#WWDC19

What's New in Authentication

Easier and more secure sign-in

Ricky Mondello, iOS Engineer

Sign In with Apple

Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS

Sign In with Apple

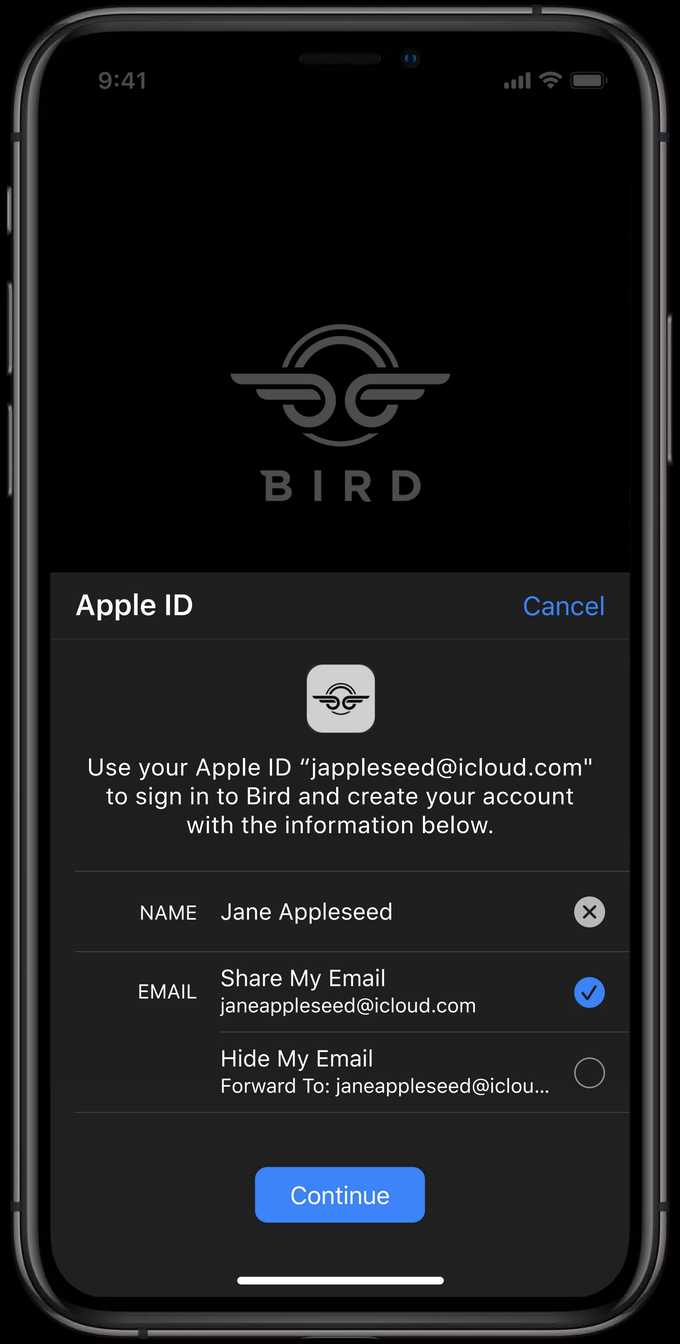
Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS


9:41
BIRD
Sign in with Apple
- or -
SIGN IN WITH EMAIL

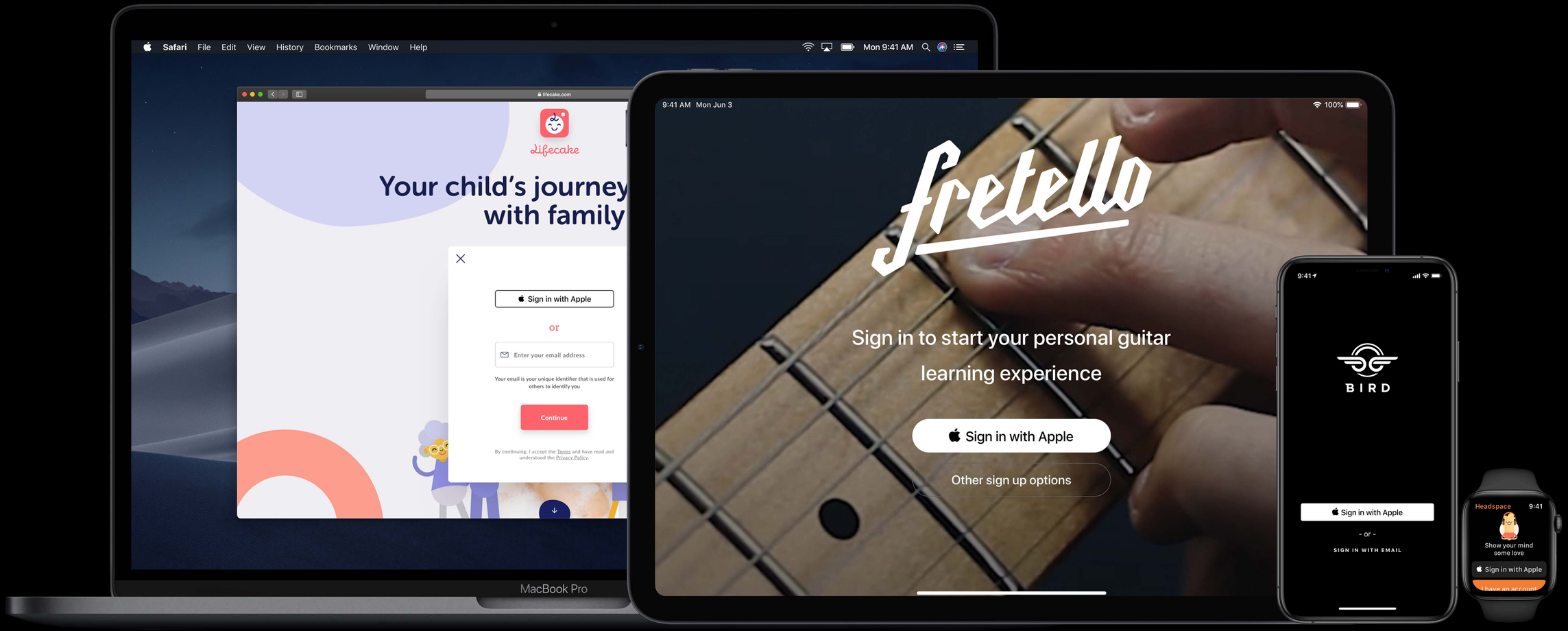

9:41
BIRD
Apple ID
Cancel
Use your Apple ID “jappleseed@icloud.com" to sign in to Bird and create your account with the information below.
NAME
Jane Appleseed
EMAIL
Share My Email
janeappleseed@icloud.com
Hide My Email
Forward To: janeappleseed@iclou...
Continue

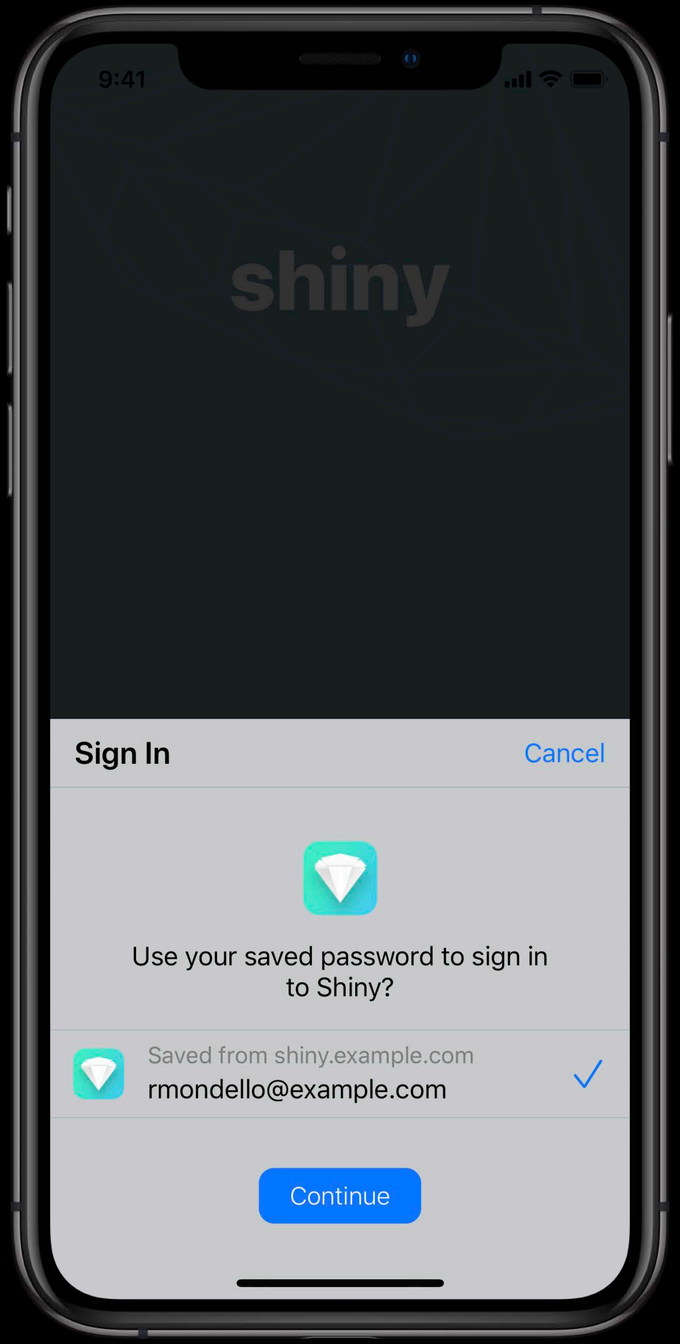

9:41
NEARBY VEHICLES
HOW TO RIDE
Rookies
Nox Cookie Bar
San Jose
Museum of Art
Fairmont
MUJI
Bijan Bakery
Four Points
Park Ave
The Tech Museum
of Innovation
Arcadia
Hyatt Place
Hilton
The UPS Store
RIDE


Sign in with Apple

Easier than passwords

Sign In with Apple accounts are secure, verified accounts


Safari File Edit View History Bookmarks Window Help
Mon 9:41 AM
lifecake.com
Lifecake
Your child's journey with family
Sign in with Apple
or
Enter your email address
Your email is your unique identifier that is used for others to identify you
Continue
By continuing, I accept the Terms and have read and understood the Privacy Policy.
MacBook Pro
9:41 AM Mon Jun 3
100%
fretello
Sign in to start your personal guitar learning experience
Sign in with Apple
Other sign up options
9:41
BIRD
Sign in with Apple
- or -
SIGN IN WITH EMAIL
Headspace
9:41
Show your mind some love
Sign in with Apple
I have an account


9:41
shiny
Sign In
Cancel
Use your saved password to sign in to Shiny?
Saved from shiny.example.com
rmondello@example.com
Continue


Sign in with Apple


Sign in with Apple

Sign In with Apple

Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS

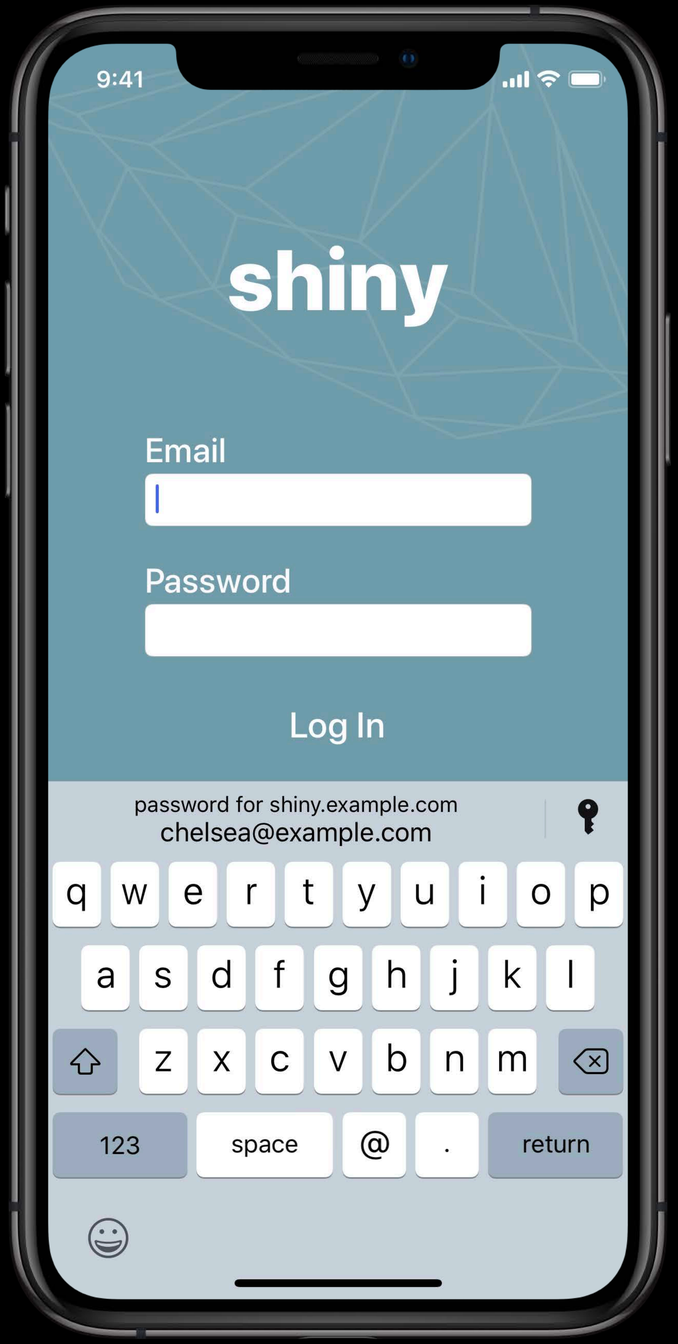
Sign In with Apple

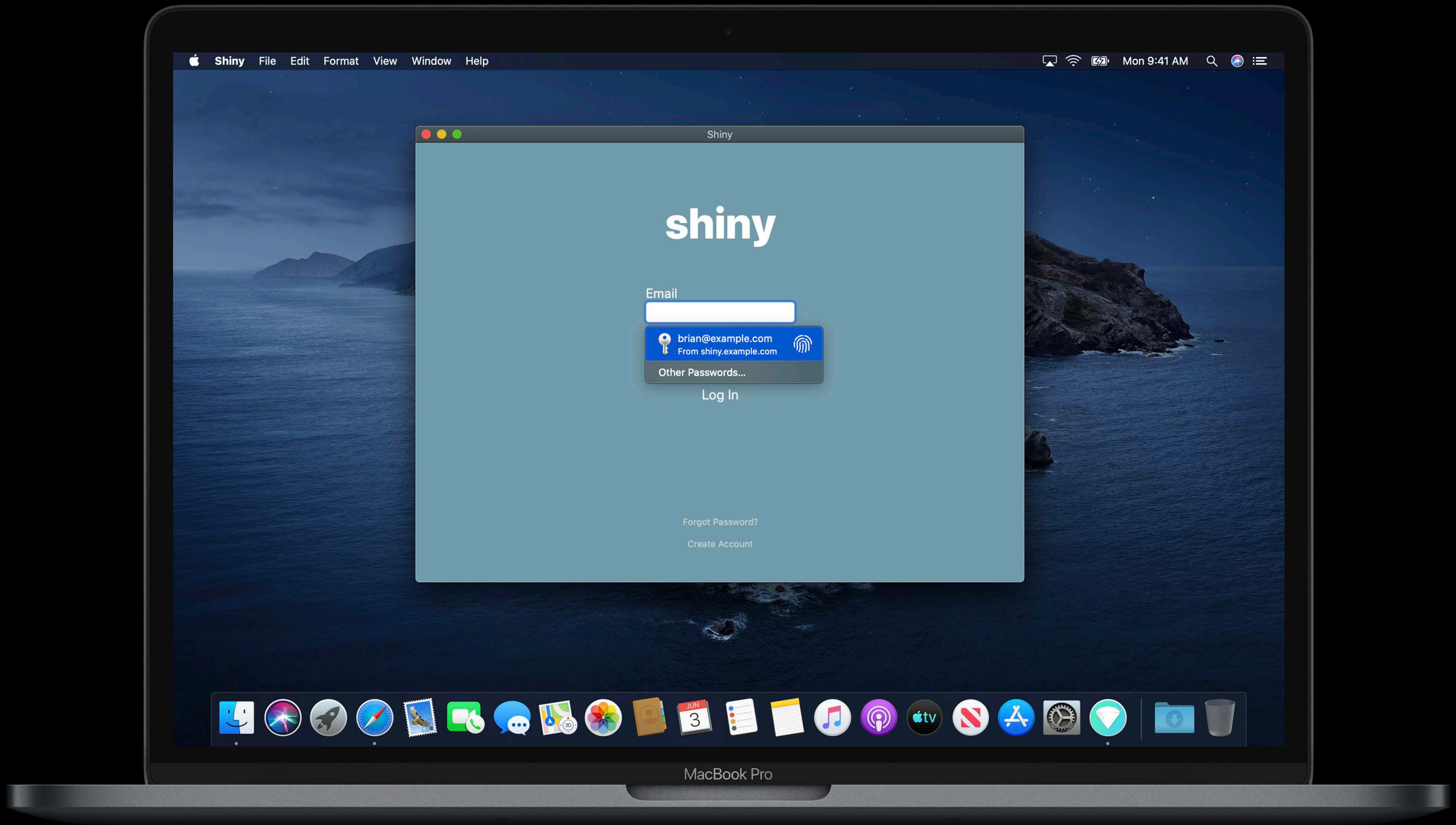
Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS


9:41
shiny
Email
Password
Log In
password for shiny.example.com
chelsea@example.com
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
123
space
@
.
return


Shiny File Edit Format View Window Help
Mon 9:41 AM
Shiny
shiny
Email
brian@example.com
From shiny.example.com
Other Passwords...
Log In
Forgot Password?
Create Account
MacBook Pro

```
// https://example.com/.well-known/apple-app-site-association

{
    "webcredentials": {
        "apps": [ "A1B2C3D4E5.com.example.Shiny" ]
    }
}
```

```
// https://example.com/.well-known/apple-app-site-association

{
    "webcredentials": {
        "apps": [ "A1B2C3D4E5.com.example.Shiny",
                  "A1B2C3D4E5.uikitformac.com.example.Shiny" ]
    }
}
```

```
// https://example.com/.well-known/apple-app-site-association

{
    "webcredentials": {
        "apps": [ "A1B2C3D4E5.com.example.Shiny",
                  "A1B2C3D4E5.uikitformac.com.example.Shiny" ]
    }
}
```

```
// https://example.com/.well-known/apple-app-site-association
// Universal Links


{
    "applinks": {
        "apps": [],
        "details": [
            {
                "appID": "A1B2C3D4E5.com.example.Shiny",
                "components": [ { "/": "/a/path/*" } ]
            }
        ]
    }
}
```

```
// https://example.com/.well-known/apple-app-site-association
// Universal Links
// Handling multiple apps (iOS 13, macOS 10.15)

{
    "applinks": {
        "apps": [],
        "details": [
            {
                "appID": "A1B2C3D4E5.com.example.Shiny",
                "appIDs": [ "A1B2C3D4E5.com.example.Shiny",
                            "A1B2C3D4E5.uikitformac.com.example.Shiny" ],
                "components": [ { "/": "/a/path/*" } ]
            }
        ]
    }
}
```

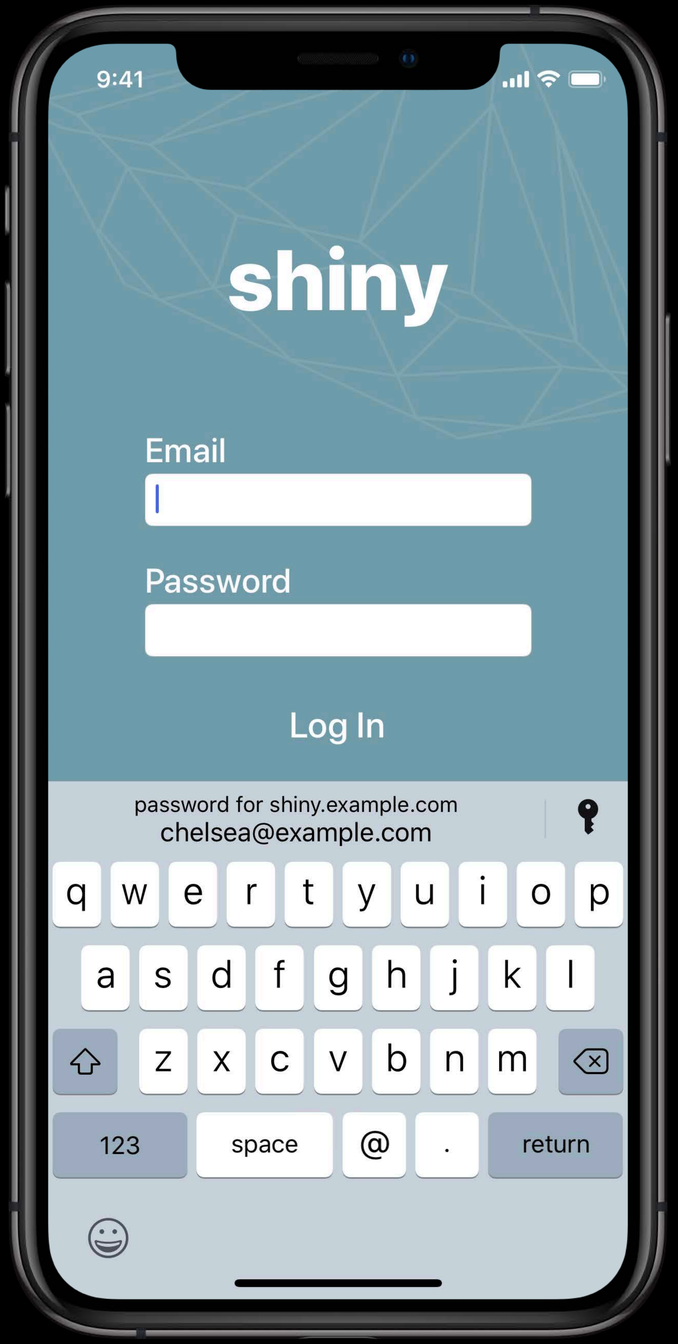
```
// https://example.com/.well-known/apple-app-site-association
// Universal Links
// Handling multiple apps (iOS 13, macOS 10.15)

{
    "applinks": {
        "apps": [],
        "details": [
            {
                "appID": "A1B2C3D4E5.com.example.Shiny",
                "appIDs": [ "A1B2C3D4E5.com.example.Shiny",
                            "A1B2C3D4E5.uikitformac.com.example.Shiny" ],
                "components": [ { "/": "/a/path/*" } ]
            }
        ]
    }
}
```

```
// https://example.com/.well-known/apple-app-site-association
// Universal Links
// Maintaining compatibility with iOS prior to iOS 13

{
    "applinks": {
        "apps": [],
        "details": [
            {
                "appID": "A1B2C3D4E5.com.example.Shiny",
                "appIDs": [ "A1B2C3D4E5.com.example.Shiny",
                            "A1B2C3D4E5.uikitformac.com.example.Shiny" ],
                "components": [ { "/": "/a/path/*" } ]
            }
        ]
    }
}
```

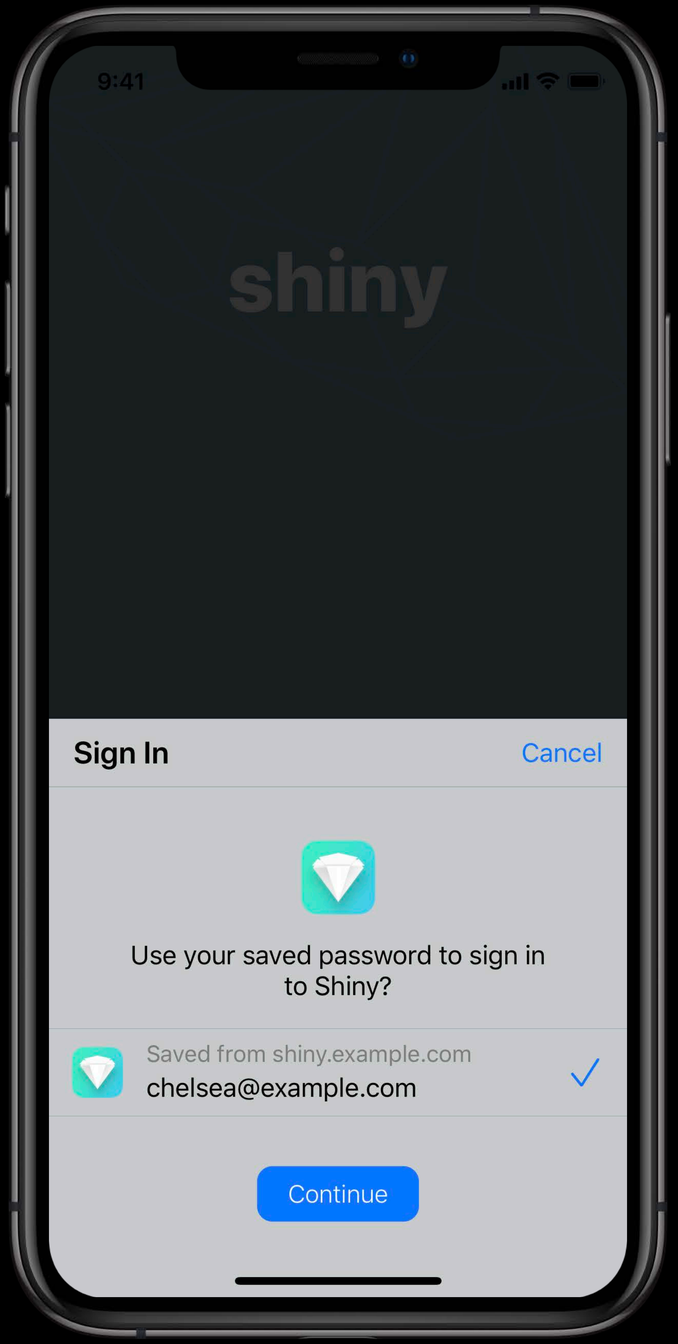
```
// https://example.com/.well-known/apple-app-site-association
// Universal Links
// Maintaining compatibility with iOS prior to iOS 13

{
    "applinks": {
        "apps": [],
        "details": [
            {
                "appID": "A1B2C3D4E5.com.example.Shiny",
                "appIDs": [ "A1B2C3D4E5.com.example.Shiny",
                            "A1B2C3D4E5.uikitformac.com.example.Shiny" ],
                "components": [ { "/": "/a/path/*" } ]
            }
        ]
    }
}
```

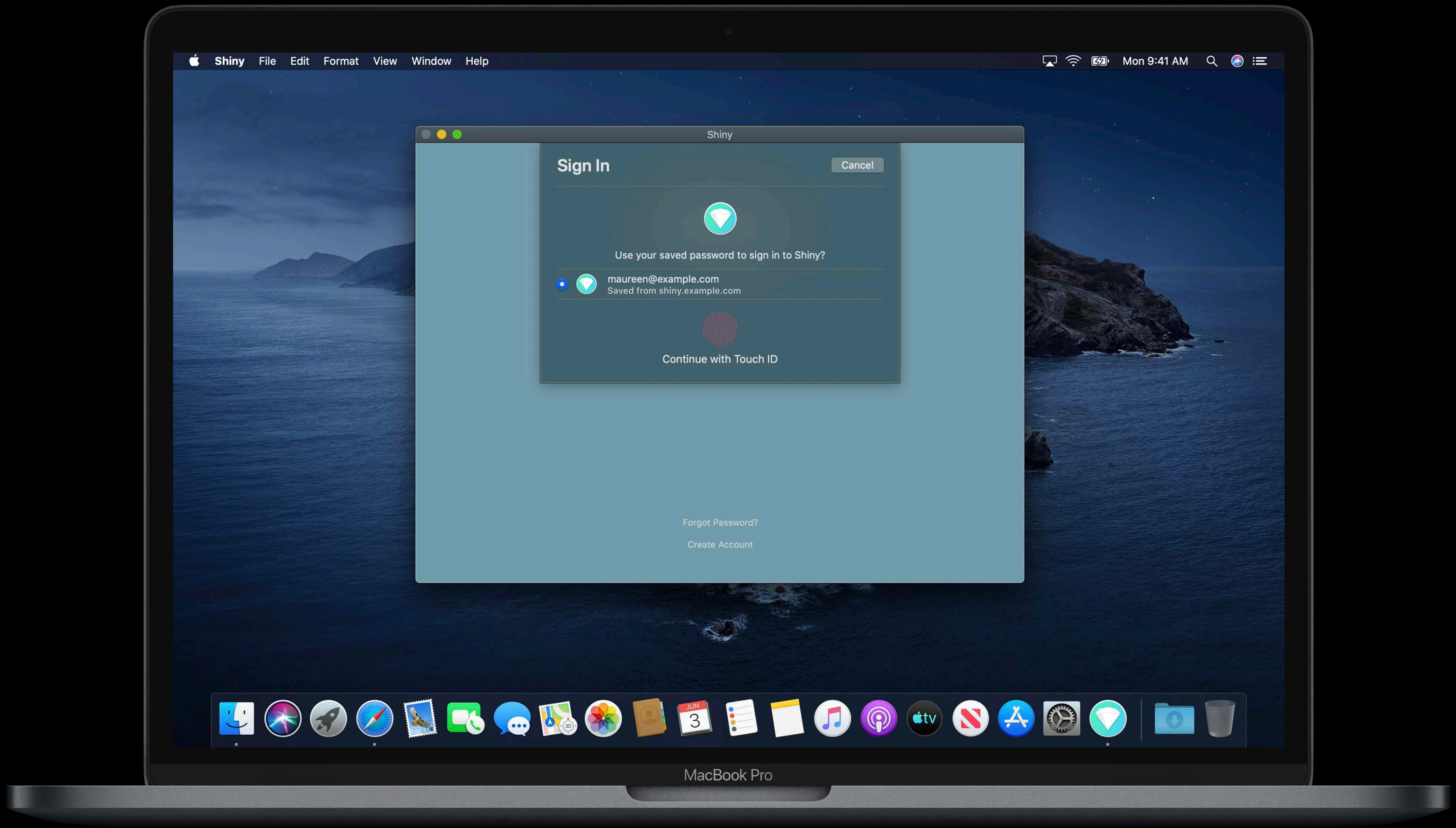
```
// https://example.com/.well-known/apple-app-site-association
// Universal Links
// Maintaining compatibility with iOS prior to iOS 13

{
    "applinks": {
        "apps": [],
        "details": [
            {
                "appID": "A1B2C3D4E5.com.example.Shiny",
                "appIDs": [ "A1B2C3D4E5.com.example.Shiny",
                            "A1B2C3D4E5.uikitformac.com.example.Shiny" ],
                "components": [ { "/": "/a/path/*" } ]
            }
        ]
    }
}
```

```
// https://example.com/.well-known/apple-app-site-association
// Universal Links
// Maintaining compatibility with iOS prior to iOS 13

{
    "applinks": {
        "apps": [],
        "details": [
            {
                "appID": "A1B2C3D4E5.com.example.Shiny",
                "appIDs": [ "A1B2C3D4E5.com.example.Shiny",
                            "A1B2C3D4E5.uikitformac.com.example.Shiny" ],
                "components": [ { "/": "/a/path/*" } ]
            }
        ]
    }
```


9:41
shiny
Email
Password
Log In
password for shiny.example.com
chelsea@example.com
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
123
space
@
.
return


9:41
shiny
Email
Password
Log In
password for shiny.example.com
chelsea@example.com
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
123
space
@
.
return


9:41
shiny
Sign In
Cancel
Use your saved password to sign in to Shiny?
Saved from shiny.example.com
chelsea@example.com
Continue


Shiny File Edit Format View Window Help
Mon 9:41 AM
Shiny
Sign In
Cancel
Use your saved password to sign in to Shiny?
maureen@example.com
Saved from shiny.example.com
Continue with Touch ID
Forgot Password?
Create Account
MacBook Pro

```
// Requesting a password-based account for sign-in

import AuthenticationServices
```

```
// Requesting a password-based account for sign-in

import AuthenticationServices
```

```
// Requesting a password-based account for sign-in

import AuthenticationServices

let controller = ASAuthorizationController(authorizationRequests: [
    ASAuthorizationPasswordProvider().createRequest()
])
```

```
// Requesting a password-based account for sign-in

import AuthenticationServices

let controller = ASAuthorizationController(authorizationRequests: [
    ASAuthorizationPasswordProvider().createRequest()
])
```

```
// Requesting a password-based account for sign-in

import AuthenticationServices

let controller = ASAuthorizationController(authorizationRequests: [
    ASAuthorizationPasswordProvider().createRequest()
])

controller.delegate = self
controller.presentationContextProvider = self
```

```
// Requesting a password-based account for sign-in

import AuthenticationServices

let controller = ASAuthorizationController(authorizationRequests: [
    ASAuthorizationPasswordProvider().createRequest()
])

controller.delegate = self
controller.presentationContextProvider = self
```

```
// Requesting a password-based account for sign-in

import AuthenticationServices

let controller = ASAuthorizationController(authorizationRequests: [
    ASAuthorizationPasswordProvider().createRequest()
])

controller.delegate = self
controller.presentationContextProvider = self

controller.performRequests()
```

```
// Requesting a password-based account for sign-in

import AuthenticationServices

let controller = ASAuthorizationController(authorizationRequests: [
    ASAuthorizationPasswordProvider().createRequest()
])

controller.delegate = self
controller.presentationContextProvider = self

controller.performRequests()
```

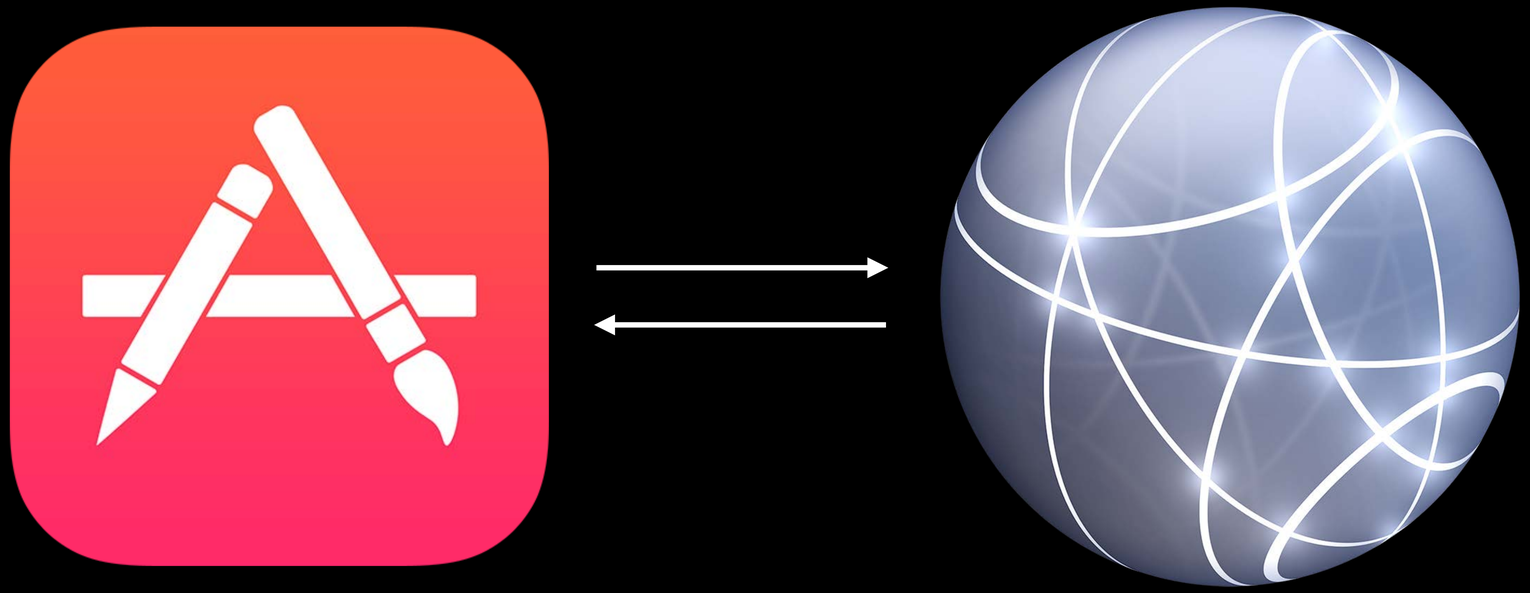
```
// Requesting a password-based account for sign-in

// ASAuthorizationControllerDelegate
func authorizationController(controller: ASAuthorizationController,
    didCompleteWithAuthorization authorization: ASAuthorization) {
    if let credential = authorization.credential as? ASPasswordCredential {
        // Use credential.user and credential.password to sign in.
    }
}
```

```
// Requesting a password-based account for sign-in

// ASAuthorizationControllerDelegate
func authorizationController(controller: ASAuthorizationController,
    didCompleteWithAuthorization authorization: ASAuthorization) {
    if let credential = authorization.credential as? ASPasswordCredential {
        // Use credential.user and credential.password to sign in.
    }
}
```

```
// Requesting a password-based account for sign-in

// ASAuthorizationControllerDelegate
func authorizationController(controller: ASAuthorizationController,
    didCompleteWithAuthorization authorization: ASAuthorization) {
    if let credential = authorization.credential as? ASPasswordCredential {
        // Use credential.user and credential.password to sign in.
    }
}

func authorizationController(controller: ASAuthorizationController,
    didCompleteWithError error: Error) {
    // Fall back to login UI.
}
```

```
// Requesting a password-based account for sign-in

// ASAuthorizationControllerDelegate
func authorizationController(controller: ASAuthorizationController,
    didCompleteWithAuthorization authorization: ASAuthorization) {
    if let credential = authorization.credential as? ASPasswordCredential {
        // Use credential.user and credential.password to sign in.
    }
}

func authorizationController(controller: ASAuthorizationController,
    didCompleteWithError error: Error) {
    // Fall back to login UI.
}
```

Associated Domains



Introducing Password AutoFill for Apps	WWDC 2017
What’s New in Universal Links	WWDC 2019

```
// Requesting both Sign In with Apple and password-based accounts for sign-in

let controller = ASAuthorizationController(authorizationRequests: [
    ASAuthorizationAppleIDProvider().createRequest()
    ASAuthorizationPasswordProvider().createRequest()
])

controller.delegate = self
controller.presentationContextProvider = self

controller.performRequests()
```

```
// Requesting both Sign In with Apple and password-based accounts for sign-in

let controller = ASAuthorizationController(authorizationRequests: [
    ASAuthorizationAppleIDProvider().createRequest()
    ASAuthorizationPasswordProvider().createRequest()
])

controller.delegate = self
controller.presentationContextProvider = self

controller.performRequests()
```

```swift
// Requesting both Sign In with Apple and password-based accounts for sign-in

// ASAuthorizationControllerDelegate
func authorizationController(controller: ASAuthorizationController,
                             didCompleteWithAuthorization authorization: ASAuthorization) {
    switch authorization.credential {
    case let credential as ASAuthorizationAppleIDCredential:
        // Sign the user in using the Apple credential.

    case let credential as ASPasswordCredential:
        // Sign the user in using the password credential.

    default:
        // Fall back to login UI.
   }
}
```

```swift
// Requesting both Sign In with Apple and password-based accounts for sign-in

// ASAuthorizationControllerDelegate
func authorizationController(controller: ASAuthorizationController,
                             didCompleteWithAuthorization authorization: ASAuthorization) {
    switch authorization.credential {
    case let credential as ASAuthorizationAppleIDCredential:
        // Sign the user in using the Apple credential.

    case let credential as ASPasswordCredential:
        // Sign the user in using the password credential.

    default:
        // Fall back to login UI.
    }
}
```

Sign In with Apple

Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS

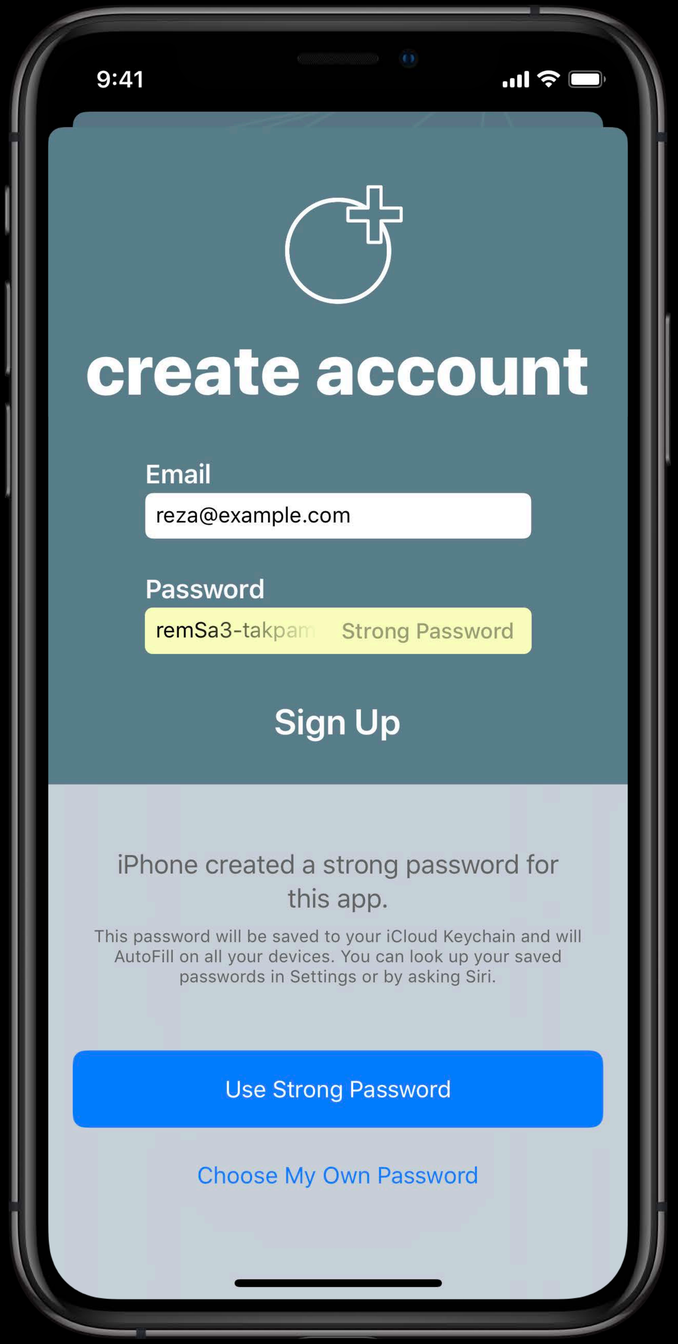
Sign In with Apple

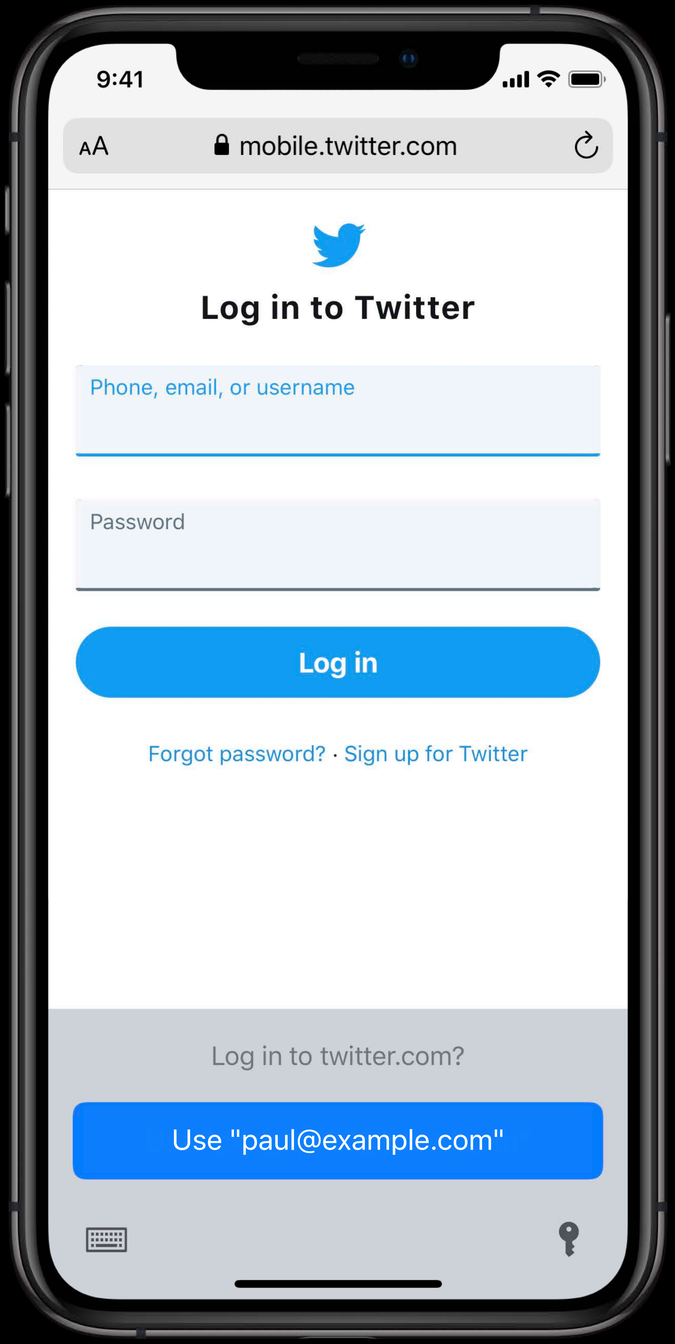
Password-based authentication

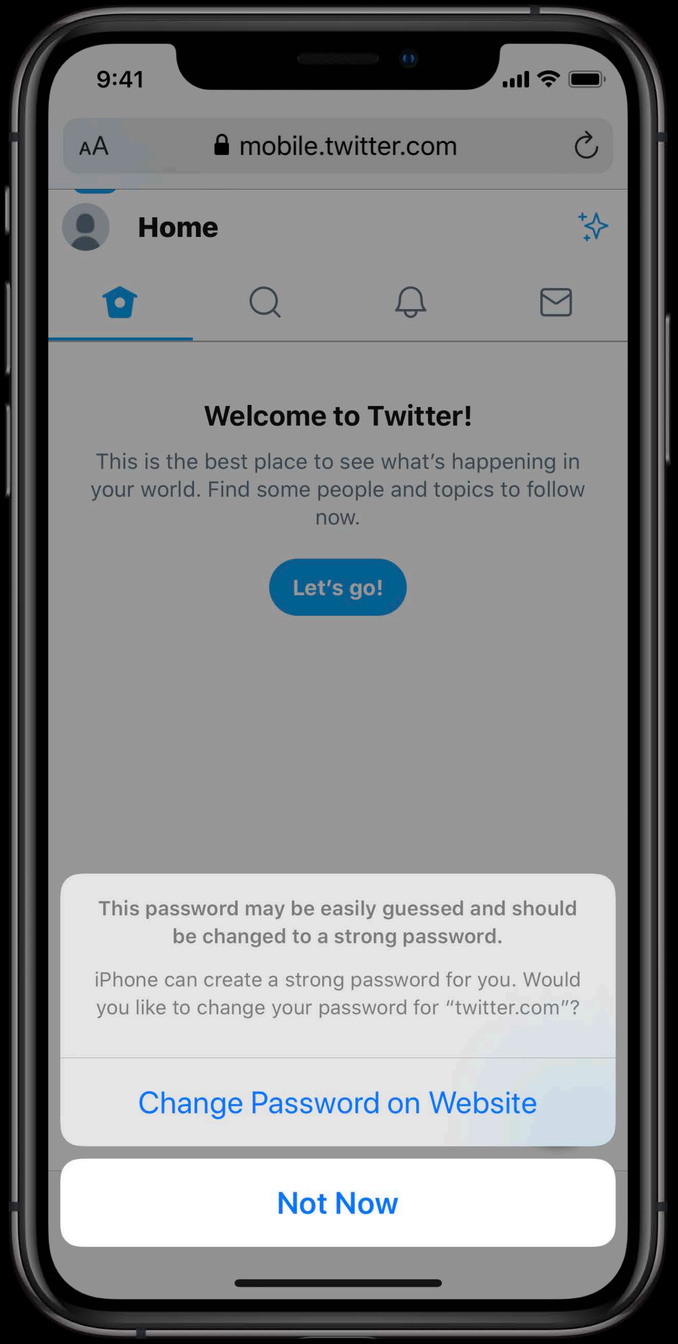
Warnings for weak passwords

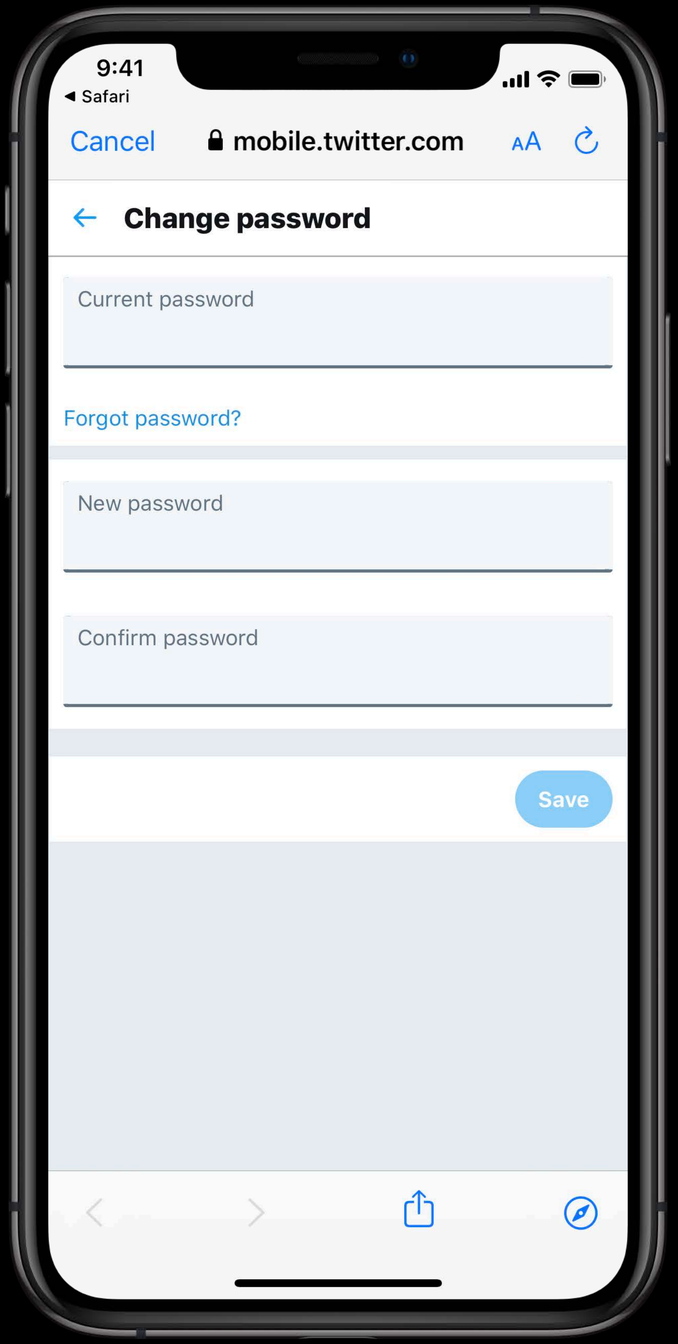
OAuth sign-in

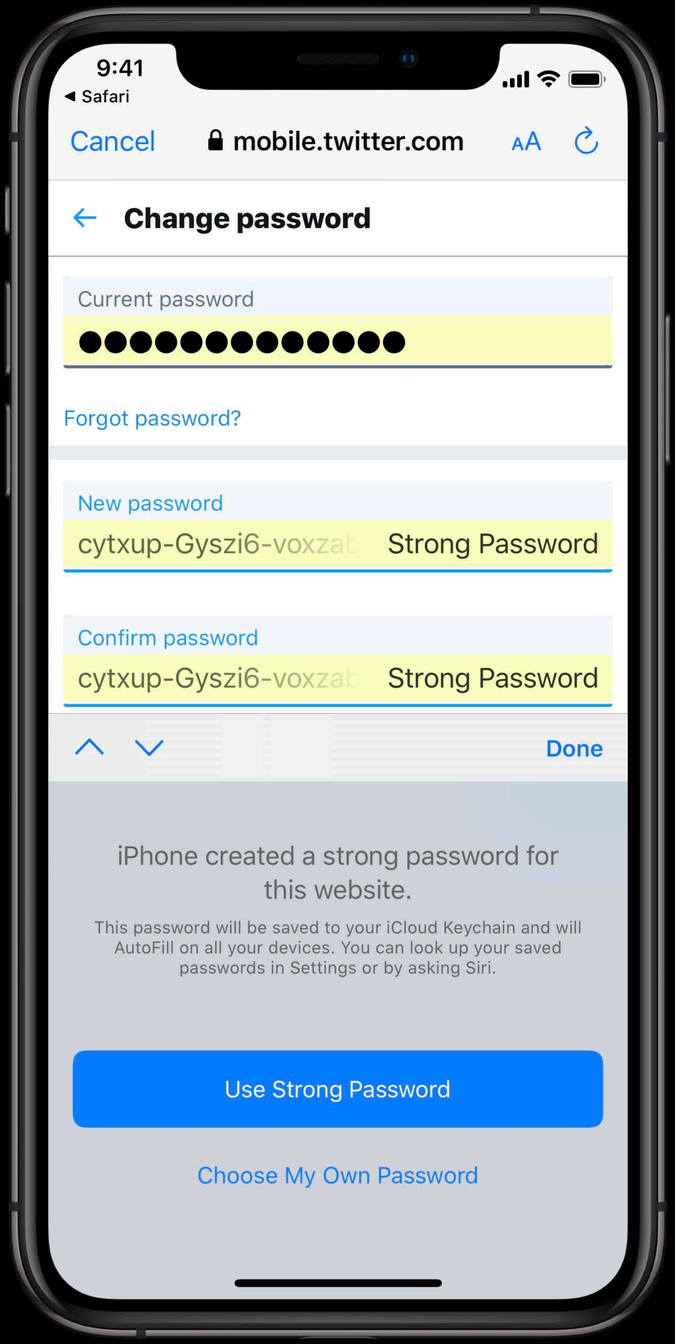
USB security keys on macOS

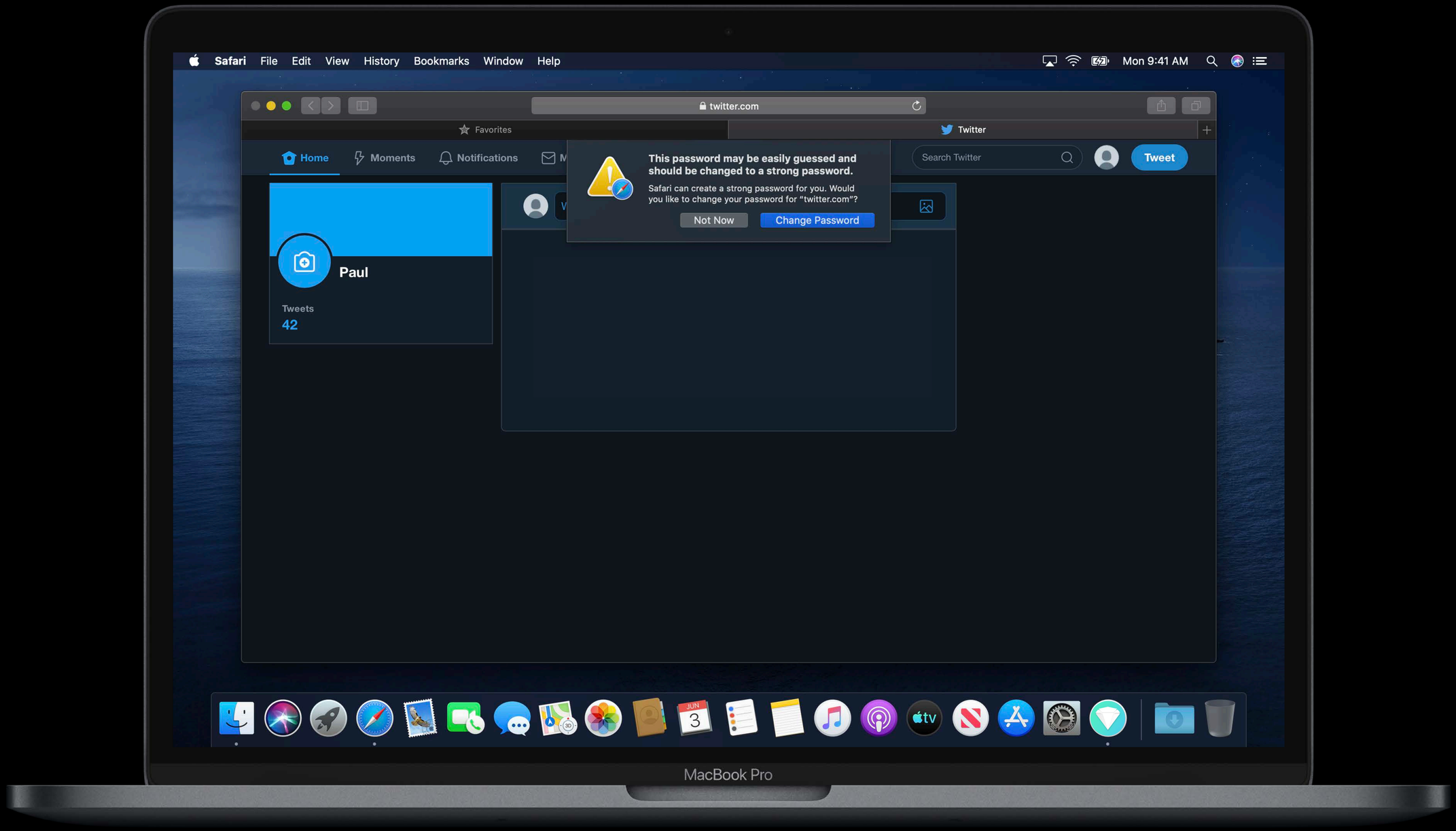

9:41
create account
Email
reza@example.com
Password
remSa3-takpam
Strong Password
Sign Up
iPhone created a strong password for this app.
This password will be saved to your iCloud Keychain and will AutoFill on all your devices. You can look up your saved passwords in Settings or by asking Siri.
Use Strong Password
Choose My Own Password

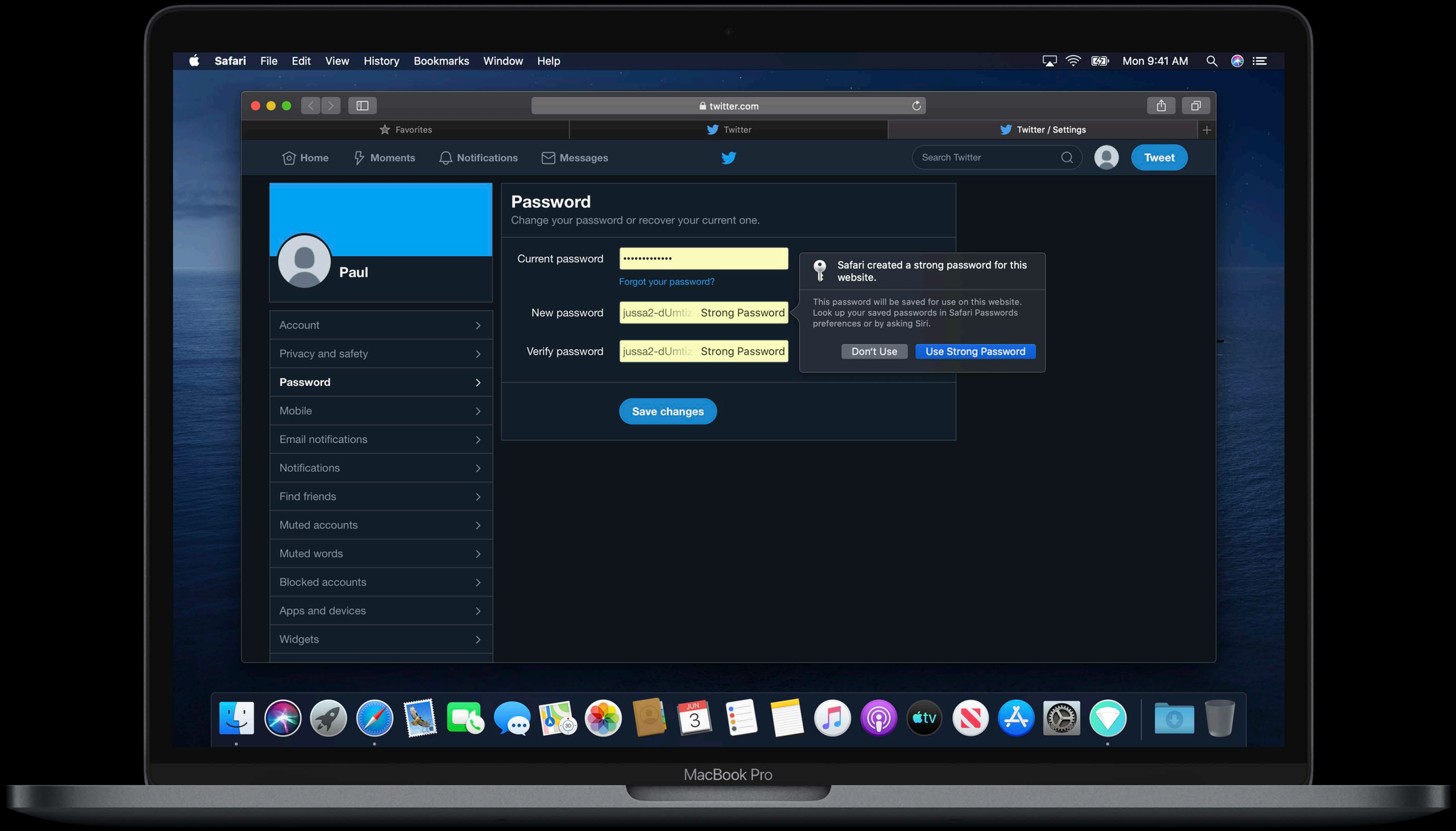

9:41
AA
mobile.twitter.com
Log in to Twitter
Phone, email, or username
Password
Log in
Forgot password? · Sign up for Twitter
Log in to twitter.com?
Use "paul@example.com"


9:41
AA
mobile.twitter.com
Home
Welcome to Twitter!
This is the best place to see what's happening in your world. Find some people and topics to follow now.
Let's go!
This password may be easily guessed and should be changed to a strong password.
iPhone can create a strong password for you. Would you like to change your password for "twitter.com"?
Change Password on Website
Not Now


9:41
Safari
Cancel
mobile.twitter.com
AA
Change password
Current password
Forgot password?
New password
Confirm password
Save


9:41
Safari
Cancel
mobile.twitter.com
AA
Change password
Current password
Forgot password?
New password
cytxup-Gyszi6-voxzab
Strong Password
Confirm password
cytxup-Gyszi6-voxzab
Strong Password
Done
iPhone created a strong password for this website.
This password will be saved to your iCloud Keychain and will AutoFill on all your devices. You can look up your saved passwords in Settings or by asking Siri.
Use Strong Password
Choose My Own Password


Safari File Edit View History Bookmarks Window Help
Mon 9:41 AM
twitter.com
Favorites
Twitter
Home
Moments
Notifications
Search Twitter
Tweet
This password may be easily guessed and should be changed to a strong password.
Safari can create a strong password for you. Would you like to change your password for "twitter.com"?
Not Now
Change Password
Paul
Tweets
42
MacBook Pro


Safari File Edit View History Bookmarks Window Help
Mon 9:41 AM
twitter.com
Favorites
Twitter
Twitter / Settings
Home
Moments
Notifications
Messages
Search Twitter
Tweet
Paul
Account
Privacy and safety
Password
Mobile
Email notifications
Notifications
Find friends
Muted accounts
Muted words
Blocked accounts
Apps and devices
Widgets
Password
Change your password or recover your current one.
Current password
Forgot your password?
New password
Strong Password
Verify password
Strong Password
Safari created a strong password for this website.
This password will be saved for use on this website. Look up your saved passwords in Safari Passwords preferences or by asking Siri.
Don't Use
Use Strong Password
Save changes
MacBook Pro

https://example.com/
.well-known/change-password

Well-Known URL for Changing Passwords

wicg.github.io/change-password-url

Sign In with Apple

Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS

Sign In with Apple

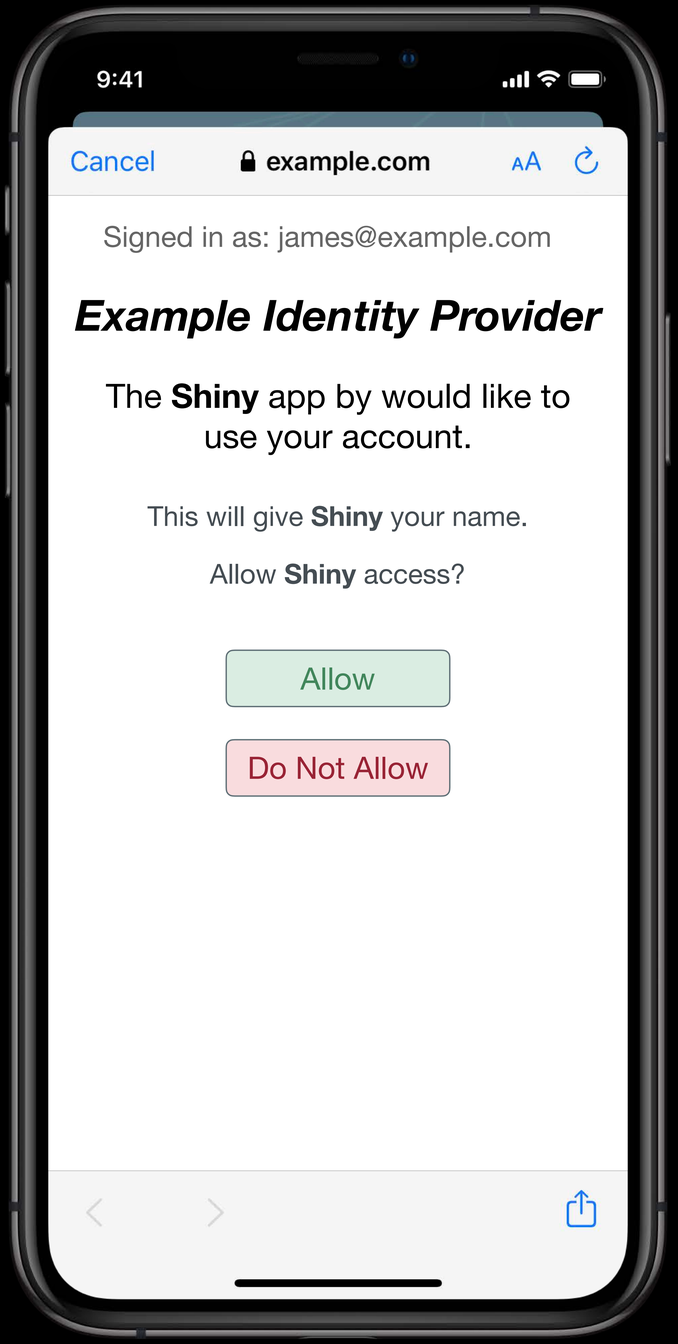
Password-based authentication

Warnings for weak passwords

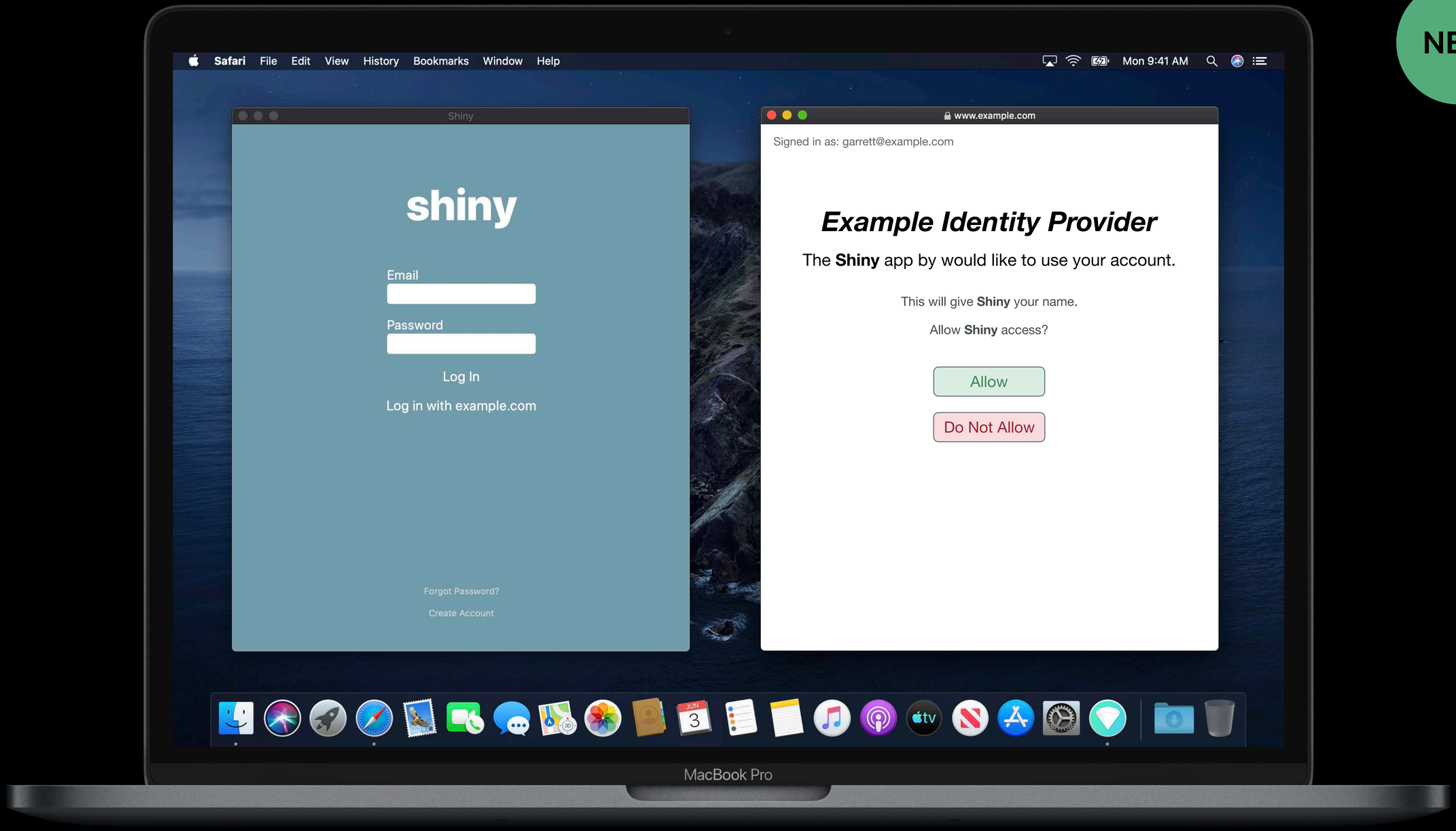
OAuth sign-in

USB security keys on macOS


9:41
shiny
Email
"Shiny" Wants to Use "example.com" to Sign In
This allows the app and website to share information about you.
Cancel
Continue
Log in with example.com
Forgot Password?
Create Account


9:41
Cancel
example.com
AA
Signed in as: james@example.com
Example Identity Provider
The Shiny app by would like to use your account.
This will give Shiny your name.
Allow Shiny access?
Allow
Do Not Allow


9:41
today's shiny
Sign Out


NEW
Safari File Edit View History Bookmarks Window Help
Mon 9:41 AM
Shiny
shiny
Email
Password
Log In
Log in with example.com
Forgot Password?
Create Account
www.example.com
Signed in as: garrett@example.com
Example Identity Provider
The Shiny app by would like to use your account.
This will give Shiny your name.
Allow Shiny access?
Allow
Do Not Allow
MacBook Pro

OAuth with ASWebAuthenticationSession

More private sign-in

Supporting multiple windows

Migrating from SFAuthenticationSession

OAuth with ASWebAuthenticationSession

More private sign-in

Supporting multiple windows

Migrating from SFAuthenticationSession


9:41
shiny
Email
"Shiny" Wants to Use "example.com" to Sign In
This allows the app and website to share information about you.
Cancel
Continue
Forgot Password?
Create Account


9:41
Cancel
example.com
AA
Example Identity Provider
Sign in to allow the app Shiny to connect to your account.
Email
Password
Sign in
Log in to example.com?
Use “rmondello@example.com"

NEW

```
// Giving users additional privacy using ASWebAuthenticationSession

import AuthenticationServices

func signIn() {
    let session = ASWebAuthenticationSession(url: URL(string: "https://example.com/login")!,
        callbackURLScheme: "custom-oauth-scheme",
        completionHandler: { (callbackURL, error) in
            // Handle login
    })
    session.prefersEphemeralWebBrowserSession = true
    session.start()
}
```

```
// Giving users additional privacy using ASWebAuthenticationSession

import AuthenticationServices

func signIn() {
    let session = ASWebAuthenticationSession(url: URL(string: "https://example.com/login")!,
        callbackURLScheme: "custom-oauth-scheme",
        completionHandler: { (callbackURL, error) in
            // Handle login
    })
    session.prefersEphemeralWebBrowserSession = true
    session.start()
}
```

NEW

OAuth with ASWebAuthenticationSession

More private sign-in

Supporting multiple windows

Migrating from SFAuthenticationSession

```
// Providing a window to the authentication session

import AuthenticationServices

func signIn() {
    let session = ASWebAuthenticationSession(url: URL(string: "https://example.com/login")!,
        callbackURLScheme: "custom-oauth-scheme",
        completionHandler: { (callbackURL, error) in
            // Handle login
    })
    session.presentationContextProvider = self
    session.start()
}

func presentationAnchor(for _: ASWebAuthenticationSession) -> ASPresentationAnchor {
    return /* the relevant window */
}
```

```
// Providing a window to the authentication session

import AuthenticationServices

func signIn() {
    let session = ASWebAuthenticationSession(url: URL(string: "https://example.com/login")!,
        callbackURLScheme: "custom-oauth-scheme",
        completionHandler: { (callbackURL, error) in
            // Handle login
    })
    session.presentationContextProvider = self
    session.start()
}

func presentationAnchor(for _: ASWebAuthenticationSession) -> ASPresentationAnchor {
    return /* the relevant window */
}
```

```swift
// Providing a window to the authentication session

import AuthenticationServices

func signIn() {
    let session = ASWebAuthenticationSession(url: URL(string: "https://example.com/login")!,
        callbackURLScheme: "custom-oauth-scheme",
        completionHandler: { (callbackURL, error) in
            // Handle login
    })
    session.presentationContextProvider = self
    session.start()
}

func presentationAnchor(for _: ASWebAuthenticationSession) -> ASPresentationAnchor {
    return /* the relevant window */
}
```

OAuth with ASWebAuthenticationSession

More private sign-in

Supporting multiple windows

Migrating from SFAuthenticationSession

SFAuthenticationSession

ASWebAuthenticationSession

Sign In with Apple

Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS

Sign In with Apple

Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS


Security Key


Security Key

FIDO2-compliant

FIDO2-compliant


Security Key

Sign In with Apple

Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS

Sign In with Apple

Password-based authentication

Warnings for weak passwords

OAuth sign-in

USB security keys on macOS

Summary

Summary

Adopt Sign In with Apple

Summary

Adopt Sign In with Apple

Make password sign-in easy

Summary

Adopt Sign In with Apple

Make password sign-in easy

Implement `/.well-known/change-password`

More Information

developer.apple.com/wwdc19/516


WWDC19