How to use SecureStore static library

This document will give you a high-level information and examples about how to use the SecureStore library. Please refer to the Integration document for getting started and adding the library to your project.

REVISION HISTORY

DATE	VERSION	AUTHOR	UPDATE DESCRIPTION
July 26, 2020	1.0	Gautham Velappan	Initial Draft

General Usage

The library has 3 important classes for keychain storage,

1. KeyStore
2. CredentialStore
3. DataStore

KeyStore

This class gives you ability to store, retrieve, update and delete keys for a specific user.

First, create a KeyStore instance.

```
1.  let keyStore = KeyStore()
```

All the methods in the KeyStore class needs a user object.

```
1.  let user = String("tester@poc.com")
```

Add a key to the KeyStore:

```
1.  let myKey = String("MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCqGKukO1De7zhZj6+H0qtjTkVxwTC
    pvKe4eCZ0FPqri0cb2JZfXJ/DgYSF6vUpwmJG8wVQZKjeGcjDOL5UlsuusFncCzWBQ7RKNUSesmQRMSGkVb1/3j
    +skZ6UtW+5u09lHNsj6tQ51s1SPrCBkedbNf0Tp0GbMJDyR4e9T04ZZwIDAQAB")
2.
3.
4.          keyStore.add(myKey, for: user, completion: { error in
5.              if let addError = error {
6.                  print("Add key failed with error: \(addError.localizedDescription)")
7.              } else {
8.                  print("Add key success")
9.              }
10.         })
```

Retrieve user keys from the KeyStore:

```
1.  let myKeys = keyStore.getValues(for: user, completion: { error in
2.              if let getError = error {
3.                  print("Get key failed with error: \(getError.localizedDescription)")
4.              } else {
5.                  print("Get key success")
6.              }
7.          })
```

```
8.
9.          print("MyKeys: \(myKeys)")
```

Update a key to the KeyStore:

```
1.  let myOldKey = String("MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCqGKukO1De7zhZj6+H0qtjTkVx
    wTCpvKe4eCZ0FPqri0cb2JZfXJ/DgYSF6vUpwmJG8wVQZKjeGcjDOL5UlsuusFncCzWBQ7RKNUSesmQRMSGkVb1
    /3j+skZ6UtW+5u09lHNsj6tQ51s1SPrCBkedbNf0Tp0GbMJDyR4e9T04ZZwIDAQAB")
2.
3.  let myNewKey = String("FncCzWBQ7RKNUSesmQRMSGkVb1/3j+skZ6UtW+5u09lHNsj6tQ51s1SPrCBkedbN
    f0Tp0GbMJDyR4e9T04ZZwIDAQABMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCqGKukO1De7zhZj6+H0qtj
    TkVxwTCpvKe4eCZ0FPqri0cb2JZfXJ/DgYSF6vUpwmJG8wVQZKjeGcjDOL5Ulsuus")
4.
5.  keyStore.update(myOldKey, with: myNewKey, for: user, completion: { error in
6.      if let updateError = error {
7.          print("Update key failed with error: \(updateError.localizedDescription)")
8.      } else {
9.          print("Update key success")
10.     }
11. })
```

Delete all the keys from the KeyStore:

```
1.  keyStore.delete(for: user, completion: { error in
2.      if let deleteError = error {
3.          print("Delete key failed with error: \(deleteError.localizedDescription)")
4.      } else {
5.          print("Delete key success")
6.      }
7.  })
```

CredentialStore

First, create a CredentialStore instance.

```
let credentialStore = CredentialStore()
```

All the methods in the CredentialStore class needs a user object.

```
let user = String("tester@poc.com")
```

Add a credential to the CredentialStore:

```
let credentialJSON: [String: Any] =
    ["@context": ["https://www.w3.org/2018/credentials/v1"],
     "type": ["VerifiableCredential"],
     "id": "did:hpass:012345:0163456789abcdef012345#vc-8baa161f-0caf-41ad-ae0a-112c51026829",
     "issuer": "did:hpass:012345:0123456789abcdef012345",
     "issuanceDate": "2010-01-01T19:73:24Z"]

//Option 1
credentialStore.add(credentialJSON, for: user, completion: { error in
    if let addError = error {
        print("Add credential failed with error: \(addError.localizedDescription)")
    } else {
        print("Add credential success")
    }
})

//Option 2
let credentialObject = Credential(value: credentialJSON)

credentialStore.add(credentialObject, for: user, completion: { error in
    if let addError = error {
        print("Add credential failed with error: \(addError.localizedDescription)")
    } else {
        print("Add credential success")
    }
})
```

Retrieve user credentials from the CredentialStore:

```
1.  //Option 1
2.  let myCredentials: [[String: Any]]? = credentialStore.getValues(for: user, completion:
     { error in
3.      if let getError = error {
4.          print("Get credential failed with error: \(getError.localizedDescription)")
5.      } else {
6.          print("Get credential success")
7.      }
8.  })
9.
10. print("My Credentials: \(myCredentials)")
11.
12. //Option 2
13. let myCredentialObjects: [Credential]? = credentialStore.getValues(for: user, completio
    n:  { error in
14.     if let getError = error {
15.         print("Get credential failed with error: \(getError.localizedDescription)")
16.     } else {
17.         print("Get credential success")
18.     }
19. })
20.
21. print("My Credentials: \(myCredentialObjects)")
```

Update a credential to the CredentialStore:

```
1.  let oldCredentialJSON: [String: Any] =
2.      ["@context": ["https://www.w3.org/2018/credentials/v1"],
3.       "type": ["VerifiableCredential"],
4.       "id": "did:hpass:012345:0163456789abcdef012345#vc-8baa161f-0caf-41ad-ae0a-
    112c51026829",
5.       "issuer": "did:hpass:012345:0123456789abcdef012345",
6.       "issuanceDate": "2010-01-01T19:73:24Z"]
7.
8.  let newCredentialJSON: [String: Any] =
9.      ["@context": ["https://www.w3.org/2018/credentials/v2"],
10.      "type": ["VerifiableCredential"],
11.      "id": "did:hpass:012345:0163456789abcdef012345#vc-8baa161f-0caf-41ad-ae0a-
    112c51026829",
12.      "issuer": "did:hpass:012345:0123456789abcdef012345",
13.      "issuanceDate": "2010-07-07T21:20:24Z"]
14.
15. //Option 1
16. credentialStore.update(oldCredentialJSON, with: newCredentialJSON, for: user, completio
    n: { error in
17.     if let updateError = error {
18.         print("Update Credential failed with error: \(updateError.localizedDescription)
    ")
19.     } else {
20.         print("Update Credential success")
21.     }
```

```
22. })
23.
24. //Option 2
25. let oldCredentialObject = Credential(value: oldCredentialJSON)
26. let newCredentialObject = Credential(value: newCredentialJSON)
27.
28. credentialStore.update(oldCredentialObject, with: newCredentialObject, for: user, compl
    etion: { error in
29.     if let updateError = error {
30.         print("Update Credential failed with error: \(updateError.localizedDescription)
    ")
31.     } else {
32.         print("Update Credential success")
33.     }
34. })
```

Delete a credential from the CredentialStore:

```
1.  let credentialJSON: [String: Any] =
2.      ["@context": ["https://www.w3.org/2018/credentials/v2"],
3.       "type": ["VerifiableCredential"],
4.       "id": "did:hpass:012345:0163456789abcdef012345#vc-8baa161f-0caf-41ad-ae0a-
    112c51026829",
5.       "issuer": "did:hpass:012345:0123456789abcdef012345",
6.       "issuanceDate": "2010-07-07T21:20:24Z"]
7.
8.  //Option 1
9.  credentialStore.delete(value: credentialJSON, for: user, completion: { error in
10.     if let deleteError = error {
11.         print("Delete Credential failed with error: \(deleteError.localizedDescription)
    ")
12.     } else {
13.         print("Delete Credential success")
14.     }
15. })
16.
17. //Option 2
18. let credentialObject = Credential(value: credentialJSON)
19.
20. credentialStore.delete(value: credentialObject, for: user, completion: { error in
21.     if let deleteError = error {
22.         print("Delete Credential failed with error: \(deleteError.localizedDescription)
    ")
23.     } else {
24.         print("Delete Credential success")
25.     }
26. })
```