SCIENCES SORBONNE UNIVERSITÉ

Handling settings

Fabrice.Kordon@lip6.fr

LIP6

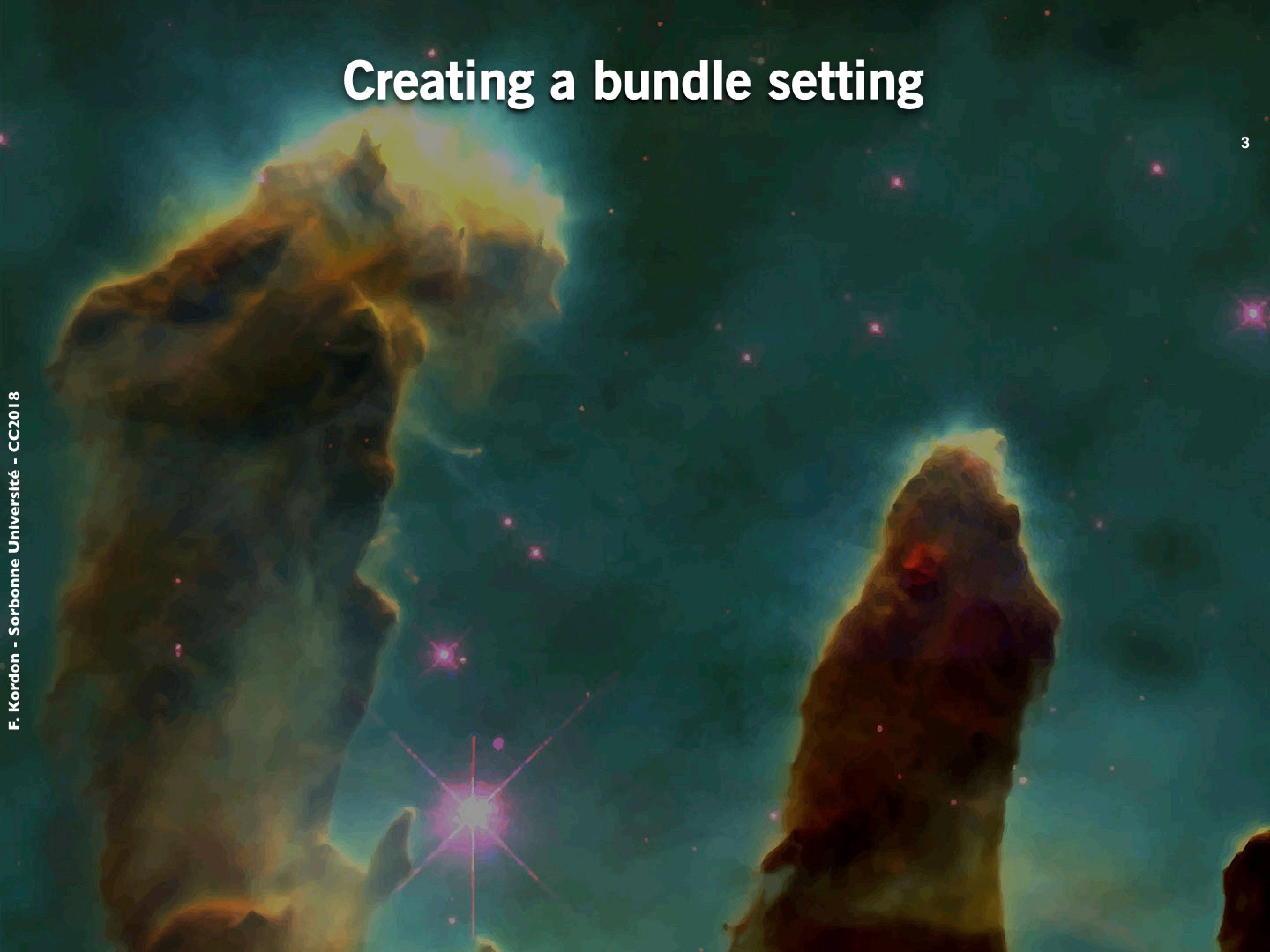
As an introduction…

Often necessary

- To remember some preference for your App
  - **Remember the level in «Asteroid»**

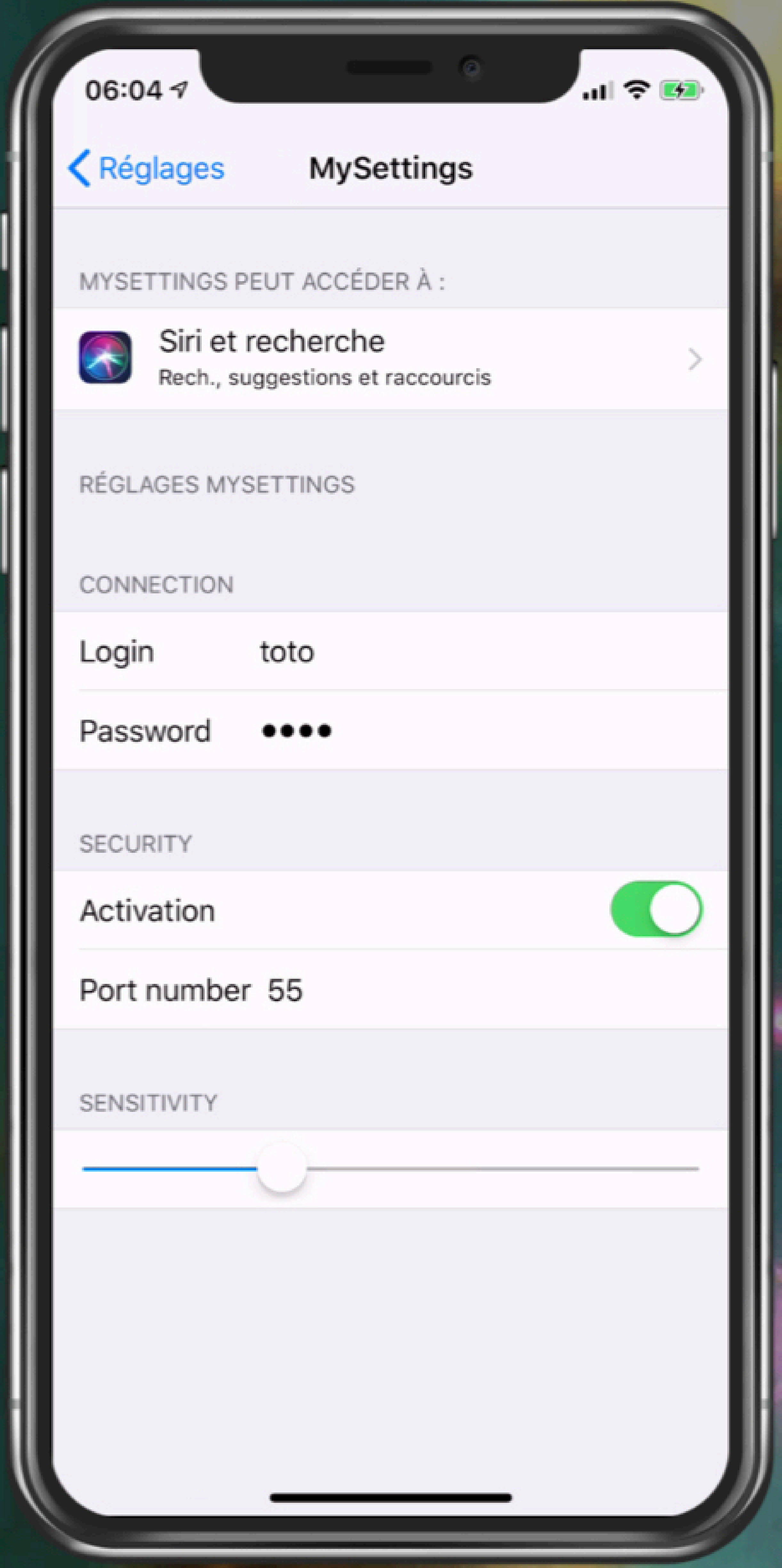
Two possibilities

- Handle it at the App level
  - **Stored in a dedicated file**
- Use of a «Setting Bundle» (preferences)
  - **Stored in the project as a resource**
  - **Setting parameters declared in a .plist file**



Creating a bundle setting

Categories in Root.plist



06:04

Réglages MySettings

MYSETTINGS PEUT ACCÉDER À :

Siri et recherche
Rech., suggestions et raccourcis

RÉGLAGES MYSETTINGS

CONNECTION

Login toto

Password ●●●●

SECURITY

Activation [✓]

Port number 55

SENSITIVITY

Catégories

- Group
- Multi Value
- Slider
- Text Field
- Title
- Toggle Switch

Access to the setting properties

Tanks to UserDefaults (or NSUserDefaults)

- Get a reference to the shared object preferences
  - **Property standard**
- Manipulation like a dictionary (via functions)

```
func string(forKey defaultName: String) -> String?
func bool(forKey defaultName: String) -> Bool
func integer(forKey defaultName: String) -> Int
func float(forKey defaultName: String) -> Float
func double(forKey defaultName: String) -> Double

func dictionaryRepresentation() -> [String : Any]
```

First execution

- Programmatically handle default values
- Set such default in the Settings

Write settings/preferences

- «register» a dictionary to UserDefaults.standard

Invoking the Apps's settings

- **Similar to the invocation of an external App**
  - Already show in a previous video
    - **Method open in UIApplication**
  - Predefined URL for this
    - **UIApplication.openSettingsURLString (Swift)**
    - **UIApplication.UIApplicationOpenSettingsURLString (Objective-C)**
- **Bonus**
  - You go directly to the settings of your App
    - **If a bundle is defined in the project**

Security concerns

- **Default storage of data is unsecure**
  - Stored in a .plist file (XML)
  - No encryption
    - **in the simulator data**
    - **in the backup space of your device (on your computer)**
      - **more difficult of backup is encrypted**
  - Never store critical information in the settings
- **For critical information, use Keychain**
  - Simplified variant of the MacOS Keychain
    - **Encrypted data**
    - **An application can only access its own informations**

KeyChain principles in a nutshell

Principles

- Offered in the Security framework
- You manipulate a dictionary
  - The App chooses the keys
  - You may add, update, destroy, compare entries
- Protection levels
  - kSecAttrAccessibleWhenUnlocked
  - kSecAttrAccessibleAfterFirstUnlock
  - kSecAttrAccessibleAlways
  - kSecAttrAccessibleWhenUnlockedThisDeviceOnly
  - kSecAttrAccessibleAfterFirstUnlockThisDeviceOnly
  - kSecAttrAccessibleAlwaysThisDeviceOnly

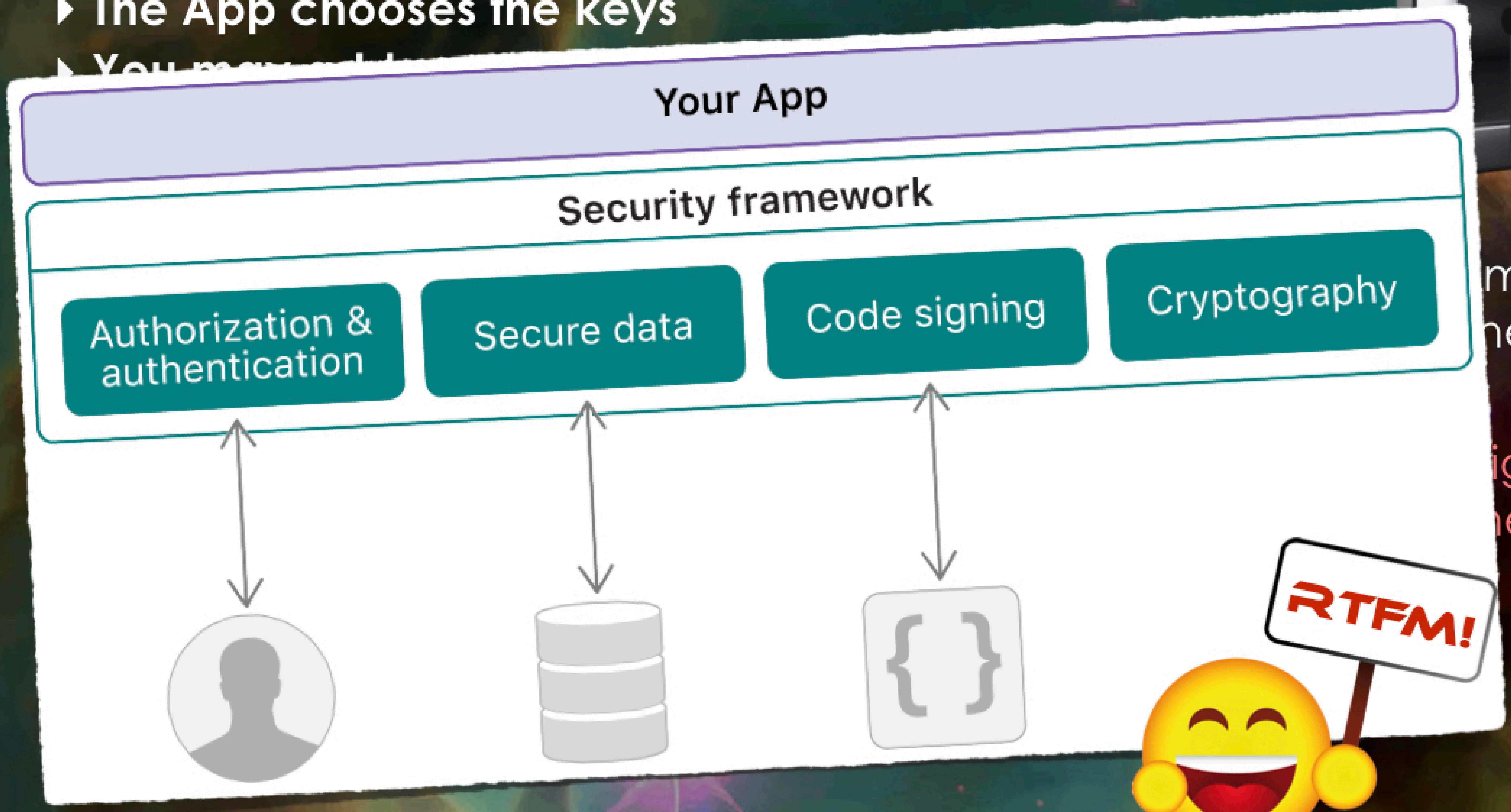
May migrate to another device

No migration to another device

KeyChain principles in a nutshell

Principles

- Offered in the Security framework
- You manipulate a dictionary
  - The App chooses the keys
  - You may add k



migrate to
her device

igration to
her device

As a conclusion…

- **Settings = standard way to store preferences**
  - Remember some parameters for your Apps
  - The standard way to proceed
  - No so complex…

… unless KeyChain is used