SIGN-IN WITH APPLE

SIGN-IN WITH APPLE

- Users can sign into third-party apps with their Apple ID
- Will be mandatory for all apps with third-party logins
- Users can hide their E-mail address
    - Apple generates a new mail address
    - Forwards mails to the user’s real address

INITIATING A LOGIN

```
let appleIDProvider = ASAuthorizationAppleIDProvider()
let request = appleIDProvider.createRequest()
request.requestedScopes = [.fullName, .email]

let authorizationController =
ASAuthorizationController(authorizationRequests: [request])

// This is not required in all setups
authorizationController.presentationContextProvider = self

authorizationController.delegate = self
authorizationController.performRequests()
```

CONTEXT PROVIDER

- Provide a window to present the login dialog
- Needed in multi-window environments

```
typealias ASPresentationAnchor = UIWindow // iOS, Mac Catalyst, tvOS
typealias ASPresentationAnchor = NSWindow // macOS

extension LoginViewController:
        ASAuthorizationControllerPresentationContextProviding {

    func presentationAnchor(for controller:
           ASAuthorizationController) -> ASPresentationAnchor {

        return UIApplication.shared.windows.first!
    }

}
```

GET LOGIN RESULTS

```
extension LoginViewController: ASAuthorizationControllerDelegate {

    func authorizationController(controller: ASAuthorizationController,
                                 didCompleteWithAuthorization authorization: ASAuthorization) {
        guard let credentials = authorization.credential
                                              as? ASAuthorizationAppleIDCredential else {
            print("Could not cast to ASAuthorizationAppleIDCredential.")
            return
        }

        // Send credentials to backend for verification
    }

    func authorizationController(controller: ASAuthorizationController,
                                 didCompleteWithError error: Error) {
        guard let authError = error as? ASAuthorizationError else {
            print(error)
            return
        }

        // Handle Error
        // Cancelling will also result in an error
    }

}
```

ASAuthorizationAppleIDCredential

- UserID
- Full Name ?
- E-Mail Address ?
- Real User Status
  - Indicates if the user is likely a real person
- Identity Token
  - JWT signed by Apple which includes user information
- Authorization Code
  - Used as proof for the app’s server backend

VERIFYING THE USER (BACKEND)

- Generate a key for sign in with Apple
  - https://developer.apple.com/account/resources/authkeys/list
- Generate the client secret using the key
  - https://developer.apple.com/documentation/signinwithapplerestapi/generate_and_validate_tokens
- Send the client secret and the authorization code to apple for verification

DEMO

Observe Changes in Authentication

- Check if the user revoked the authentication

```
ASAuthorizationAppleIDProvider()
    .getCredentialState(forUserID: user) { state, error in
    // Check error and current authentication state
}
```

- NotificationCenter observer

~~`NSNotification.Name.ASAuthorizationAppleIDProviderCredentialRevoked`~~

  - No longer available since Xcode Beta 3

PRIVATE MAIL RELAY

- SPF DNS TXT Entries Mandatory for Domains
- Domain needs to be verified in the Developer Portal
  - https://developer.apple.com/account/resources/services/configure
- Mails need to be sent from a verified domain

Thank You