모바일앱프로그래밍 숙제#1

컴퓨터학과 14학번 김소연

Intro

이름	김소연
학번	2014111517
전공	컴퓨터
주제 선정 이유	지금까지 수업에서 배웠던 많은 기능들을 한 어플에 구현하고 싶었고, 그 과정에서 재미를 느끼고 싶었기 때문에 초성 퀴즈라는 주제를 선정하게 되었습니다. 해당 어플은 초성을 맞출 카테고리와 난이도를 선정할 수 있으며 5문제로 구성되어 있습니다. 어플의 사용자 입장에서, 이동 중이나 쉬는시간같이 일상 속 틈틈히 즐길 수 있도록 구현했습니다.
GitHub ID	aa9390
Project Name	ChosungQuiz (https://github.com/aa9390/chosungQuiz)

Scene #1 : 메인화면 (ViewController.swift)

앱 진입 시 보이는 화면입니다.

Navigation controller

- 네비게이션 컨트롤러를 달아 뷰를 쌓지 않고도 언제든 이전 화면으로 이동할 수 있게 만들었습니다.
- 시작 버튼 - segue를 사용해 이름을 입력할 수 있는 NameViewController로 이동합니다.

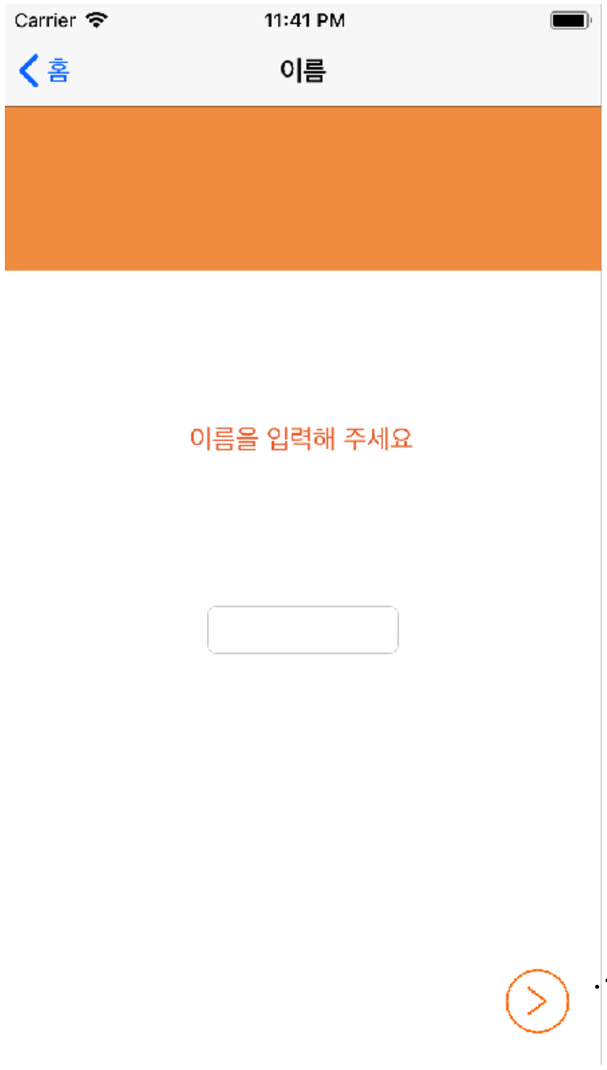
prepare()

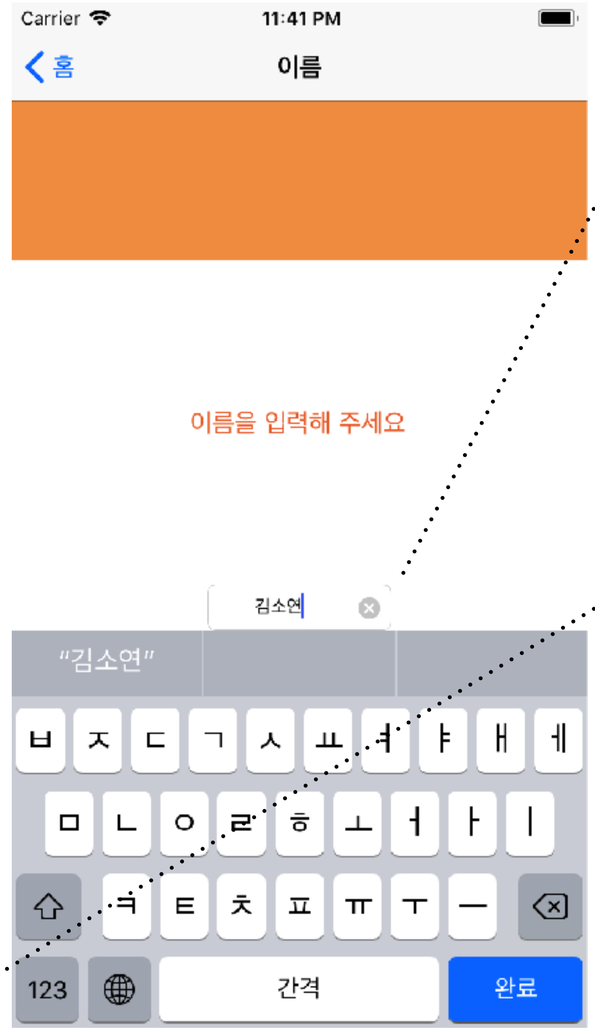
- 시작 버튼과 연결되어 있습니다.
- NameViewController의 타이틀을 “이름”으로 선언합니다.

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "toNameView" {
        let destVC = segue.destination as! NameViewController

        destVC.title = "이름"
    }
}
```

Scene #2 : 이름 입력 화면 (NameVIewController.swift)

어플 사용자의 이름을 입력할 수 있는 화면입니다.





NameTextField

- 사용자의 이름을 입력할 수 있습니다.
- 입력한 이름은 String 변수인 userName에 저장됩니다.
- Return key는 Done이며, 클릭 시 키보드가 사라집니다.

```
@IBOutlet var nameTextField: UITextField!
    func textFieldShouldReturn(_ textField: UITextField) -> Bool
{ textField.resignFirstResponder()
        return true
    }
```

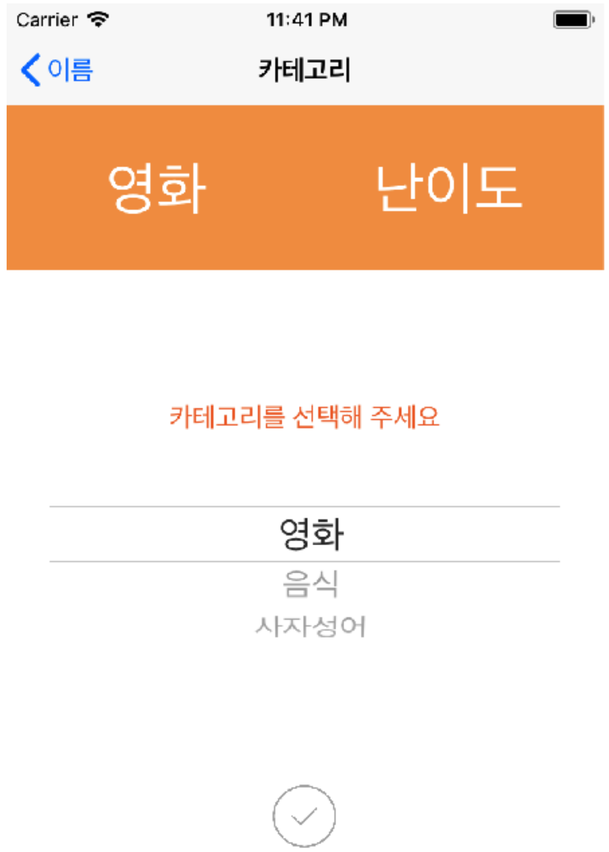
Prepare()

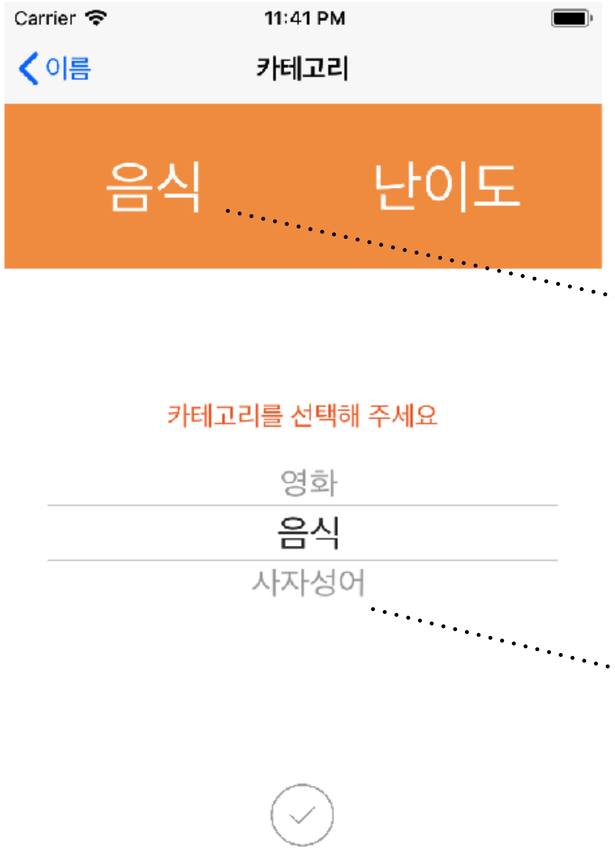
- 주황 화살표 아이콘과 연결되어 있습니다.
- CategoryViewController의 타이틀을 “카테고리”로 선언하며, userName을 동일한 이름의 변수로 넘겨줍니다.

```
    override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
        if segue.identifier == "toCategoryView" {
            let destVC = segue.destination as! CategoryViewController
            let userName: String! = nameTextField.text!

            destVC.title = "카테고리"
            destVC.userName = userName
        }
    }
```

Scene #3 : 카테고리 선택 화면 (CategoryViewController.swift)

카테고리를 입력할 수 있는 화면입니다.





categoryTitle

- okButton 클릭 시, picker view 에서 선택한 카테고리명이 표시됩니다.

```
@IBOutlet var categoryTitle: UILabel!
```

categoryArray

- 카테고리명이 선언된 배열입니다.

```
let categoryArray: [String] = ["영화", "음식", "사자성어"]
```

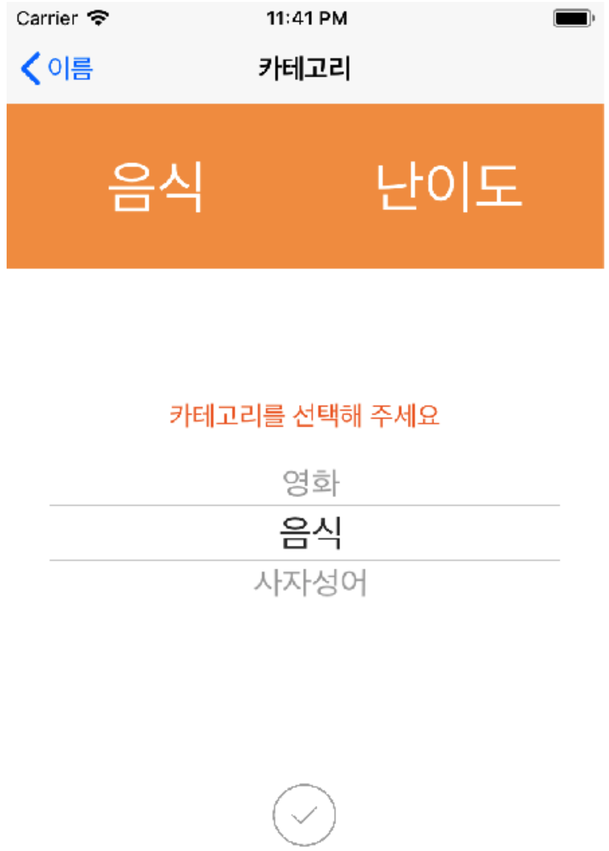
categoryName

- 카테고리명입니다.

```
var categoryName: String!
```

Scene #3 : 카테고리 선택 화면 (CategoryViewController.swift)

카테고리를 입력할 수 있는 화면입니다.



Custom picker View에 관련된 함수

- numberOfComponents() : picker view의 열을 정의합니다. 하나의 열만 쓸 것이기 때문에 1을 리턴합니다.

```
func numberOfComponents(in pickerView: UIPickerView) -> Int {
    return 1
}
```

- Int를 리턴하는 pickerView() :

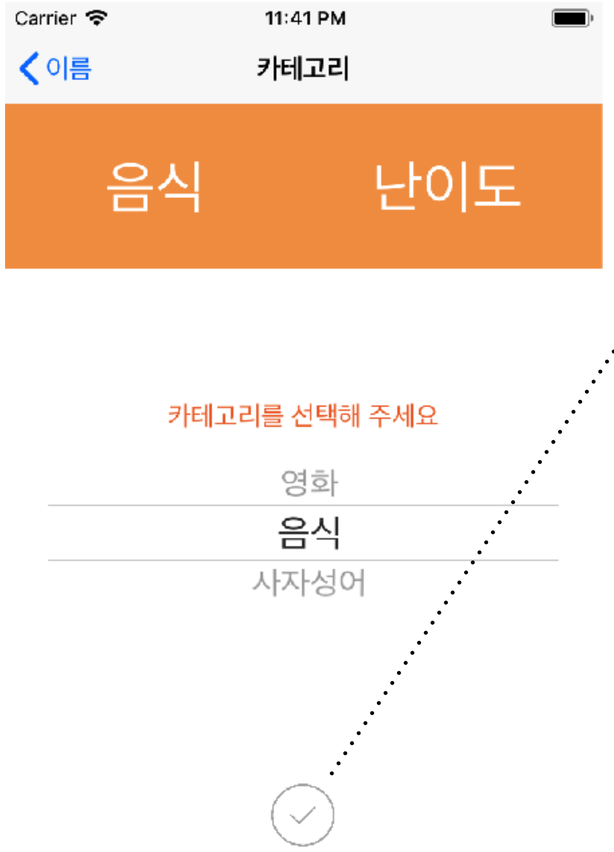
```
func pickerView(_ pickerView: UIPickerView, numberOfRowsInComponent component: Int) -> Int {
        return categoryArray.count
}
```

- String을 리턴하는 pickerView() :

```
func pickerView(_ pickerView: UIPickerView, titleForRow row: Int, forComponent component: Int) -> String? {
        return categoryArray[row]
}
```

Scene #3 : 카테고리 선택 화면 (CategoryViewController.swift)

카테고리를 입력할 수 있는 화면입니다.



okButton

- 카테고리 선택 후 확인 버튼입니다.
- 클릭 시 picker view에서 선택된 카테고리를 categoryName으로 지정하고, categoryTitle에 해당 카테고리명을 표시합니다.

```
@IBOutlet var okButton: UIButton!
@IBAction func okButtonClick(_ sender: Any) {
    categoryName = categoryArray[self.categoryPicker.selectedRow(inComponent: 0)]
    categoryTitle.text = categoryName
}
```

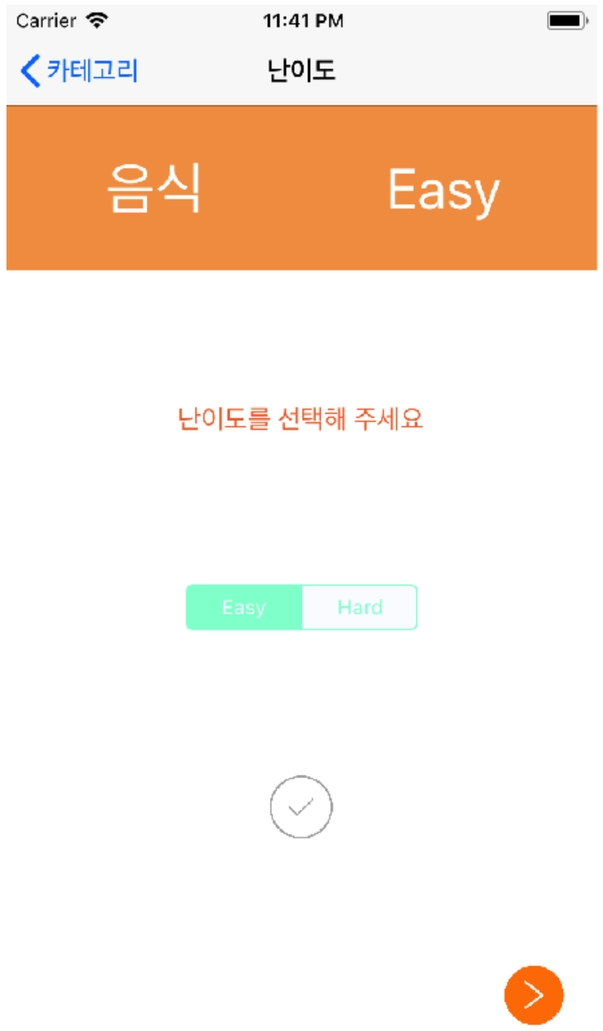
Prepare()

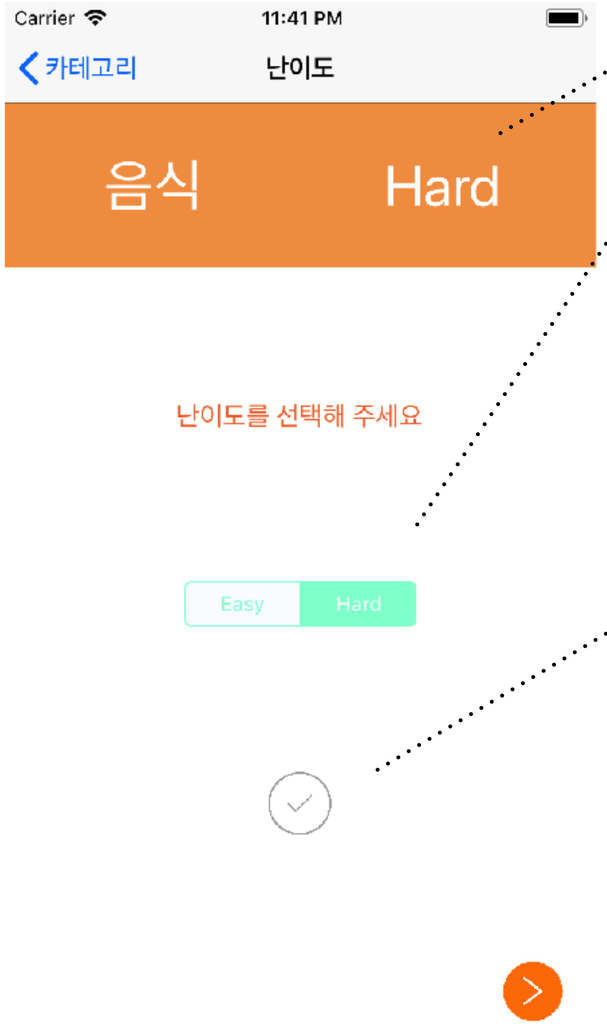
- 주황 화살표 아이콘과 연결되어 있습니다.
- LevelViewController의 타이틀을 “난이도”로 선언하며, userName, categoryName을 동일한 이름의 변수로 넘겨줍니다.

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "toLevelView" {
        let destVC = segue.destination as! LevelViewController
        userName = self.userName!
        categoryName = self.categoryName!

        destVC.title = "난이도"
        destVC.userName = userName
        destVC.categoryName = categoryName
    }
```

Scene #4 : 난이도 선택 화면 (LevelViewController.swift)

난이도를 선택할 수 있는 화면입니다.





levelTitle

- okButton 클릭 시, segmentedControl에서 선택한 난이도가 표시됩니다.

```
@IBOutlet var levelTitle: UILabel!
```

levelSegmentedControl

- 난이도를 선택할 수 있는 segmented Control입니다.
- 선택한 난이도명을 levelName으로 선언합니다.

```
@IBOutlet var levelSegment: UISegmentedControl!
@IBAction func categorySelect(_ sender: UISegmentedControl) {
    levelName = sender.titleForSegment(at: sender.selectedSegmentIndex)
}
```

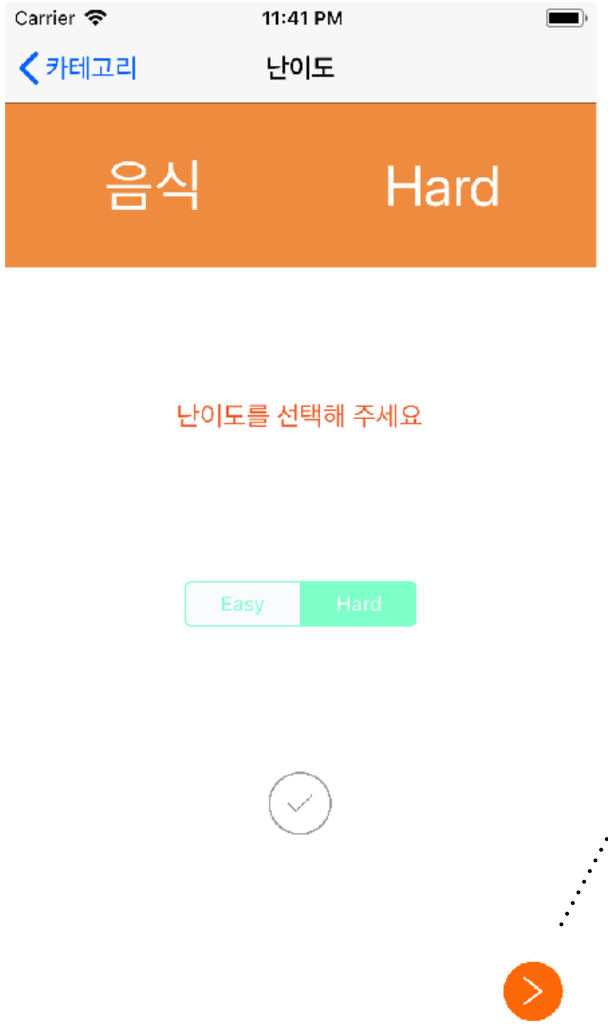
okButton

- 난이도 선택 후 확인 버튼입니다.
- 클릭 시 segmented Control에서 선택된 난이도를 levelName으로 지정하고,
- levelTitle에 해당 난이도명을 표시합니다.

```
@IBOutlet var okButton: UIButton!
    @IBAction func okButtonClick(_ sender: Any) {
        categoryName =
categoryArray[self.categoryPicker.selectedRow(inComponent: 0)]
        categoryTitle.text = categoryName
    }
```

Scene #4 : 난이도 선택 화면 (LevelViewController.swift)

난이도를 선택할 수 있는 화면입니다.



Prepare()

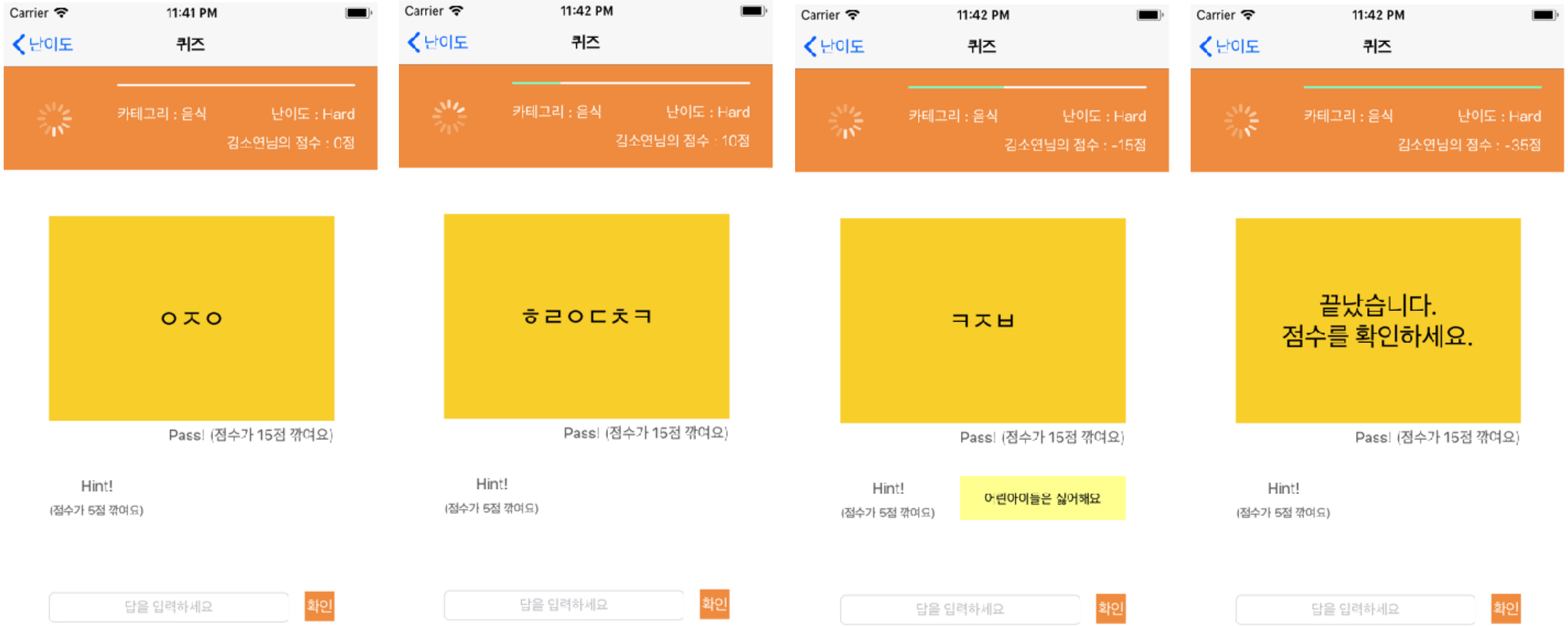
- 주황 화살표 아이콘과 연결되어 있습니다.
- GameViewController의 타이틀을 “난이도”로 선언하며,
- userName, categoryName, levelName을 동일한 이름의 변수로 넘겨줍니다.

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "toLevelView" {
        let destVC = segue.destination as! LevelViewController
        userName = self.userName!
        categoryName = self.categoryName!

        destVC.title = "난이도"
        destVC.userName = userName
        destVC.categoryName = categoryName
    }
}
```

Scene #5 : 게임 화면 (GameViewController.swift)

초성 게임을 진행할 수 있는 화면입니다.

상단에는 **선택한 카테고리, 난이도, 사용자 이름이 표시**되며 **총 5문제**중 진행 상황에 따라 **progress view가 0.2만큼** 움직입니다.
게임 **진행중에는 indicator가 움직이고, 게임이 끝나면 멈춥니다.**
가운데에 제시되는 초성을 보고 하단 입력 필드에 **정답을 입력하면 10점이 증가**하며,
Pass를 누를 경우 15점 감소, hint를 누를 경우 5점이 감소합니다.
게임이 모두 끝난 후에는 점수를 확인하라는 텍스트가 출력됩니다.



Scene #5 : 게임 화면 (GameViewController.swift)

초성 게임을 진행할 수 있는 화면입니다.

```
switch categoryName {
    if levelName == “Easy” {…}
    else levelName == “Hard” {…}
    }
```

chosungArray

answerArray

hintArray

chosungArray, answerArray, hintArray

- 순서대로 초성, 정답, 힌트 배열입니다.
- 5개씩의 항목이 들어 있으며 항목을 새로 추가하거나 삭제해도 정상적으로 작동합니다.

```
var chosungArray: [String] = []
var answerArray: [String] = []
var hintArray: [String] = []
```

viewDidLoad()

- 게임 화면을 초기화합니다.
- 이전 화면에서 받았던 userName, categoryName, levelName을 상단 view에 배치하고,
- 초성 배열의 첫번째 항을 출력합니다.
- 카테고리와 난이도의 선택에 따라 배열의 값을 다르게 줍니다.

Scene #5 : 게임 화면 (GameViewController.swift)

초성 게임을 진행할 수 있는 화면입니다.

switch categoryName {

if levelName == “Easy” {…}
else levelName == “Hard” {…}

}

chosungArray

answerArray

hintArray

```
switch categoryName {
case "영화":
    if(levelName=="Easy") {
        chosungArray = ["ㄷㄷㄷ", "ㅅㅇㅇㅊㅇ", "ㅂㅌㄹ", "ㄱㅇㅇㄱ", "ㅋㄹㅅ"]
        answerArray = ["도둑들", "살인의추억", "베테랑", "겨울왕국", "클래식"]
        hintArray = ["마카오 박을 찾아라!", "향숙이?", "어이가 없네?", "렛잇고", "손예진의 멜로영화"]
    }
    else if (levelName=="Hard") {
        chosungArray = ["ㅅㄹㄴㅇㅇㅈㅈㄷ", "ㅂㅁㅈㅈㅇㅊㅅㄹ", "ㅎㅇㅇㅇㅈㅇㄴㅅ", "ㅌㄹ", "ㅇㅌㅅㅌㄹ"]
        answerArray = ["시라노연애조작단", "백만장자의첫사랑", "하울의움직이는성", "토르", "인터스텔라"]
        hintArray = ["은밀한 연애작전", "이연희, 현빈이 주인공", "미야자키 하야오", "천둥의 신", "Stay..."]
    }
  …
}
```

Scene #5 : 게임 화면 (GameViewController.swift)

초성 게임을 진행할 수 있는 화면입니다.

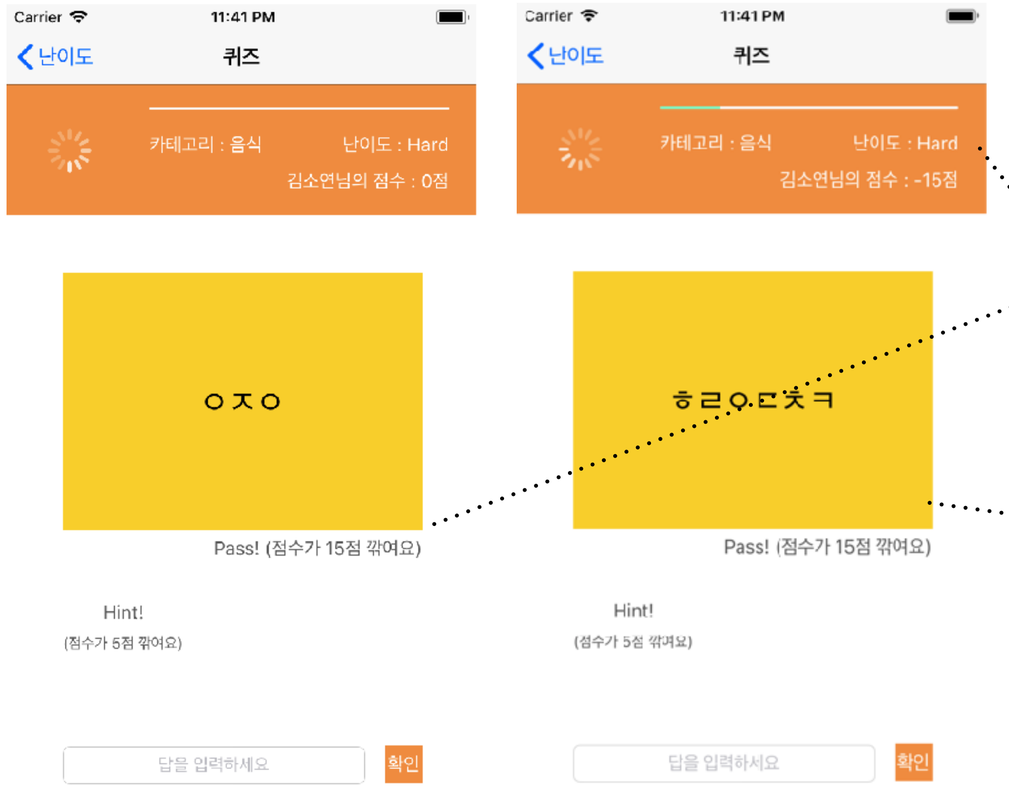
inputTextField

- 정답을 입력할 수 있는 Text Field입니다.
- Return key는 Done이며, 클릭 시 키보드가 사라집니다.

```
@IBOutlet var inputTextField: UITextField!

func textFieldShouldReturn(_ textField: UITextField) -> Bool { textField.resignFirstResponder()
    return true
}
```

Scene #5 : 게임 화면 (GameViewController.swift)

초성 게임을 진행할 수 있는 화면입니다.



buttonClick()

- passButton, hintButton, okButton을 sender로 묶어 게임 구성을 했습니다.

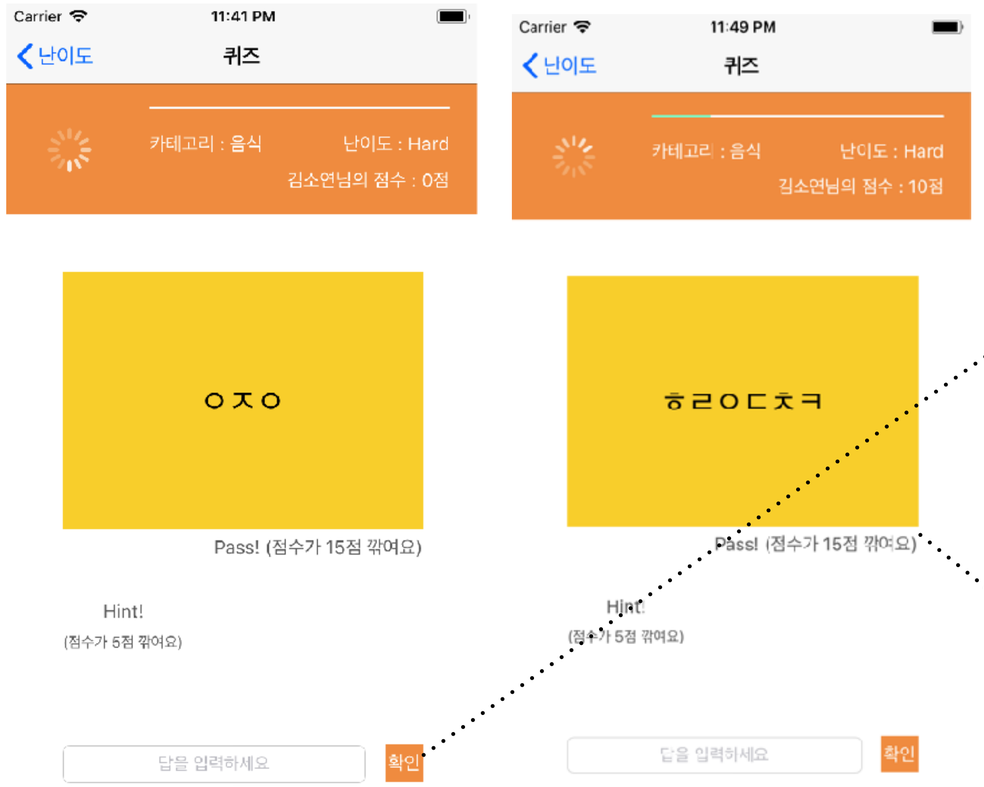
1. 패스 버튼을 눌렀을 때

- 다음 초성을 출력하며,
- 더 이상 출력할 초성이 없을 경우에는 게임이 끝났음을 알립니다.

```
if (sender==passButton) {
    hintView.isHidden = true
    score = score - 15
    scoreString! = String(score!)
    infoLabel.text = userName + "님의 점수 : " + scoreString + "점"
    progressBar.progress += 0.2

    // 다음 초성으로 이동
    i = i + 1
    if(i<answerArray.count) {
        chosungLabel.text = chosungArray[i]
    }
    else {
        chosungLabel.text = "끝났습니다. \n점수를 확인하세요."
        indicatorView.stopAnimating()
    }
}
```

Scene #5 : 게임 화면 (GameViewController.swift)

초성 게임을 진행할 수 있는 화면입니다.



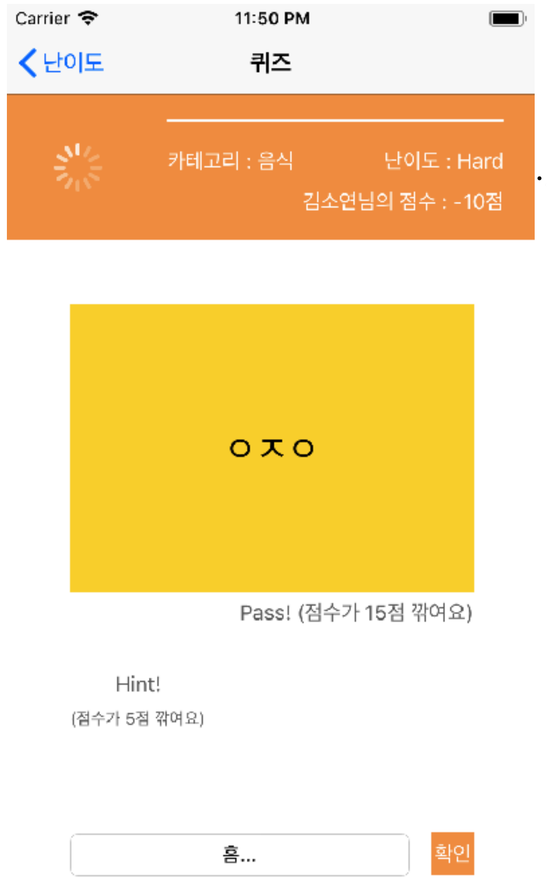
2. 정답 버튼을 눌렀을 때

- 정답이 맞았을 경우에는 다음 초성을 출력하며,
- 더 이상 출력할 초성이 없을 경우에는 게임이 끝났음을 알립니다.

```
- if (sender==okButton) {
      hintView.isHidden = true

      // 정답을 입력하지 않았을 때
      if (inputTextField.text == "") {}

      // 정답이 맞았을 때
      else if(inputTextField.text == answerArray[i]) {
          score = score + 10
          scoreString! = String(score!)
          inputTextField.text = ""
          progressBar.progress += 0.2
          infoLabel.text = userName + "님의 점수 : " + scoreString + "점"
          hintView.isHidden = true

          // 다음 초성으로 이동
          i = i + 1
          if(i<answerArray.count) {
              chosungLabel.text = chosungArray[i]
          }
```

Scene #5 : 게임 화면 (GameViewController.swift)

초성 게임을 진행할 수 있는 화면입니다.



2. 정답 버튼을 눌렀을 때

- 정답이 맞았을 경우에는 다음 초성을 출력하며,
- 더 이상 출력할 초성이 없을 경우에는 게임이 끝났음을 알립니다.

-

```
// 정답이 틀렸을 때
else if (inputTextField.text != answerArray[i]){
    score = score - 10
    scoreString! = String(score!)
    infoLabel.text = userName + "님의 점수 : " + scoreString + "점"
}
```

Scene #5 : 게임 화면 (GameViewController.swift)

초성 게임을 진행할 수 있는 화면입니다.

3. 힌트 버튼을 눌렀을 때

- 힌트 뷰가 출력됩니다.

```
if (sender==hintButton) {
    if(hintView.isHidden) {
        hintView.isHidden = false
        score = score - 5
        scoreString! = String(score!)
        hintLabel.text = hintArray[i]
        infoLabel.text = userName + "님의 점수 : " + scoreString + "점"
    }
}
```