Swift Study 11

2017. 02.11

Swift 문법 및 ios

- dictionary / enum
- error handle (Error, try/catch, throws)
- UIImagePickerController

Dictionary

- swift에 제공하는 mutable한 collection type 종류
- key:value라는 쌍데이터를 이루어져있으며, 적용할 타입제한이 없다.(단, 일치된 타입 사용)
- key의 타입 경우 해시연산 가능한 타입이어야 함 (식별가능한 키)

<형태>
[key : value, key : value, …]

ex)
[1 : “aaa”, 2 : “bbb”, 3 : “ccc”]

<타입 선언>
Dictionary<key type : value type>

ex)
Dictionary<Int, String>

Dictionary

<선언>
Dictionary<Int, String>()
[Int, String]()

<초기화>
let dic:Dictionary<Int, String> = [1 : “aaa”, 2 : “bbb”, 3 : “ccc”]
let dic = [1 : “aaa”, 2 : “bbb”, 3 : “ccc”]

<추가/수정/삭제>
var dic = [String:String]()

dic[“test1”] = “abc” //test1라는 키이름으로 abc값 추가.
dic.updateValue(“test2”,forKey:”bbb”) //test2키의 값을 bbb로 수정(키값 존재하지 않을시 추가)
dic.updateValue(“test1”,forKey:”bbb”) //test1키의 값을 bbb로 수정
dic.removeValue(forKey: "test1") //test1키의 키와 값을 삭제

Dictionary

- array와 달리 순차적인 저장을 하지 않음 (key값을 기준으로 정렬, 순차적 저장순서X)

<순회탐색>

```
let dictionary = [“test2”:”bbb”, ”test1”:”aaa”]            // [String:String]()

for (key,value) in dictionary {
    print(key, value)
}
```

⇒ 출력

```
test1 bbb
test2 bbb
```

Enum

- 열거형, 특정 주제에 관련 데이터를 멤버로 구성하기 위한 자료형 객체
- 남/여, 국가, 지역 등 구분 지어지는 데이터를 분류할 용도
- enum이라는 키워드로 선언하여 case별 값 카테고리 구분

<형태>

```
enum 열거형 이름 {
   case 멤버값
   case 멤버값
   case 멤버값
   case 멤버값
}
```

ex)

```
enum NATION {
   case korea
   case america
   case japan
   case china
}
```

Enum

<특정값 적용>

```
enum NATION : String {
   case korea = “KR”
   case america = “EN”
   case japan = “JP”
   case china = “CN”
}
```

<사용>

```
NATION.korea
let nation:NATION = NATION.korea
let nation:NATION = .korea
```

<enum 특정값 사용>

```
NATION.korea.rawValue()   // => return “KR”
```

<활용>

```
switch nation{
  case .korea:  print(nation.rawValue())
  case .america: print(nation.rawValue())
  case .japan: print(nation.rawValue())
  case .china: print(nation.rawValue())
}
```

Error handle - Error protocol

- 특정 조건에 대한 에러 제어 흐름을 제어하기 위한 문법적 장치
- 언어에서는 보통 exception (예외처리) 용어로 많이 사용
- Error라는 protocol타입을 구현한 enum타입에 에러를 정의
- Error protocol은 의미없는 빈 프로토콜로 표시의 의미가 강함.

```
public protocol Error {
}
```

<형태>

```
enum 에러명 : Error {
   case 에러함수명
   case 에러함수명(매개변수)
}
```

<형태>

```
enum IntegerParseError : Error{
   case nilNotParsing
   case characterNotParsing(char:Character)
}
```

Error handle - throws / throw

- throws 키워드를 통해 에러 예외처리 호출정의
- throw로 작성한 에러를 호출 (에러를 던지다는 표현함)

```
<형태>
func 함수명(매개변수) throws -> 리턴형 {

    if 조건문 {
        //조건 실행
    } else {
        throw 에러명.에러함수
    }
    ...
}
```

```
ex)
func numCheck(value:Any?) throws ->
Int {
    if let num = value as? Int {
        return num
    } else {
        throw NumCheckError.notNum
    }
  }
```

Error handle - do / try ~ catch

- throw로 던져지는 error를 호출받아 분기시키는 구문
- try로 throws에 대한 결과를 처리하여 실패시 catch로 에러처리

```
do {
    try expression
    statements
} catch pattern 1 {
    statements
} catch pattern 2 where condition {
    statements
}
```

ex)

```
func testNum(_ value:Any){
        do {
            let num = try numCheck(value)
            print(num)
         } catch {
            print((error as! NumCheckError).description)
         }
    }
```

UIImagePickerController

- 사진/동영상 기능을 제공하는 사용자 인터페이스

피커 소스 설정

```
class func availableMediaTypes(for: UIImagePickerControllerSourceType)
```

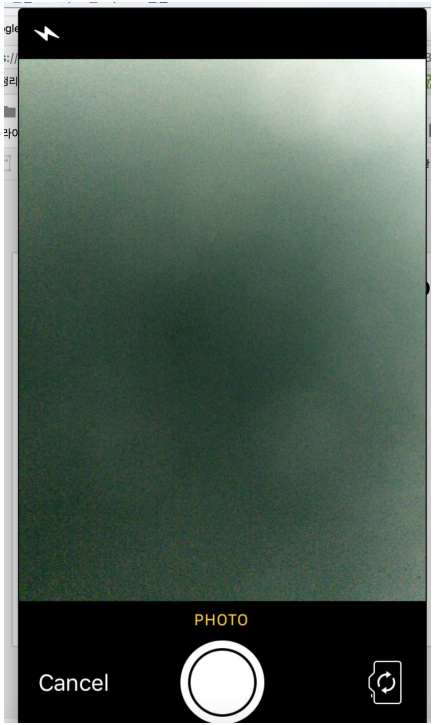
지정된 소스 유형에 사용할 수있는 미디어 유형의 배열을 반환합니다.

```
class func isSourceTypeAvailable(UIImagePickerControllerSourceType)
```

장치가 지정된 소스 유형을 사용하여 미디어를 선택할 수 있는지 여부를 나타내는 부울 값을 반환합니다.

```
var sourceType: UIImagePickerControllerSourceType
```

컨트롤러가 표시 할 피커 인터페이스 유형입니다.



UIImagePickerControllerSourceType

`case photoLibrary`

장치의 사진 라이브러리를 이미지 선택 컨트롤러의 원본으로 지정합니다.

`case camera`

장치의 내장 카메라를 이미지 선택 컨트롤러의 소스로 지정합니다. 사용하여 (사용 가능한 등의 전면 또는 후면) 당신이 원하는 특정 카메라 나타내는 `cameraDevice`속성입니다.

`case savedPhotosAlbum`

장치의 카메라 롤 앨범을 이미지 선택 컨트롤러의 소스로 지정합니다. 장치에 카메라가 없으면 저장된 사진 앨범을 소스로 지정합니다.

UIImagePickerController

선택기 구성

```
var allowsEditing: Bool
```
사용자가 선택한 스틸 이미지 또는 동영상을 편집 할 수 있는지 여부를 나타내는 부울 값입니다.

```
var delegate: (UIImagePickerControllerDelegate & UINavigationControllerDelegate)?
```
이미지 피커의 위임 객체입니다.

```
var mediaTypes: [String]
```
미디어 선택 컨트롤러가 액세스 할 미디어 유형을 나타내는 배열입니다.

비디오 캡처 옵션 구성하기

```
var videoQuality: UIImagePickerControllerQualityType
```
비디오 녹화 및 트랜스 코딩 품질.

```
var videoMaximumDuration: TimeInterval
```
비디오 녹화의 최대 지속 시간 (초).

UIImagePickerController

카메라 컨트롤 사용자 정의하기

```
var showsCameraControls: Bool
```
이미지 선택기에 기본 카메라 컨트롤이 표시되는지 여부를 나타냅니다.

```
var cameraOverlayView: UIView?
```
기본 이미지 선택기 인터페이스 맨 위에 표시 할보기입니다.

```
var cameraViewTransform: CGAffineTransform
```
카메라의 미리보기 이미지에 적용 할 변형입니다.

스틸 이미지 또는 동영상 캡처

```
func takePicture()
```
카메라를 사용하여 정지 이미지를 캡처합니다.

```
func startVideoCapture()
```
에 의해 지정된 카메라를 사용하여 비디오 캡처 시작
`UIImagePickerControllerCameraDevice`속성을.

```
func stopVideoCapture()
```
비디오 캡처를 중단합니다.

UIImagePickerController

카메라 구성

```
var cameraDevice: UIImagePickerControllerCameraDevice
```
이미지 선택 컨트롤러가 사용하는 카메라.

```
class func isCameraDeviceAvailable(UIImagePickerControllerCameraDevice)
```
지정된 카메라가 사용 가능한지 여부를 나타내는 부울 값을 반환합니다.

```
class func availableCaptureModes(for: UIImagePickerControllerCameraDevice)
```
배열 돌려 `NSNumber`소정의 카메라 장치에 의해 지원되는 촬영 모드를 나타내는 개체.

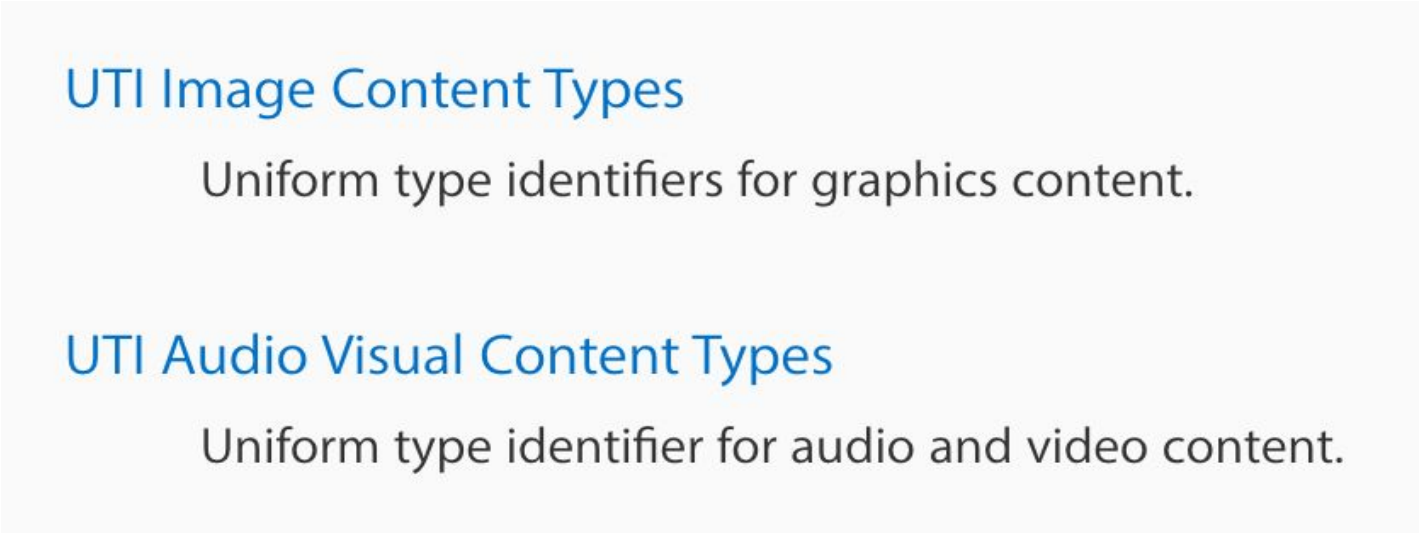
```
var cameraCaptureMode: UIImagePickerControllerCameraCaptureMode
```
카메라가 사용하는 캡처 모드.

```
var cameraFlashMode: UIImagePickerControllerCameraFlashMode
```
활성 카메라에서 사용하는 플래시 모드.

```
class func isFlashAvailable(for: UIImagePickerControllerCameraDevice)
```
지정된 카메라에 플래시 조명 기능이 있는지 여부를 나타냅니다.

UTType

- Uniform Type Identifiers
- 파일 형식이나 메모리 내 데이터 형식, 디렉터리, 볼륨, 패키지와 같은 다른 종류의 엔터티 형식을 설명하는데도 사용
- MobileCoreServices 프레임워크 포함됨



UTI Image Content Types

```
let kUTTypeImage: CFString
```
The abstract type identifier for image data.

```
let kUTTypeJPEG: CFString
```
The type identifier for a JPEG image.

```
let kUTTypeJPEG2000: CFString
```
The type identifier for a JPEG-2000 image.

```
let kUTTypeTIFF: CFString
```
The type identifier for a TIFF image.

```
let kUTTypePICT: CFString
```
The type identifier for a Quickdraw PICT.

```
let kUTTypeGIF: CFString
```

UTI Audio Visual Content Types

```
let kUTTypeMovie: CFString
```
An abstract type identifier for a media format which m
what users would label a "movie"

```
let kUTTypeVideo: CFString
```
An abstract type identifier for pure video data(no audi

```
let kUTTypeAudio: CFString
```
An abstract type identifier for pure audio data (no vide

```
let kUTTypeQuickTimeMovie: CFString
```
The type identifier for a QuickTime movie.

```
let kUTTypeMPEG: CFString
```
The type identifier for a MPEG-1 or MPEG-2 movie.

```
let kUTTypeMPEG4: CFString
```