plist data 저장

강사 주영민

Property list

Key	Type	Value
▼ Information Property List	Dictionary	(14 items)
Localization native development re...	String	en
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle creator OS Type code	String	????
Bundle version	String	1
Application requires iPhone enviro...	Boolean	YES
Launch screen interface file base...	String	LaunchScreen
Main storyboard file base name	String	Main
▼ Required device capabilities	Array	(1 item)
Item 0	String	armv7
▶ Supported interface orientati...	Array	(3 items)

Property list - plist

- 심플한 “파일” 저장 방법 중 하나.
- Key,Value구조로 데이터 저장
- File 형태로 저장되다 보니 외부에서 Access가능(보안 취약)

파일 위치

- 파일이 저장되는곳 Bundle & Documents 폴더
- Bundle은 프로젝트에 추가된 Resources가 모인 곳
- 프로그램이 실행되며 저장하는 파일은 Documents폴더에 저장된다.
- **즉! plist파일의 데이터만 불러오는 역할은 Bundle을 통해서, plist파일에 데이터를 쓰고 불러오는 역할은 Documents폴더에 저장된 파일로!**

Plist File In Bundle

1. bundle에 있는 파일 Path 가져오기
2. Path를 통해 객체로 변환, 데이터 불러오기
3. 사용

Bundle

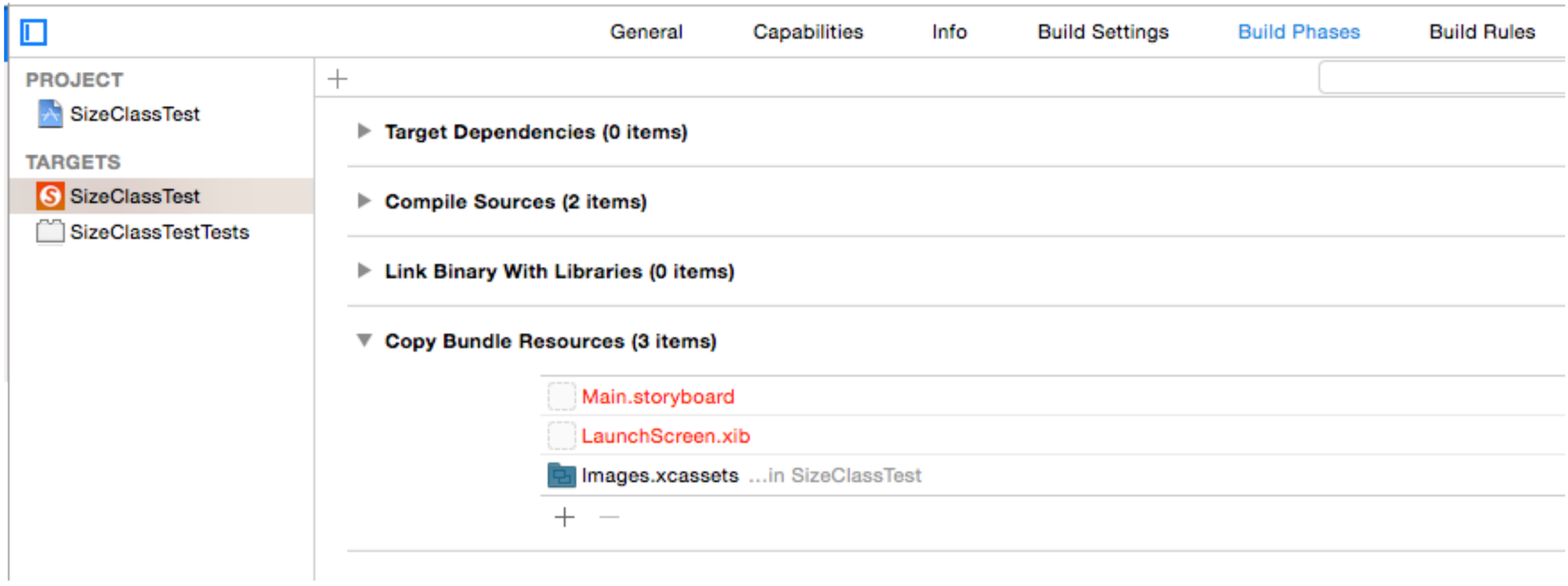
강사 주영민

Fast campus

Bundle

- 실행코드, 이미지, 사운드, nib 파일, 프레임 워크,설정파일 등 코드와 리소스가 모여있는 file system내의 Directory

Bundle 확인



Main Bundle 가져오기

```swift
// Get the main bundle for the app.
let mainBundle = Bundle.main
```

Bundle 파일 주소 가져오기

```
// Get the main bundle for the app.
let mainBundle = Bundle.main
let filePaht = mainBundle.path(forResource: "rName", ofType:
"rType")
```

데이터 불러오기

```
if let path = filePaht {

    let image = UIImage(contentsOfFile: filePaht!)

}
```

Plist File In Bundle

```swift
if let filePaht = mainBundle.path(forResource: "rName", ofType: "rType"),
   let dict = NSDictionary(contentsOfFile: filePaht) as? [String: AnyObject]
{
   // use swift dictionary as normal
}
```

Plist File In Document

1. Document folder Path 찾기

2. Document folder에 plist 파일이 있는지 확인
    - 만약 없다면 : bundle에 있는 파일 Document에 복사

3. Path를 통해 객체로 변환, 데이터 불러오기

4. writeToFile 메소드로 파일 저장

1. 파일 불러오기 (NSSearchPathForDirectoriesInDomains)

```
let path:[String] =
NSSearchPathForDirectoriesInDomains(.documentDirectory, .userDomainMask,
true)
let basePath = path[0] + "/fileName.plist"
```

- doucument 폴더에 Path구하기

2. Document folder에 파일 있는지 확인

```
if !FileManager.default.fileExists(atPath: basePath)
{

}
```

- doucument 폴더에 plist파일이 존재 하지는지 확인

3. bundle에 있는 파일 Document에 복사

```
 if let fileUrl = Bundle.main.path(forResource: "fileName", ofType: "plist")
{
    do {
        try FileManager.default.copyItem(atPath: fileUrl, toPath: basePath)
    } catch  {
        print("fail")
    }
}
```

- 만약 doucument에 해당 plist파일이 존재 하지 않을때, bundle에 있는 파일을 document폴더로 복사

4. Dictionary 인스턴스 불러오기

```swift
if let dict = NSDictionary(contentsOfFile: basePath) as? [String: AnyObject]
{
   // use swift dictionary as normal
}
```

- document 폴더에 있는 파일을 NSDictionary을 통해서 Dictionary인스턴스로 불러오기

5. write(toFile)메소드를 통해 파일 저장

```
if let dict = NSDictionary(contentsOfFile: basePath) as? [String: Any]
{
    var loadData = dict
    loadData.updateValue("addData", forKey: "key")

    let nsDic:NSDictionary =  NSDictionary(dictionary: loadData)
    nsDic.write(toFile: basePath, atomically: true)
}
```

- dictionary를 수정
- NSDictionary로 변경후 writeTofile 메소드를 통해 파일에 저장

실습

- 한번 같이 만들어 볼까요?

Notification

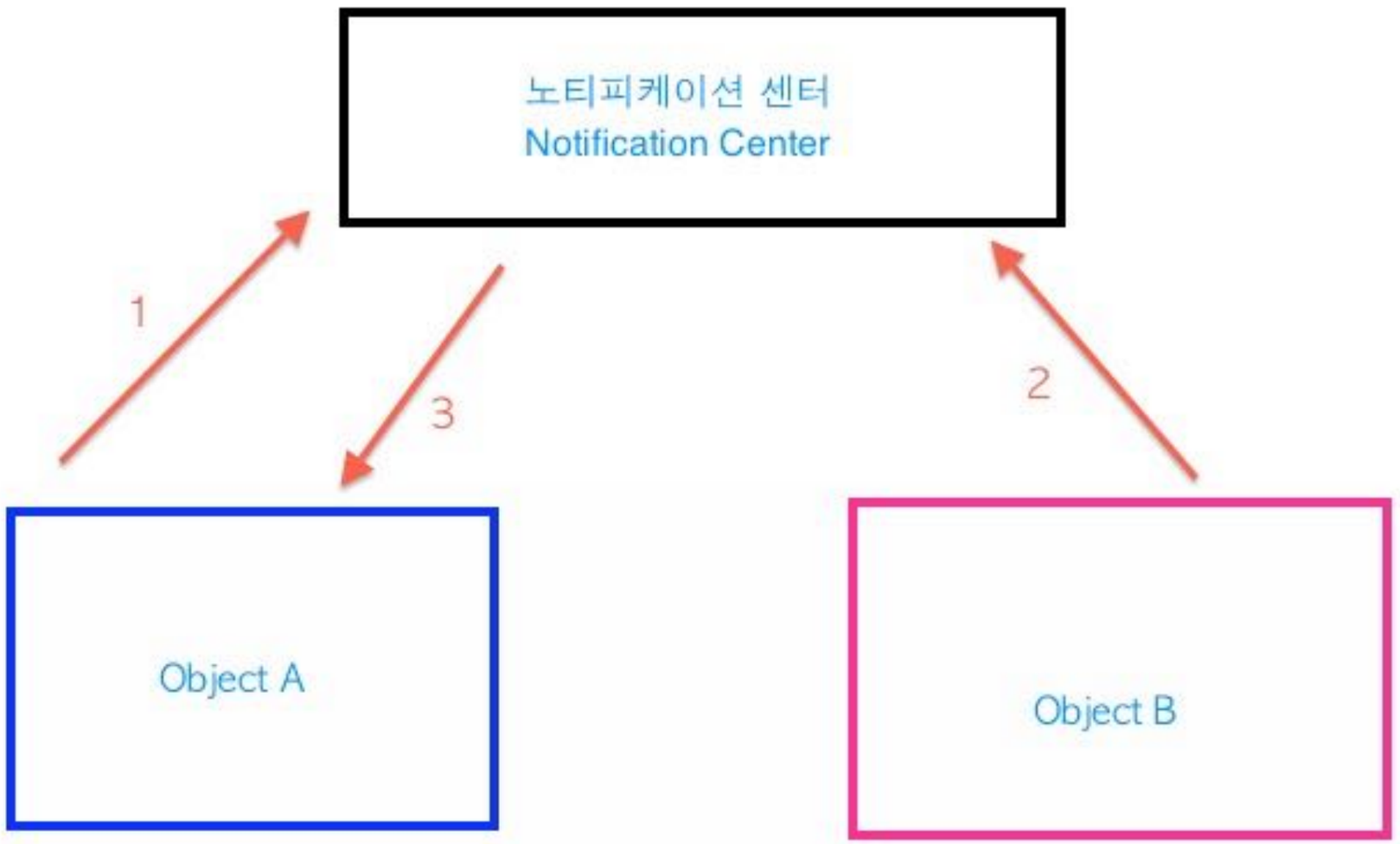
강사 주영민

Fast campus

NotificationCenter

- 특정 이벤트가 발생 하였음을 알리기 위해 불특정 다수의 객체에게 알리기 위해 사용하는 클래스
- 어떤 객체라도 특정 이벤트가 발생했다는 알림을 받을 것이라고 관찰자(Observer)로 등록을 해두면 노티피케이션 센터가 모든 관찰자 객체에게 알림을 준다

Notification 구조



1. 객체A가 노티피케이션 센터에 자신이 노티피게이션을 받을 것이라고 등록.(addObserver)

2. 객체B가 필요한 시점에 노티피케이션 송출 (postNotification)

3. 노티피케이션 센터에서 적절한 객체와 메소드를 찾아 호출

Notification 주요 Method

- Initializing

```
open class var `default`: NotificationCenter { get }
```

- Add Observer

```
open func addObserver(_ observer: Any,
                      selector aSelector: Selector,
                          name aName: NSNotification.Name?,
                        object anObject: Any?)
```

- Post Notification

```
open func post(name aName: NSNotification.Name,
               object anObject: Any?,
           userInfo aUserInfo: [AnyHashable : Any]? = nil)
```

- Remove Observer

```
open func removeObserver(_ observer: Any)
```

예제

Observer

```
func observerNoti(noti:Notification){
    NotificationCenter.default.addObserver(self,
                selector: #selector(ViewController.ob(noti:)),
                    name: NSNotification.Name(rawValue: "key"),
                  object: nil)
}
```

```
func trakingPost(noti:Notification)
{
    //noti 내용
}
```

Poster

```
func postNoti() {
    NotificationCenter.default.post(name:
NSNotification.Name(rawValue: "key"), object: nil)
}
```

- 한번 만들어 볼까요?

System Notification

Observer

```
func observerNoti(noti:Notification){
    NotificationCenter.default.addObserver(self,
                selector: #selector(ViewController.ob(noti:)),
                    name: Notification.Name.UIKeyboardWillShow,
                  object: nil)
}
```

```
func trakingPost(noti:Notification)
{
    //noti 내용
}
```

Poster

키보드가 올라올때 시스템에서 자동으로 Noti를 post해준다.