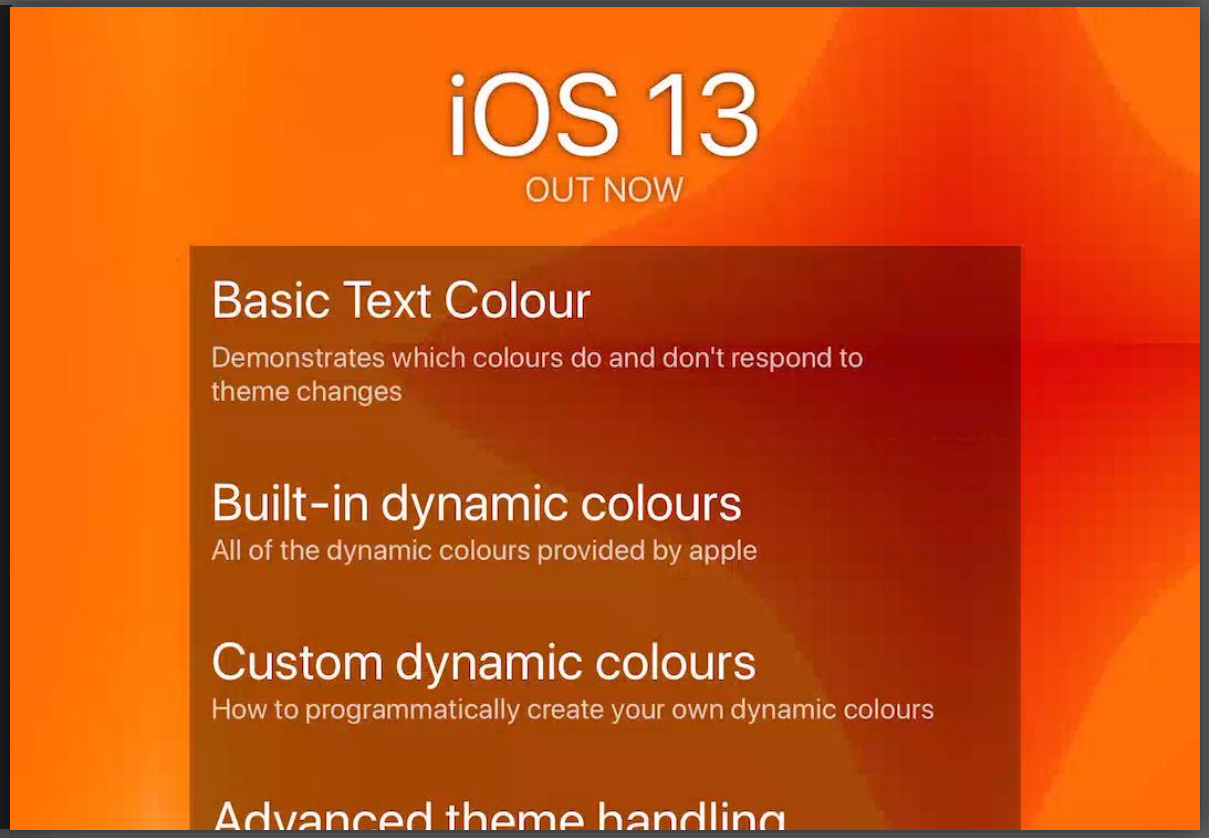
there are many ways to create a new window

Using Drag and Drop

```
MenuTableView.DragDelegate = new InlineUITableViewDragDelegate
{
    _GetItemsForBeginningDragSession = (tv, session, indexPath) =>
    {
        var demo = Demos[indexPath.Row];
        var userActivity = // same as previous slide
        var itemProvider = new NSItemProvider(userActivity);

        itemProvider.RegisterObject(
                userActivity,
                NSItemProviderRepresentationVisibility.All);

        return new[]
        {
            new UIDragItem(itemProvider)
            {
                LocalObject = userActivity
            }
        };
    },
};
```



there are many ways to create a new window

Using a Shortcut Item

```
<key>UIApplicationShortcutItems</key>
<array>
    <dict>
        <key>UIApplicationShortcutItemType</key>
        <string>com.helloios13.repl</string>
        <key>UIApplicationShortcutItemTitle</key>
        <string>REPL</string>
        <key>UIApplicationShortcutItemSubtitle</key>
        <string>Yes, a REPL!</string>
    </dict>
</array>
```

lots to consider

Homework slide

- Revisit assumptions that there is a single window in the application
- Separation of **process level concerns** and **window/scene level concerns**
  - Setting up DI, Xamarin.Forms, etc. – (probably) process level
- Multiple lifecycles and the interplay
- State restoration at the scene level
- Handling legacy versions of iOS
- Handling non-iPad devices
- Xamarin.Forms?

-= some of what's new in ios13 =-

sign in with apple

easy auth for your users (and you?????)

SIWA leverages Apple ID to improve the auth experience for developer and user.

Developer

- Free, consistent, sign-in UI
- Stable user ID across devices
- Verified email address
- Built-in 2FA
- “Real User” detection
- OIDC compliant / story for x-plat

User

- Streamlined auth/signup
- Familiar UI
- No passwords
- Email hiding
- Forced 2FA

mandatory for some apps

Homework slide

Apps that meet certain criteria must implement SIWA per App Store guidelines.

4.8 Sign in with Apple

Apps that exclusively use a third-party or social login service (such as Facebook Login, Google Sign-In, Sign in with Twitter, Sign In with LinkedIn, Login with Amazon, or WeChat Login) to set up or authenticate the user's primary account with the app must also offer Sign in with Apple as an equivalent option.

Sign in with Apple is not required if:

- Your app exclusively uses your company's own account setup and sign-in systems.
- Your app is an education, enterprise, or business app that requires the user to sign in with an existing education or enterprise account.
- Your app uses a government or industry-backed citizen identification system or electronic ID to authenticate users.
- Your app is a client for a specific third-party service and users are required to sign in to their mail, social media, or other third-party account directly to access their content.

New apps now(?), existing ones after April 2020.

demo - sign in with apple

Add SIWA to your App ID in the developer portal

```
private void SignInWithAppl()
{
    var provider = new ASAuthorizationAppleIdProvider();
    var request = provider.CreateRequest();

    request.RequestedScopes = new[]
    {
        ASAuthorizationScope.Email,
        ASAuthorizationScope.FullName
    };

    var authorizationController =
        new ASAuthorizationController(new[] { request })
        {
            Delegate = this,
            PresentationContextProvider = this
        };

    authorizationController.PerformRequests();
}
```

Configure an ASAuthorizationController and call PerformRequests() to have iOS present an appropriate interface to the user.

```
public void DidComplete(ASAuthorizationController _, ASAuthorization auth)
{
    var cred = auth.GetCredential<ASAuthorizationAppleIdCredential>();

    var userIdentifier = cred.User;
    var fullName = cred.FullName;
    var email = cred.Email;
    var identityToken = cred.IdentityToken;
    var authCode = cred.AuthorizationCode;

    // send details to the server
}
```

On first auth, iOS will give you name and email if requested. Subsequent authentications will not include these details.

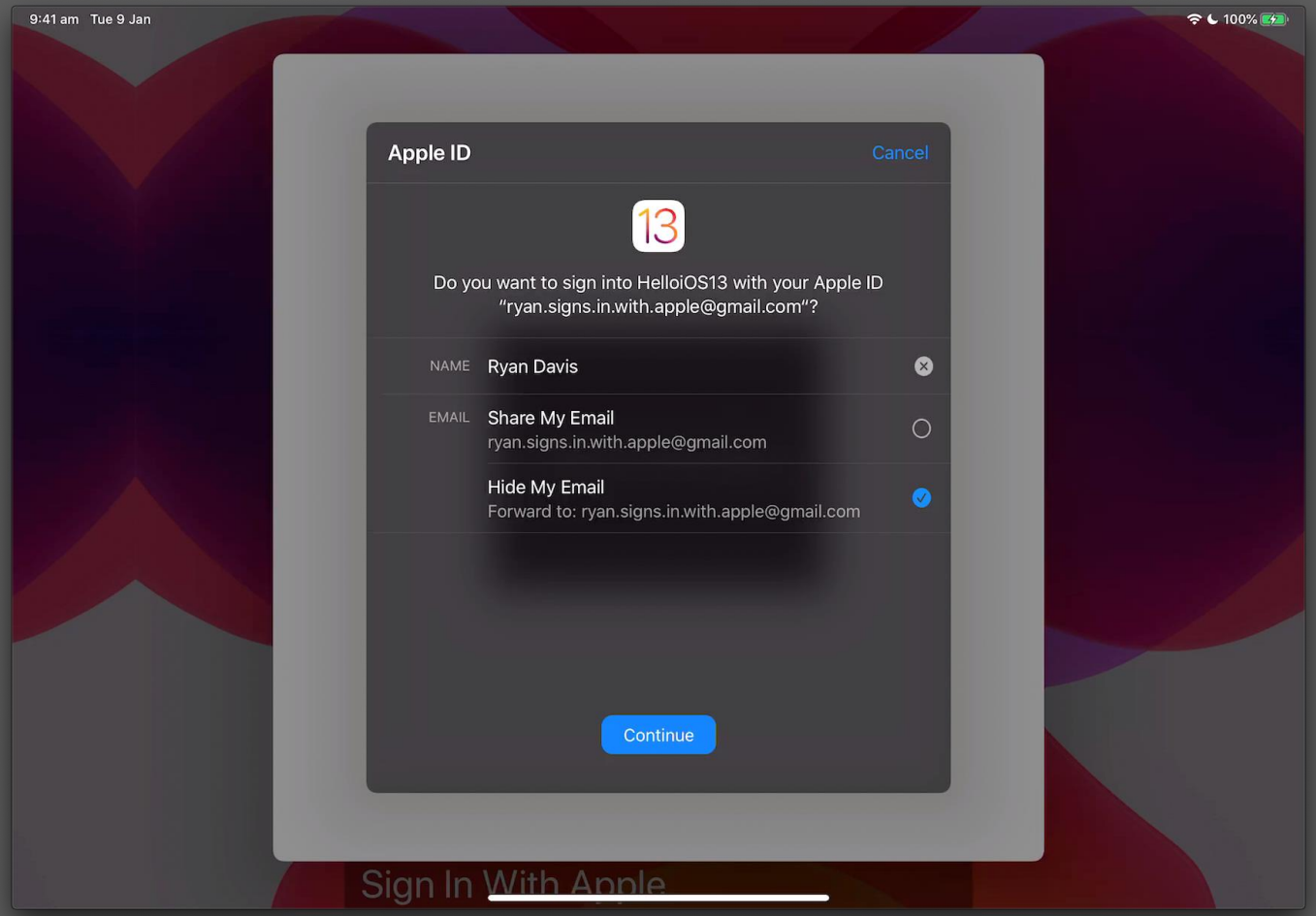
```
var provider = new ASAuthorizationAppleIdProvider();

provider.GetCredentialState(
    "theUserId", (credentialState, error) => { /* .. */ });
```

```
Authorized
NotFound
Revoked
Transferred
```

Check the state of credential at startup and present UI appropriately.

demo - sign in with apple



-= some of what's new in ios13 =-

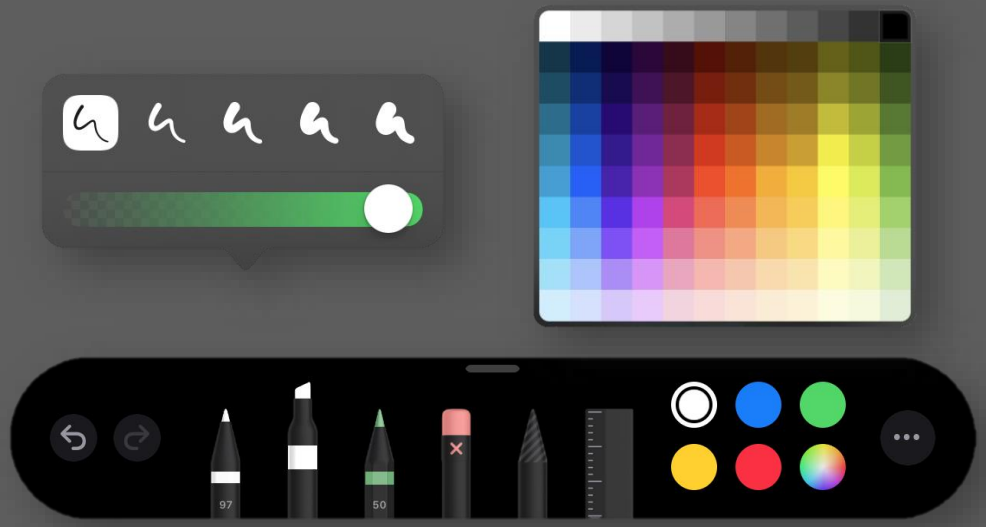
pencilkit

precise, performant pencil input

or finger!

PencilKit is a high performance input framework that lets you provide users with a sophisticated drawing environment with just a few lines of code.

Optimised for
(but does not require)
Apple Pencil



Includes familiar tooling UI with undo/redo,
vector-based select/erase

Transparently handles light
and dark mode themes

two classes, many superpowers

PKCanvasView

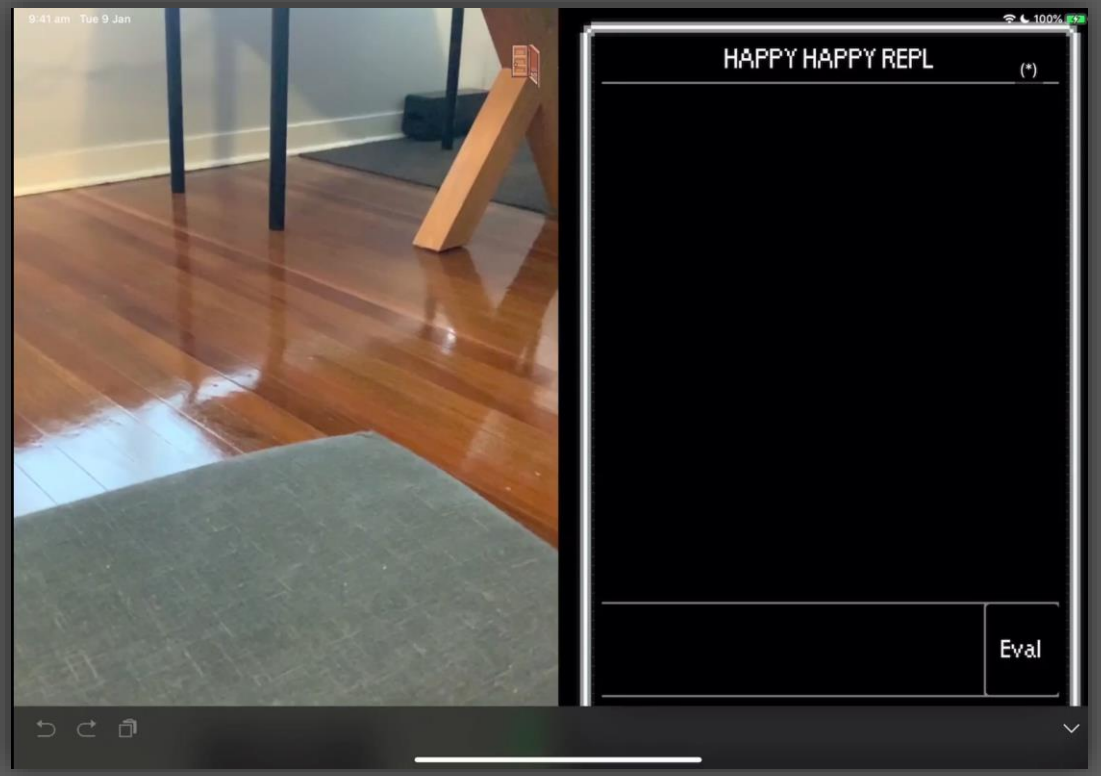
- Accepts input from user's finger or Pencil
- Provides change callbacks via a delegate
- Supports bitmap and vector interactions
- Recolours content on theme change

PKToolPicker

- Floating/dockable toolbox
- Colours, tools, lasso selection, undo/redo
- Consistent UI with built in iOS applications

demo – pencilkit

Use PencilKit to create a
hand drawn pattern background



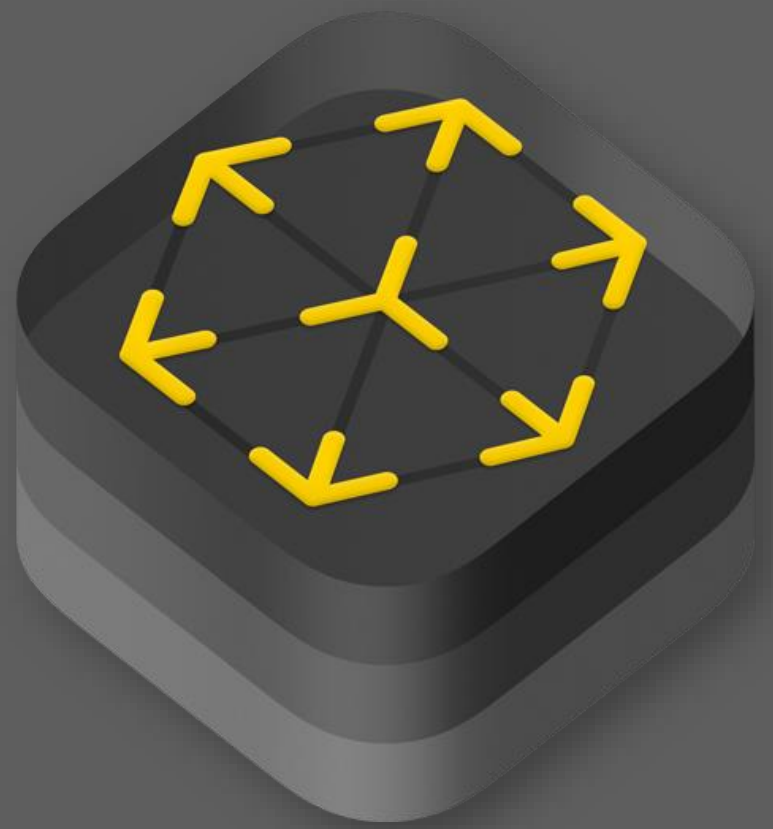
Use PencilKit to texture
virtual content in ARKit

-= some of what's new in ios13 =-

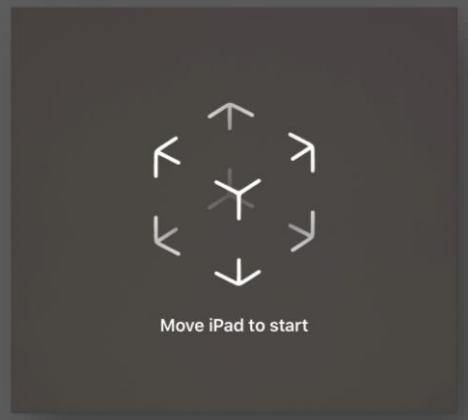
arkit 3

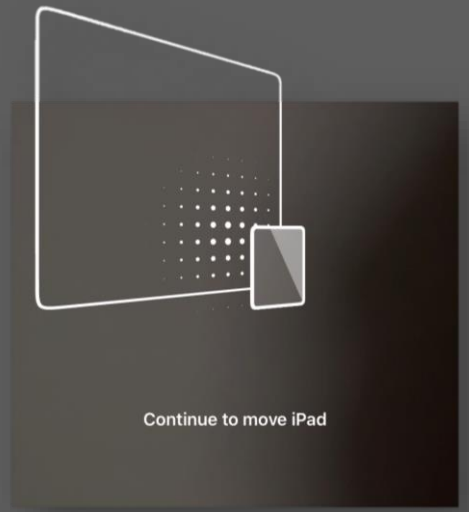
augmenting apple's augmented offering

- **Performance improvements** driven primarily by advances in apple's ML capability
- **Features to improve end-user experience** such as automated coaching and additions to collaborative experiences
- **New capabilities** like people occlusion, body motion tracking and dual camera AR



demo – automated coaching






```
Coach = new ARCoachingOverlayView
{
    ActivatesAutomatically = true,
    Delegate = new ARDelegate(this),
    Session = SCNView.Session
};
```

```
Coach.Goal =
    ARCoachingGoal.
                  AnyPlane
                  HorizontalPlane
                  Tracking
                  VerticalPlane
```

Set a goal and ARCoachingOverlayView automatically guides the user when it is not currently satisfied

```
public class CoachingDelegate : ARCoachingOverlayViewDelegate
{
    readonly ARCoachingViewController Parent;

    public CoachingDelegate(ARCoachingViewController parent)
        => Parent = parent;

    public override void WillActivate(ARCoachingOverlayView _)
        => Parent.DimUserInterface(true);

    public override void DidDeactivate(ARCoachingOverlayView _)
        => Parent.DimUserInterface(false);

    public override void DidRequestSessionReset(ARCoachingOverlayView _)
        => Parent.ResetTracking();
}
```

Implement ARCoachingOverlayViewDelegate to be informed of coaching lifecycle events

demo – people occlusion

```
public override ARConfiguration GetARConfiguration()
    => new ARWorldTrackingConfiguration
    {
        FrameSemantics =
            ARFrameSemantics.
    };
```

```
BodyDetection
None
PersonSegmentation
PersonSegmentationWithDepth
```

Enable segmentation by setting the appropriate flag on the new FrameSemantics property

```
public override void OnFrameUpdate(ARSession s, ARFrame f)
{
    base.OnFrameUpdate(s, f);

    var depth = f.EstimatedDepthData;
    var seg = f.SegmentationBuffer;

    // do phd things here
}
```

If you are smarter than me you can make use of the depth and segmentation data that ARKit detects

So realistic I could put my hand through it

demo – multi camera tracking


```
public override ARConfiguration GetARConfiguration()
    => new ARWorldTrackingConfiguration
    {
        PlaneDetection = ARPlaneDetection.Horizontal,
        UserFaceTrackingEnabled = true
    };
```

Rather than use ARFaceTrackingConfiguration, set the new UserFaceTrackingEnabled flag on ARWorldTrackingConfiguration to enable multi-camera tracking

```
public override void OnNodeAddedForAnchor(
    ISCNSceneRenderer renderer, SCNNode node, ARAnchor anchor)
{
    switch (anchor)
    {
        case ARFaceAnchor faceAnchor:
            // do things with face
            break;

        case ARPlaneAnchor planeAnchor:
            // do things with plane
            break;
    }
}
```

In OnNodeAdded/Updated/Removed respond appropriately based on the type of anchor detected

-= some of what's new in ios13 =-

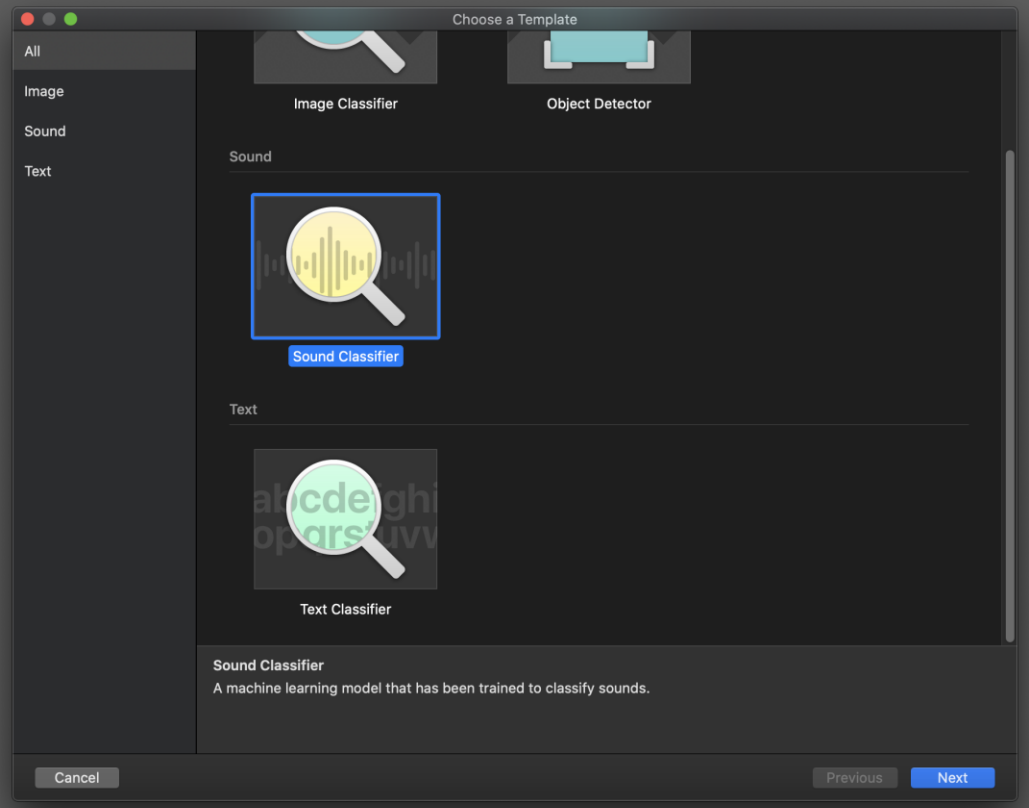
coreml 3

too many updates to mention

- New model types supported, kNN, ItemSimilarityRecommender, SoundAnalysisPreprocessor, LinkedModel
- Many new NN layer types (increased expression = greater external compatibility), including control flow layers
- Mutable models – on-device retraining (for a subset of model types)
- Improvements to CreateML – new tasks like Audio Classification
- Improvements to Turi Create

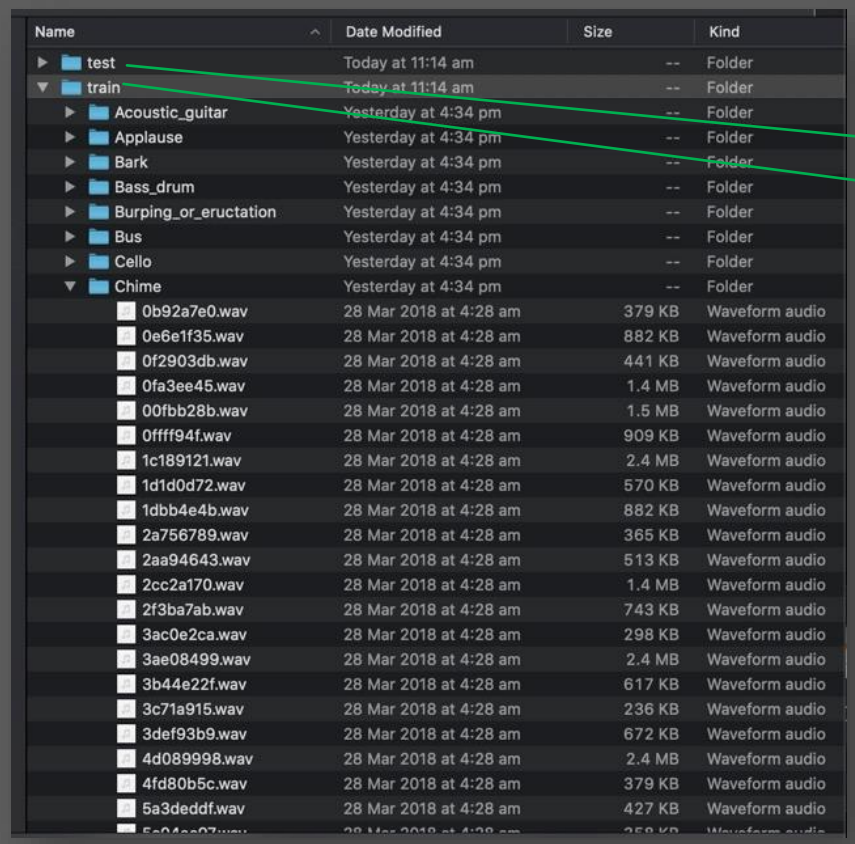
demo – SoundAnalysis with CreateML

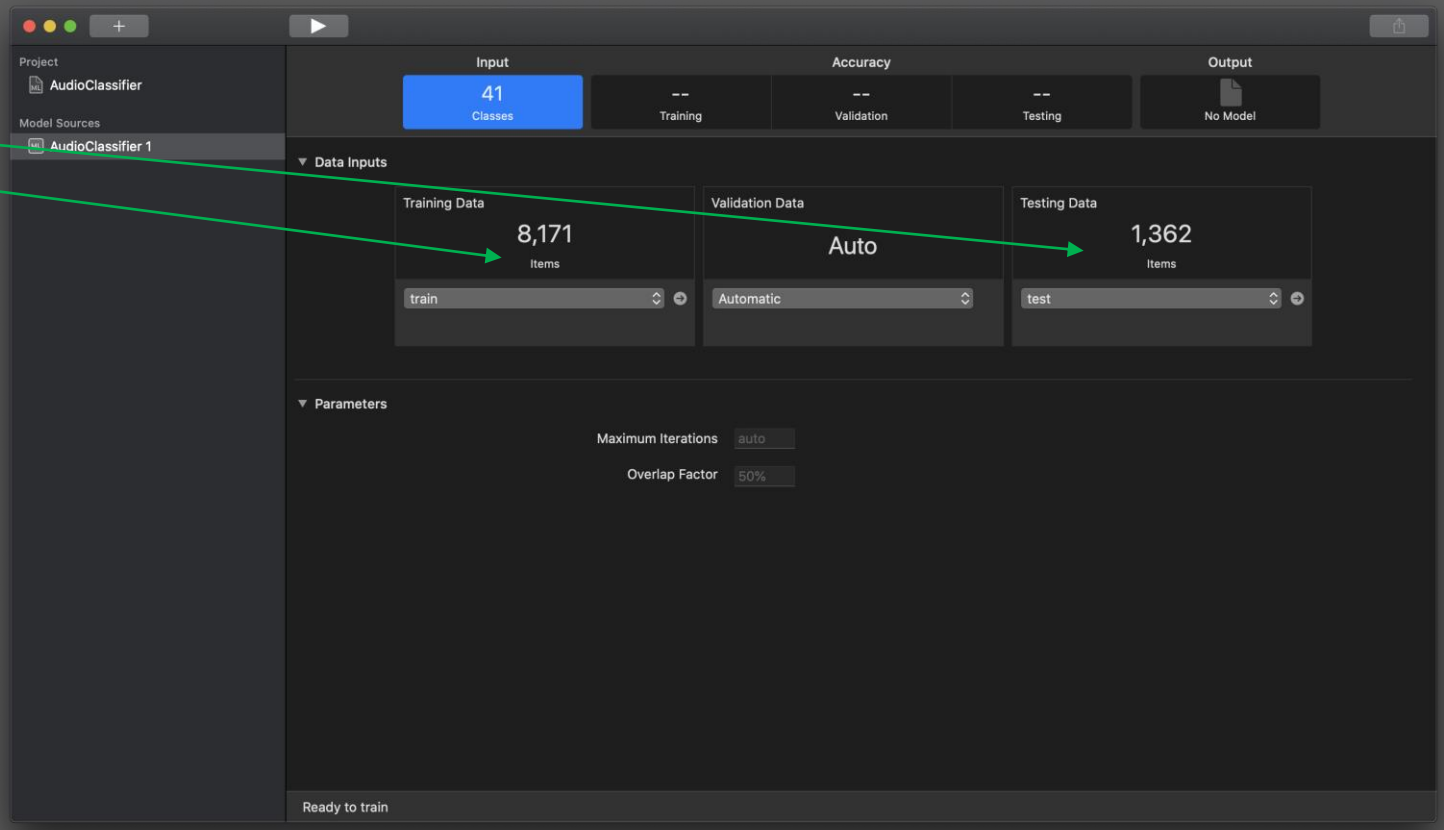
- CreateML has a new Sound Classifier template



demo – SoundAnalysis with CreateML

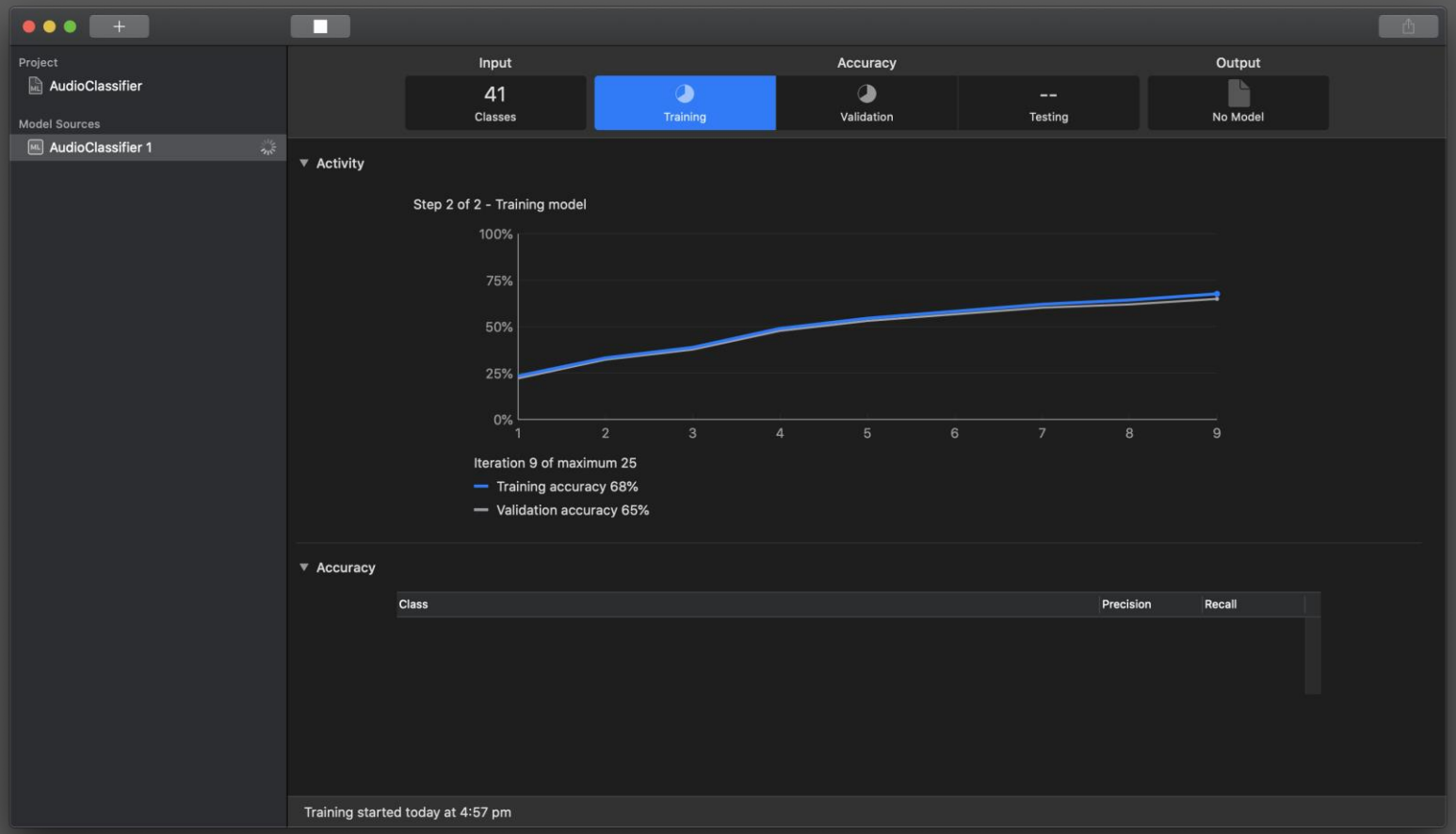
- Wizard flow is the same as for existing templates





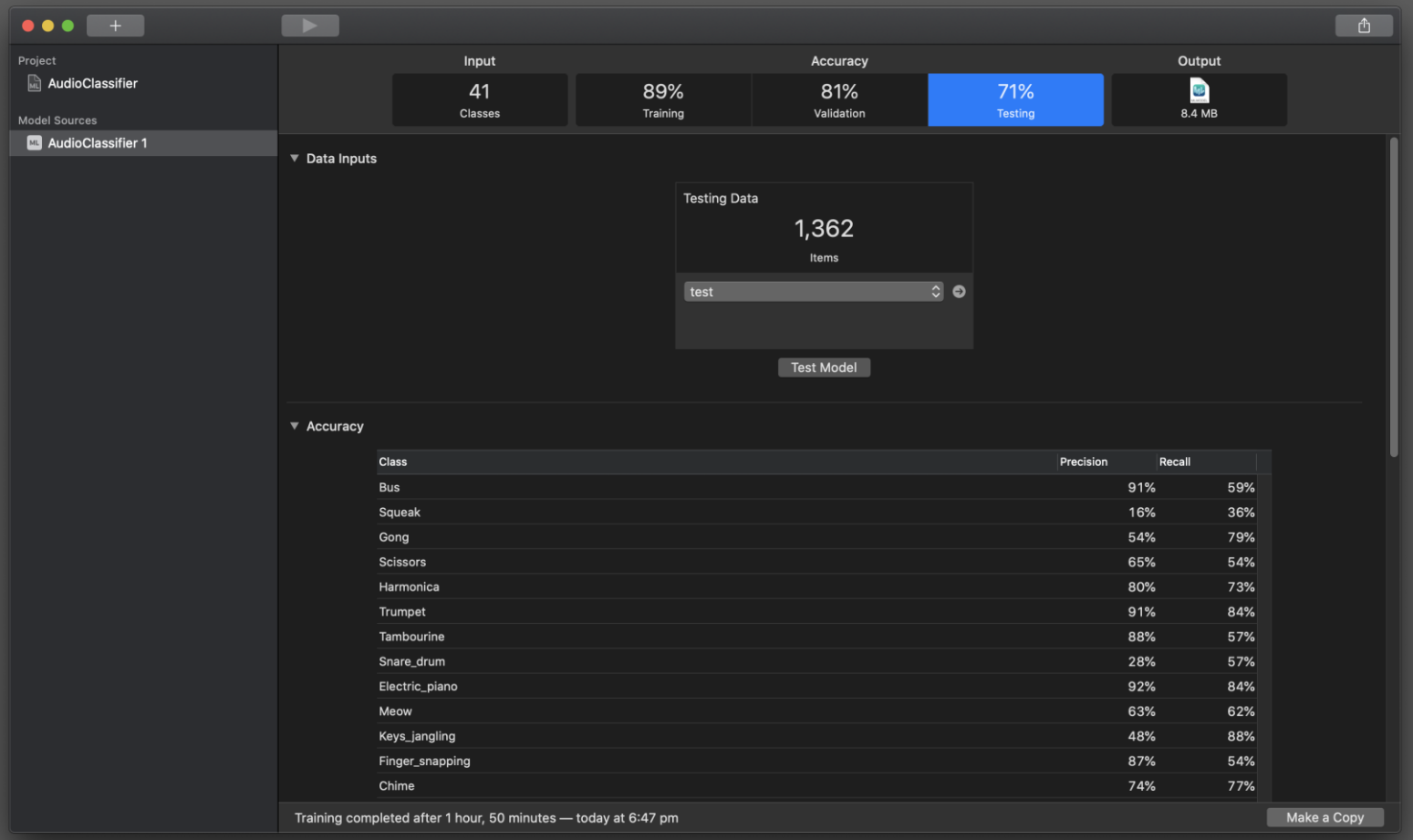
demo – SoundAnalysis with CreateML

- Training time depends on the number of examples and number of iterations



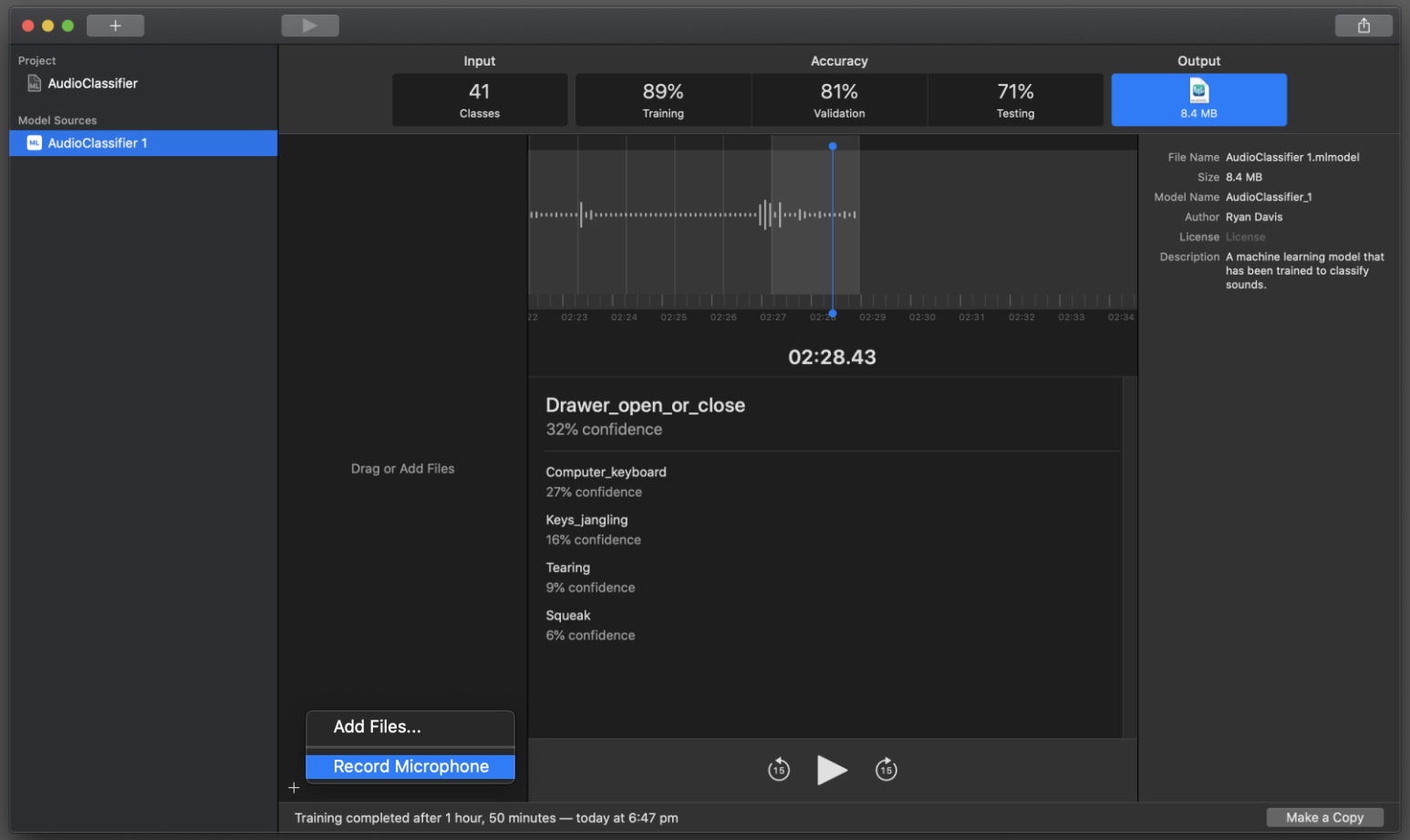
demo – SoundAnalysis with CreateML

- CreateML automatically evaluates the model and presents results

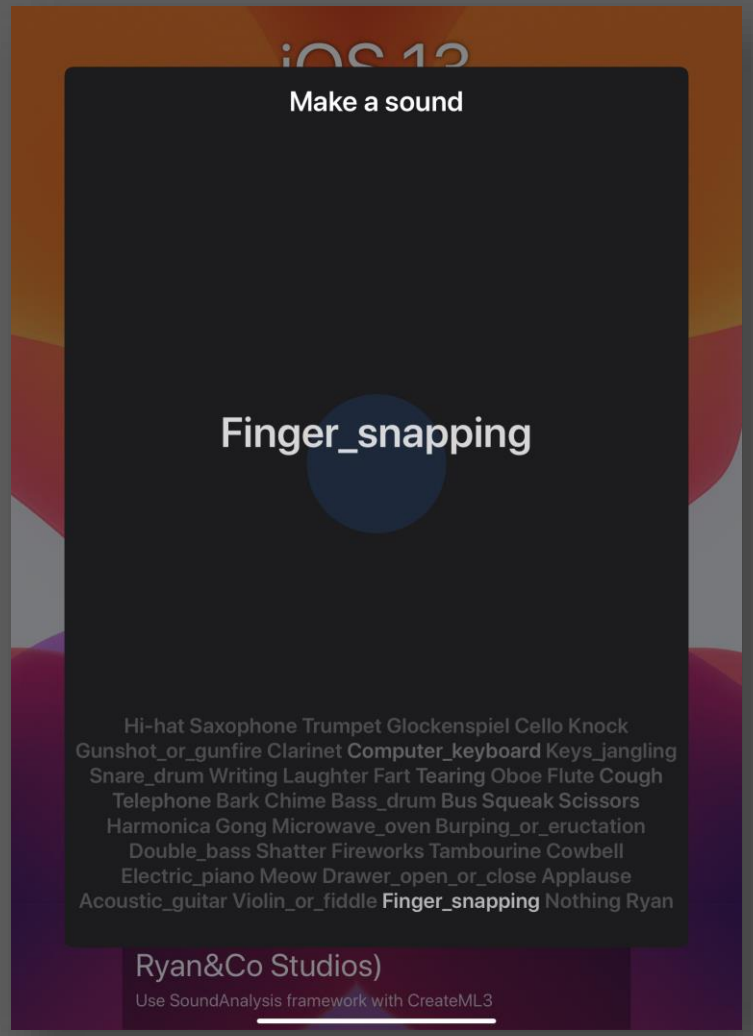


demo – SoundAnalysis with CreateML

- CreateML allows you to test the model using samples or your microphone



demo – SoundAnalysis in app



-= some of what's new in ios13 =-

corenfc

a substantial expansion to the NFC offering

- iOS11 – NDEF tag reading
- iOS12 – background NDEF tag scanning
- iOS13 – Lots of good stuff:

NDEF Writing

- New **NFCNDEFReaderSession** helper
- Query NDEF status (capacity, kind)
- Read/Write NDEF message
- Lock an NDEF tag
- **Your app can create Siri Shortcut compatible tags**

Native Tag Access

- New **NFCTagReaderSession**
- Support for ISO14443, ISO15693, ISO18092
  - Electronic IDs, transit cards
  - Payment cards deliberately excluded
- Unique Identifier (UID) access
- Amiibo

Demo – MiFare native protocol interaction

```
public void BeginSession()
{
    TryInvalidateSession();

    Session = new NFCTagReaderSession(
        NFCPollingOption.Iso14443,
        GetSessionDelegate(),
        DispatchQueue.MainQueue);

    Session.BeginSession();
}
```

```
public NFCTagReaderSessionDelegate GetDelegate() =>
    new InlineNFCTagReaderSessionDelegate
    {
        _DidBecomeActive = (session)
            => session.RestartPolling(),

        _DidDetectTags = (session, tags) =>
        {
            // process the tag here
        },

        _DidInvalidate = (session, error) =>
        {
            // handle an error here
        }
    };
```

Use an NFCTagReaderSession to scan for ISO14443 tags, then process them via the delegate

```
public async Task ReadTagData(
    NFCTagReaderSession session, INFCTag tag)
{
    await session.ConnectToAsync(tag);

    var mifare = tag.GetNFC
                    GetNFCFeliCaTag
                    GetNFCIso15693Tag
                    GetNFCIso7816Tag
                    GetNFCMiFareTag
```

INFCTag exposes methods to get 'native tag' representations

```
// send commands
await mifare.SendMifareCommand(
    0x3a,
    0x0, 0x86);
```

Native tag representations facilitate the sending of native protocol commands

10.3 FAST_READ

The FAST_READ command requires a start page address and an end page address and returns the all n*4 bytes of the addressed pages. For example if the start address is 03h and the end address is 07h then pages 03h, 04h, 05h, 06h and 07h are returned. If the addressed page is outside of accessible area, the MF0ULx1 replies a NAK. For details on those cases and the command structure, refer to Figure 14 and Table 18.

Table 19 shows the required timing.

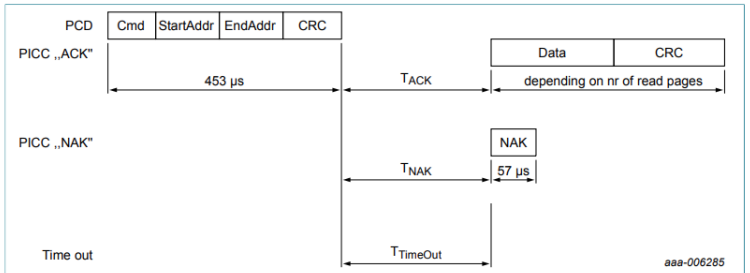


Figure 14. FAST_READ command

Table 18. FAST_READ command

Name	Code	Description	Length
Cmd	3Ah	read multiple pages	1 byte
StartAddr	-	start page address	1 byte
EndAddr	-	end page address	1 byte
CRC	-	CRC according to Ref. 1	2 bytes
Data	-	data content of the addressed pages	n*4 bytes
NAK	see Table 10	see Section 9.3	4-bit

CoreNFC handles a lot of the busywork, but you'll still probably need to consult the datasheet

Demo – MiFare native tag access

```
<key>com.apple.developer.nfc.readersession.formats</key>
<array>
    <string>NDEF</string>
    <string>TAG</string>
</array>
```

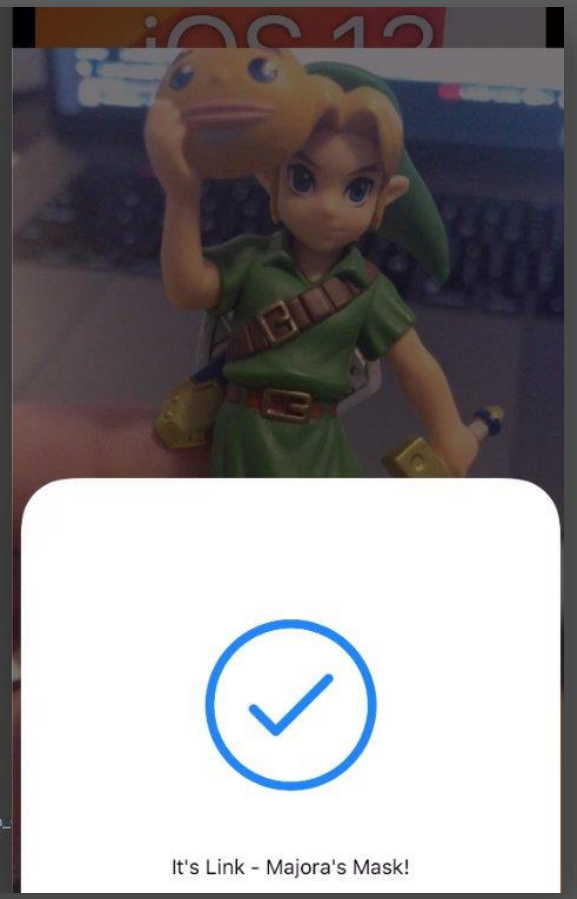
Don't forget to add the TAG format to Entitlements.plist

(and don't tell VS about it)

```
Thread finished: <Thread Pool> #29
Thread started: <Thread Pool> #33
Thread started:  #34
== Raw tag data ==
00000000   04 23 BF 10 0A BA 4C 81  7D 48 0F E0 F1 10 FF EE   ·#¿··ºL}H·àñ·ÿî
00000010   A5 00 01 00 88 8C A1 33  4E BB C8 D7 67 F7 20 28   ¥···¡3N»È×g÷ (
00000020   0D 37 5D 31 EE DC 5F 25  62 32 A0 8E FB BD 3A 3D   ·7]1îÜ_%b2 Ûº:=
00000030   7D 85 5A 10 38 5C 9F F4  D7 23 5E C5 16 A8 72 F0   }…Z·8\ô×#^Å·¨rð
00000040   6D 2E 68 89 BF E7 6E CE  90 BB CA 39 65 57 61 09   m.h‰¿çnÎ»Ê9eWa·
00000050   75 20 42 4F 01 00 00 00  03 4C 09 02 0D 12 47 13   u BO·····L····G·
00000060   93 9E 00 3D A6 C3 35 18  8B 70 5F 56 0D E6 11 F1   ·=¦Ã5·p_V·æ·ñ
00000070   5E 26 7C 8B 6C 81 B5 34  49 3D D0 69 20 D3 3C 61   ^&|‹lµ4I=Ði Ó<a
00000080   33 9E 41 42 55 D9 29 43  F8 7C 0C D7 05 69 4D 76   3ABUÙ)Cø|·×·iMv
00000090   AE B0 CE 5B 93 98 FB A4  73 EA 3E D4 77 32 A6 A2   ®°Î[“˜û¤sê>Ôw2¦¢
000000A0   C9 05 0D F5 E7 67 77 A0  18 B6 1C 0E B3 F6 6D 91   É··õçgw ·¶··³öm‘
000000B0   FD 1D CC B9 1D 78 3A AF  69 43 E4 51 BC 9B CE DE   ý·Ì¹·x:¯iCäQ¼›ÎÞ
000000C0   2A EB 24 7F B1 7D BB A6  80 27 4C E7 34 F0 1D 03   *ë$±}»¦'Lç4ð··
000000D0   ED 73 9E 42 B0 BC F4 60  26 3D E7 C0 E9 3F FE 22   ísB°¼ô`&=çÀé?þ"
000000E0   67 FE 40 5C 7D 87 AF C4  8D 6B 84 80 6A 0D BA E2   gþ@\}‡¯Äkj·ºâ
000000F0   08 98 50 1F B2 0F 96 50  BE 6A 71 C3 4F EE A7 B8   ·P·²·P¾jqÃOî§¸
00000100   05 5A FE 81 43 56 A2 6B  48 38 8C 64 80 49 0A 02   ·Zþ‹CV¢kH8dI··
00000110   5A 7C 6E 2B 12 0F 74 8F  C1 E7 ED D4 61 03 BA 0C   Z|n+··tÁçíÔa·º·
00000120   B3 6F F1 3D 41 D3 D7 E4  EF 32 D7 51 A3 70 00 21   ³oñ=AÓ×äï2×Q£p·!
00000130   9D A7 58 1E BC 21 C1 F9  1D A2 10 1C 9A 5D 8C F7   §X·¼!Áù·¢··]+
00000140   D7 14 E2 39 9A 25 F5 9B  40 DE DD 18 36 43 61 2A   ×·â9š%õ›@ÞÝ·6Ca*
00000150   45 91 9B 8C 56 BA D0 95  2C 81 21 7C B0 99 4B AE   E‘›ŒVºÐ•,‹!|°™K®
00000160   73 5E E0 66 71 9B BE 27  44 65 FA 0E F6 23 5E 7A   s^àfq›¾'Deú·ö#^z
00000170   C1 E8 E3 7F 53 5D C1 B7  19 1C FD BB 09 D5 5A C4   Áèã S]Á···ý»·ÕZÄ
00000180   3B 06 82 84 AB 7E 00 7B  55 DA 36 57 13 8E F4 C4   ;·‚„«~·{UÚ6W·ôÄ
00000190   FC B0 ED C6 2B 6F C1 AB  43 DC 68 6D D2 F7 39 49   ü°íÆ+oÁ«CÜhmÒ÷9I
000001A0   27 77 F5 08 06 80 E1 5E  60 F4 70 02 65 40 8C 55   'wõ··€á^`ôp·e@ŒU
000001B0   E8 52 BD AA C3 15 93 E5  72 68 46 0D CF EC 1A EF   èR½ªÃ·“århF·Ïì·ï
000001C0   B5 B4 BA F8 DF D4 C8 02  4B 3D CE E5 9E C6 31 80   µ´ºøßÔÈ·K=ÎåžÆ1€
000001D0   44 1D 3C C6 D9 56 64 CC  FD 84 87 7E 54 0C C9 D3   D·<ÆÙVdÌý„‡~T·ÉÓ
000001E0   F3 E7 1E CB 95 91 02 7A  2C 05 F4 C6 4C CC D7 A1   óç·Ë•‘·z,·ôÆLÌ×¡
000001F0   AB 9D 01 F7 36 B2 0C FE  F7 59 FD CE 88 64 3B 75   «··÷6²·þ÷YýÎˆd;u
00000200   C7 B5 12 1D 31 55 97 60  01 00 0F BD 00 00 00 04   Çµ··1U—`···½····
00000210   5F 00 00 00                                       _···

== Identifiers ==
Nfc Uid: 0423BF0AB
Game Id: 01000000
Char Id: 034C0902

Thread started: <Thread Pool> #35
Thread started: <Thread Pool> #36
Thread started: <Thread Pool> #37
== Metadata : https://www.amiiboapi.com/api/ ==
{
  "Name": "Link - Majora's Mask",
  "Character": "Link",
  "GameSeries": "The Legend of Zelda",
  "AmiiboSeries": "Legend Of Zelda",
  "Image": "https://raw.githubusercontent.com/N3evin/AmiiboAPI/master/images/icon_
  "Head": "01000000",
  "Tail": "034c0902",
  "Type": "Figure"
}
```



-= some of what's new in ios13 =-

wrapping up

what we saw

New | Improved

Dark mode	RealityKit	CryptoKit	PencilKit	CoreNFC	SwiftUI
ARKit3	SFSymbols	CoreHaptics	CoreML3	Metal	Sound Analysis
Sign in With Apple	Background Tasks	Metric Kit	VisionKit	SiriKit	Multi window

useful resources

- **iOS13 on Apple Developer**
  https://developer.apple.com/ios/
- **WWDC2019 Videos**
  https://developer.apple.com/videos/wwdc2019/
- **Xamarin iOS13 Documentation**
  https://docs.microsoft.com/en-us/xamarin/ios/platform/ios13/
- **Merge Conflict Podcast – WWDC Roundup**
  https://www.mergeconflict.fm/special-edition-wwdc19
- **(Some of) What's new in iOS13**
  https://ryandavis.io/some-of-whats-new-in-ios13/
- **(More of) What's new in iOS13**
  ???

questions