lecture 9

mid semester review

cs198-001 : fall 2017

announcements

- Snapchat Clone Part 2 due next Monday
- next lab: Snapchat Camera
- future labs all project work days

rest of the semester

Mon 10/30 Wed 11/1	No class	Mid-Semester Review	
Mon 11/6 Wed 11/8	AVFoundation, Location, Mapkit	Snapchat Camera	Snapchat Clone Part 2 due 11/6
Mon 11/13 Wed 11/15	Advanced Swift and Objective C	Project Work Day	
Mon 11/20 Wed 11/22	UI/UX and Programmatic Design	No lab	
Mon 11/27 Wed 11/29	ARKit and Core ML	Project Work Day	
Wed 12/6 Fri 12/8	Custom app due Wed 12/6 at noon	Final Presentation **Attendance required**	Custom App due

today's lecture

mid-semester review

we'll ask some questions, talk to the people next to you

Swift

Syntax

This code doesn’t compile. What’s the problem?

```
func foo(with bar: Double) -> Int {
    return Int(bar + 198.001)
}

let x = foo(bar: 5.0)
```

Syntax

This code doesn’t compile. What’s the problem?

```
func foo(with bar: Double) -> Int {
    return Int(bar + 198.001)
}

let x = foo(with: 5.0)
```

bar is an internal parameter

! vs ?

What is the difference between types ending with a “!” and a “?”

! vs ?

What is the difference between types ending with a “!” and a “?”

? - can take on a value of `nil`

! - cannot take on value of `nil`

Classes versus structs

Question: What's the main difference between a Class and a Struct?

Classes versus structs

Question: What's the main difference between a Class and a Struct?

Classes - Pass by reference
Structs - Pass by value

Value Types

Question: What are some other value types?

Value Types

Question: What are some other value types?

Structs
Enums
Arrays
String
Dictionary
...

Reference Types

Question: What about reference types?

Reference Types

Question: What about reference types?

Classes

Closures

Syntax

Will this compile?

```
var threads: [String: [Post]] = ["Memes": []]
let p = Post(username: "Oski", read: false)

let memeThread = threads["Memes"]!
memeThread.append(p)

print(threads["Memes"]!.count)
```

Syntax

Will this compile?

```
var threads: [String: [Post]] = ["Memes": []]
let p = Post(username: "Oski", read: false)

let memeThread = threads["Memes"]!
memeThread.append(p)

print(threads["Memes"]!.count)
```

Nope

Syntax

What does this print?

```
var threads: [String: [Post]] = ["Memes": []]
let p = Post(username: "Oski", read: false)

var memeThread = threads["Memes"]!
memeThread.append(p)

print(threads["Memes"]!.count)
```

Syntax

What does this print?

```
var threads: [String: [Post]] = ["Memes": []]
let p = Post(username: "Oski", read: false)

var memeThread = threads["Memes"]!
memeThread.append(p)

print(threads["Memes"]!.count)
```

0

Syntax

What does this print?

```
var threads: [String: [Post]] = ["Memes": []]
let p = Post(username: "Oski", read: false)
threads["Memes"]!.append(p)
print(threads["Memes"]!.count)
```

1

Syntax

And this? (`Post` is a `struct`)

```
var threads: [String: [Post]] = ["Memes": []]

let p = Post(username: "Oski", read: false)

threads["Memes"]!.append(p)

p.username = "Osky"

print(threads["Memes"]!.first!.username)
```

Syntax

And this? (`Post` is a `struct`)

```
var threads: [String: [Post]] = ["Memes": []]

var p = Post(username: "Oski", read: false)

threads["Memes"]!.append(p)

p.username = "Osky"

print(threads["Memes"]!.first!.username)
```

Syntax

And this? (`Post` is a `struct`)

```
var threads: [String: [Post]] = ["Memes": []]

var p = Post(username: "Oski", read: false)

threads["Memes"]!.append(p)

p.username = "Osky"

print(threads["Memes"]!.first!.username)
```

Oski

Syntax

And what if `Post` is a `class`?

```
var threads: [String: [Post]] = ["Memes": []]

var p = Post(username: "Oski", read: false)

threads["Memes"]!.append(p)

p.username = "Osky"

print(threads["Memes"]!.first!.username)
```

Syntax

And what if `Post` is a `class`?

```
var threads: [String: [Post]] = ["Memes": []]

var p = Post(username: "Oski", read: false)

threads["Memes"]!.append(p)

p.username = "Osky"

print(threads["Memes"]!.first!.username)
```

Osky

Syntax

Finally, what is socially unacceptable?

```
var threads: [String: [Post]] = ["Memes": []]

var p = Post(username: "Oski", read: false)

threads["Memes"]!.append(p)

p.username = "Osky"

print(threads["Memes"]!.first!.username)
```

Syntax

Finally, what is socially unacceptable?

```
var threads: [String: [Post]] = ["Memes": []]
var p = Post(username: "Oski", read: false)
threads["Memes"]!.append(p)
p.username = "Osky"
print(threads["Memes"]!.first!.username)
```

force unwrapping optional variables

Variables

```
import UIKit

var view1 = UIView()
view1.alpha = 0.5

let view2 = UIView()
view2.alpha = 0.5 // will this line compile?
```

Question: Will the last line compile? Why or why not?

credit: Antonio Bello

Variables

```
import UIKit

var view1 = UIView()
view1.alpha = 0.5

let view2 = UIView()
view2.alpha = 0.5 // will this line compile?
```

Question: Will the last line compile? Why or why not? Yes it will. Even though view2 is declared with let, it's properties are mutable

credit: Antonio Bello

Storyboard

Prepare for Segue

Question: Can I do this?

```
override func prepare(for segue: UIStoryboardSegue,
sender: Any?) {

    if let dest = segue.destination as?
ChooseThreadViewController {
        dest.chosenImageView.image = selectedImage
    }

}
```

Prepare for Segue

Question: Can I do this?

```
override func prepare(for segue: UIStoryboardSegue,
sender: Any?) {

    if let dest = segue.destination as?
ChooseThreadViewController {
        dest.chosenImageView.image = selectedImage
    }

}
```

No! Why?

Prepare for Segue

Question: Is this better?

```
override func prepare(for segue: UIStoryboardSegue,
sender: Any?) {
    if let dest = segue.destination as?
ChooseThreadViewController {
        dest.chosenImage = selectedImage
    }
}


...
override func viewDidLoad() {
    super.viewDidLoad()
    chosenImageView.image = chosenImage
}
```

Prepare for Segue

Question: Is this better?

```
override func prepare(for segue: UIStoryboardSegue,
sender: Any?)
    if let destination
ChooseTableViewController {
        destinationImage = chosenImage
    }
}

...
override func viewDidLoad() {
    super.viewDidLoad()
    chosenImageView.image = chosenImage
}
```

YES!

View Controller Lifecycle

Question: What's the difference between the methods `viewDidLoad` and `viewWillAppear`?

View Controller Lifecycle

Question: What's the difference between the methods `viewDidLoad` and `viewWillAppear`?

`viewDidLoad` - called once when the view controller is created

`viewWillAppear` - called every time the view appears on the screen

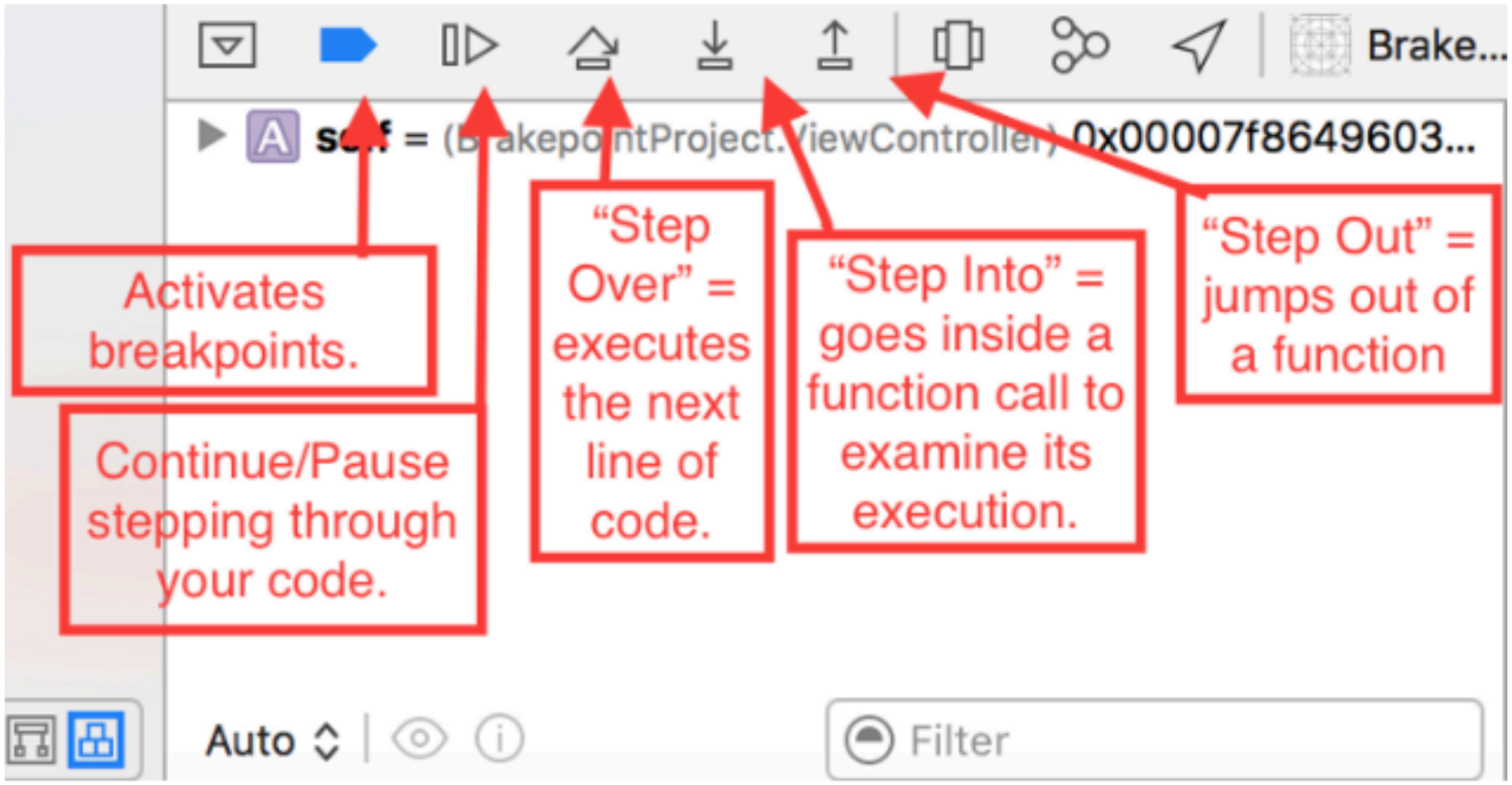
Xcode

breakpoints

Question: What do each of the following buttons do?

breakpoints

Question: What do each of the following buttons do?



Auto Layout

Autolayout?

Question: What are some key benefits of using AutoLayout over frame-based layout?

Autolayout?

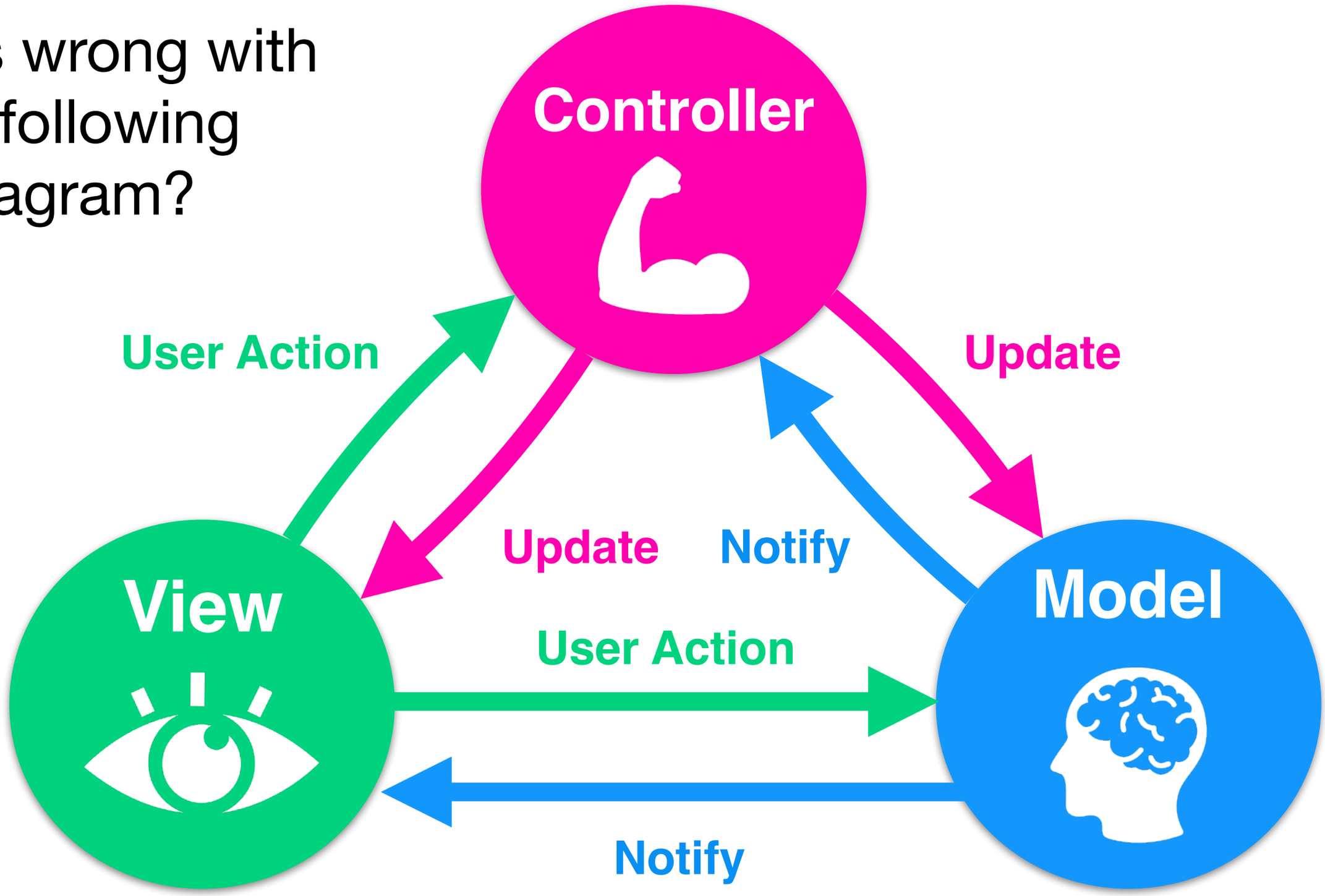
Question: What are some key benefits of using AutoLayout over frame-based layout?

- ☆ constraints automatically adapt to different screen sizes and orientations, reducing the amount of code you need to write
- ☆ constraints can be set in Storyboard
- ☆ relational rather than calculation-based (could be a con, depending on what you find

MVC

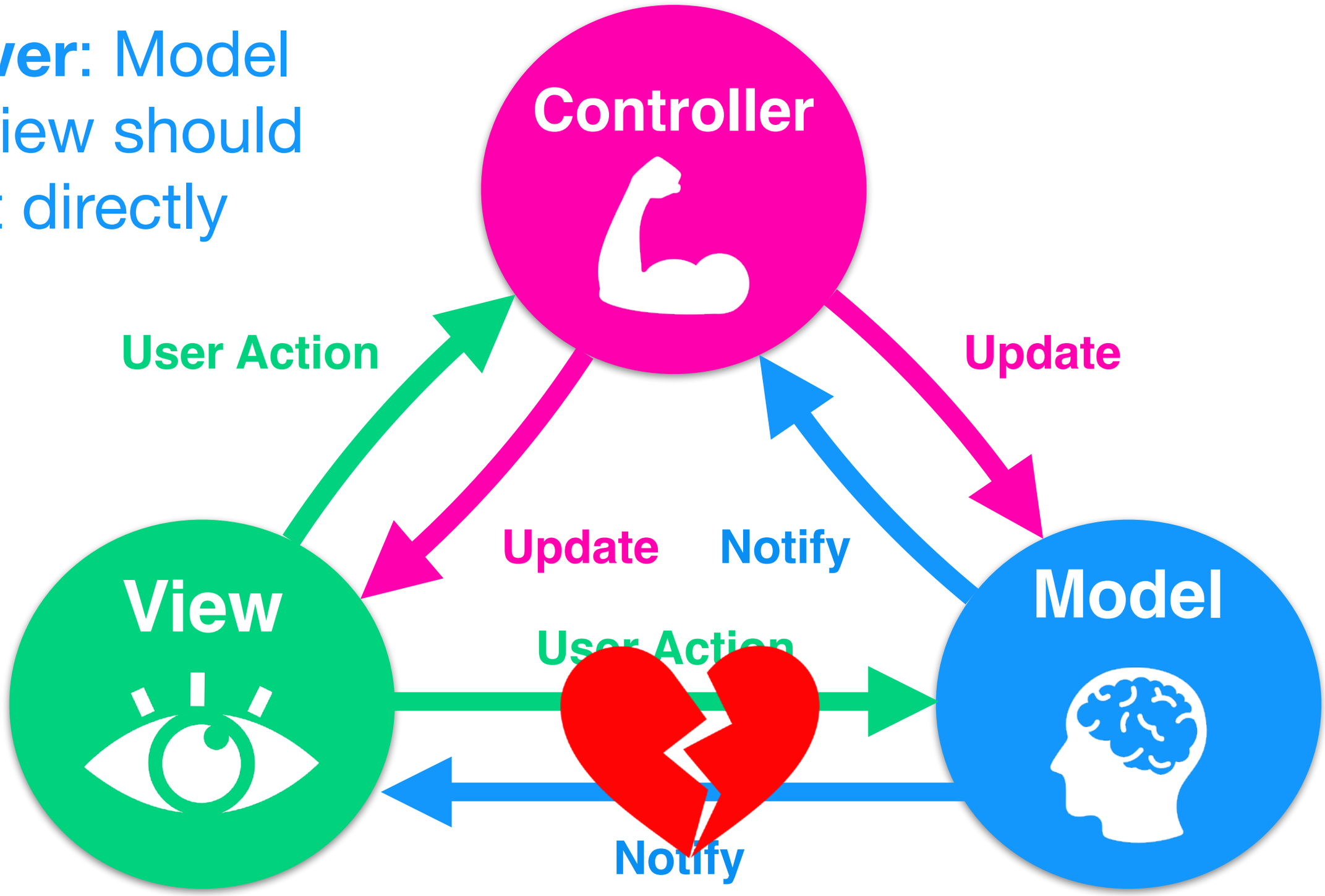
MVC design

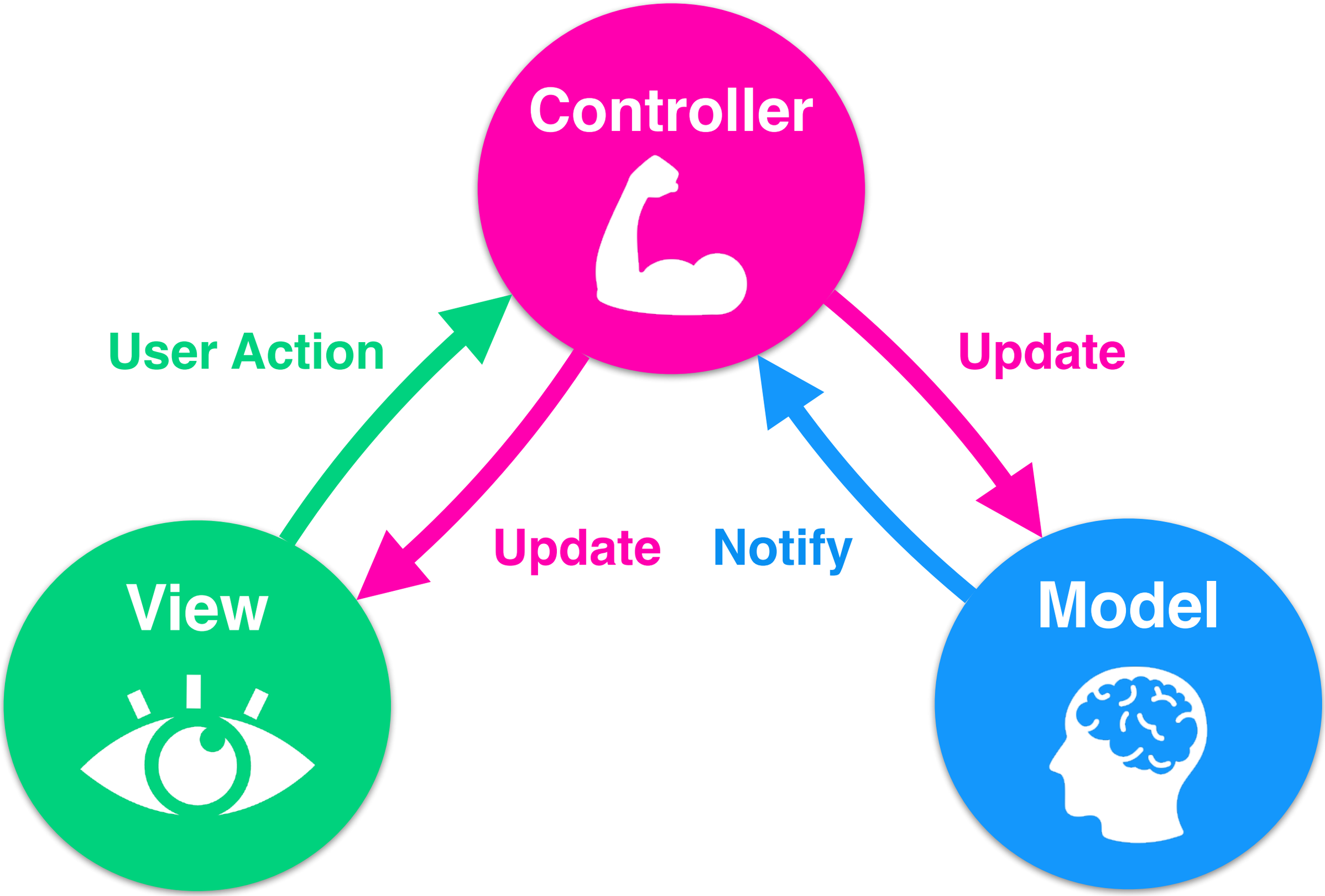
What's wrong with the following diagram?



MVC design

Answer: Model and View should not directly





Model View Controller

MVC q2

```
var threads: [String: [UIImage]] = ["memes": [], "dog
spots": [], "random": []]

var read: [String: [Bool]] = ["memes": [], "dog spots":
[], "random": []]

var postTime: [String: [NSDate]] = ["memes": [], "dog
spots": [], "random": []]

var username: [String: [String]] = ["memes": [], "dog
spots": [], "random": []]
```

Question: This is one implementation of the Model for the Snapchat project. What's a better way to implement this? (hint: how can we make this object oriented)

MVC q2

```
class Post {
    var read: Bool = false
    let username: String
    let thread: String
    let datePosted: Date = NSDate()
    let imageName: String
    let postId: String
    ///implementation hidden
}

var threads: [String: [Post]] = ["memes": [], "dog
                                 spots": [], "random": []]
```

Question: This is one implementation of the Model for the Snapchat project. What's a better way to implement this?

Create a Post / Snap Model class! See example Post class here: github.com/iosdecal/hw3-part2/blob/master/SnapchatClone/

Collection views and table views

Collection/table views

What are the 2 required methods for Tableviews?

Collection/table views

What are the 2 required methods for Tableviews?

```
func cellForRow(at indexPath:
IndexPath) -> UITableViewCell?

func numberOfRows(inSection
section: Int) -> Int
```

```
class ImagePickerController: UIViewController,
UICollectionViewDataSource {

    @IBOutlet var imageCollectionView: UICollectionView!

    override func viewDidLoad() {
        super.viewDidLoad()
        imageCollectionView.dataSource = self
    }

    func collectionView(_ collectionView: UICollectionView,
         numberOfItemsInSection section: Int) -> Int {
        // implementation hidden
    }

    func collectionView(_ collectionView: UICollectionView,
       cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {
        // implementation hidden
    }

    func collectionView(_ collectionView: UICollectionView,
         didSelectItemAt indexPath: IndexPath) {
        // implementation hidden
    }
}
```

My collectionview isn’t behaving correctly. What problem do you think I have, and how could I solve it?

```
class ImagePickerController: UIViewController,
UICollectionViewDataSource, UICollectionViewDelegate {

    @IBOutlet var imageCollectionView: UICollectionView!

    override func viewDidLoad() {
        super.viewDidLoad()
        imageCollectionView.dataSource = self
        imageCollectionView.delegate = self
    }

    func collectionView(_ collectionView: UICollectionView,
         numberOfItemsInSection section: Int) -> Int {
        // implementation hidden
    }

    func collectionView(_ collectionView: UICollectionView,
       cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {
        // implementation hidden
    }

    func collectionView(_ collectionView: UICollectionView,
          didSelectItemAt indexPath: IndexPath) {
        // implementation hidden
    }
}
```

`didSelectItemAt` will never be called, since the delegate is not set

```
class ImagePickerController: UIViewController {

    @IBOutlet var imageCollectionView: UICollectionView!

    override func viewDidLoad() {
        super.viewDidLoad()
        imageCollectionView.dataSource = self
    }
}

class ImagePickerDataSource: NSObject,
                        UICollectionViewDataSource {
    func collectionView(_ collectionView: UICollectionView,
          numberOfItemsInSection section: Int) -> Int {
        // implementation hidden
    }

    func collectionView(_ collectionView: UICollectionView,
            cellForItemAt indexPath: IndexPath) ->
                  UICollectionViewCell {
        // implementation hidden
    }
}
```

What's wrong with my code? How can I fix it (multiple answers)

```
class ImagePickerController: UIViewController {

    @IBOutlet var imageCollectionView: UICollectionView!

    override func viewDidLoad() {
        super.viewDidLoad()
        imageCollectionView.dataSource = self
    }
}

class ImagePickerDataSource: NSObject,
                             UICollectionViewDataSource {
    func collectionView(_ collectionView: UICollectionView,
          numberOfItemsInSection section: Int) -> Int {
        // implementation hidden
    }

    func collectionView(_ collectionView: UICollectionView,
            cellForItemAt indexPath: IndexPath) ->
                    UICollectionViewCell {
        // implementation hidden
    }
}
```

We are setting the dataSource to self, but ImagePickerController does not conform to `UICollectionViewDataSource`

```
class ImagePickerController: UIViewController {

    @IBOutlet var imageCollectionView: UICollectionView!

    override func viewDidLoad() {
        super.viewDidLoad()
        imageCollectionView.dataSource = ImagePickerDataSource()
    }
}

class ImagePickerDataSource: NSObject,
                        UICollectionViewDataSource {
    func collectionView(_ collectionView: UICollectionView,
          numberOfItemsInSection section: Int) -> Int {
        // implementation hidden
    }

    func collectionView(_ collectionView: UICollectionView,
            cellForItemAt indexPath: IndexPath) ->
                    UICollectionViewCell {
        // implementation hidden
    }
}
```

Solution 1: create instance of ImagePickerDataSource()
and use it as the datasource

```
class ImagePickerController: UIViewController,
                             UICollectionViewDataSource {

    @IBOutlet var imageCollectionView: UICollectionView!

    override func viewDidLoad() {
        super.viewDidLoad()
        imageCollectionView.dataSource = self
    }

    func collectionView(_ collectionView: UICollectionView,
          numberOfItemsInSection section: Int) -> Int {
        // implementation hidden
    }

    func collectionView(_ collectionView: UICollectionView,
            cellForItemAt indexPath: IndexPath) ->
                    UICollectionViewCell {
        // implementation hidden
    }
}
```

Solution 2: make your ImagePickerController conform to UICollectionViewDataSource, and move code into it

```
class ImagePickerController: UIViewController {

    @IBOutlet var imageCollectionView: UICollectionView!

    override func viewDidLoad() {
        super.viewDidLoad()
        imageCollectionView.dataSource = self
    }
}

extension ImagePickerController: UICollectionViewDataSource {

    var cellImages = [UIImage(named: "dog"), UIImage(named: "dog2")]

    func collectionView(_ collectionView: UICollectionView,
        numberOfItemsInSection section: Int) -> Int {
        return cellImages.count
    }

    func collectionView(_ collectionView: UICollectionView,
       cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {
        // implementation hidden
    }
}
```

What's wrong with my code? How can I fix it (multiple answers)

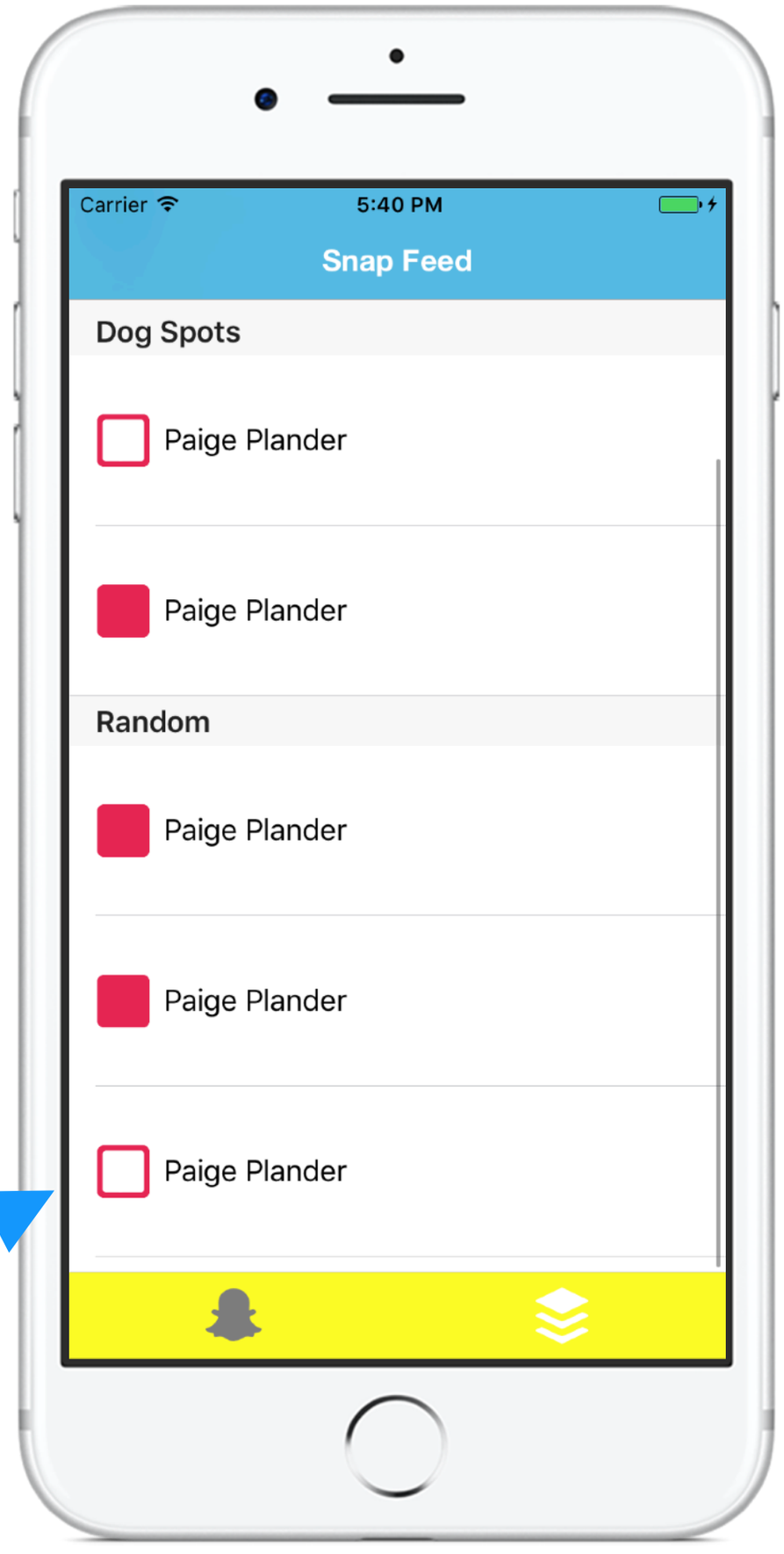
```
class ImagePickerController: UIViewController {

    var cellImages = [UIImage(named: "dog"), UIImage(named: "dog2")]
    @IBOutlet var imageCollectionView: UICollectionView!

    override func viewDidLoad() {
        super.viewDidLoad()
        imageCollectionView.dataSource = self
    }
}

extension ImagePickerController: UICollectionViewDataSource {

    func collectionView(_ collectionView: UICollectionView,
        numberOfItemsInSection section: Int) -> Int {
        return cellImages.count
    }

    func collectionView(_ collectionView: UICollectionView,
       cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {
        // implementation hidden
    }
}
```

extensions can’t store properties - to fix, move it into the main class

Collection/table views

Question: Some of my cells show up as read, even though they shouldn't be - what's the problem in my

Wasn't read, but still clickable!



Collection/table views

Question: Some of my cells show up as read, even though they shouldn't be - what's the problem in my code?

```
func tableView(_ tableView: UITableView, cellForRowAt
    indexPath: IndexPath) -> UITableViewCell {
    let cell =
        tableView.dequeueReusableCell(withIdentifier:
        "postCell", for: indexPath) as! PostsTableViewCell
    if let post = getPostFromIndexPath(indexPath:
        indexPath) {
        if post.read {
            cell.readImageView.image = UIImage(named:
                "read")
        }
    }
    return cell
}
```

Collection/table views

Since Table view Cells are **recycled** , you need to check if the cell has is not read, and set image to “unread”

```
func tableView(_ tableView: UITableView, cellForRowAt
    indexPath: IndexPath) -> UITableViewCell {
    let cell =
        tableView.dequeueReusableCell(withIdentifier:
        "postCell", for: indexPath) as! PostsTableViewCell
    if let post = getPostFromIndexPath(indexPath:
        indexPath) {
        if post.read {
            cell.readImageView.image = UIImage(named:
                "read")
        }
    }
    return cell
}
```

Collection/table views

```
func tableView(_ tableView: UITableView, cellForRowAt indexPath:
    IndexPath) -> UITableViewCell {
    let cell = tableView.dequeueReusableCell(withIdentifier:
        "postCell", for: indexPath) as! PostsTableViewCell
    if let post = getPostFromIndexPath(indexPath: indexPath) {
        if post.read {
            cell.readImageView.image = UIImage(named: "read")
        }
        else {
            cell.readImageView.image = UIImage(named: "unread")
        }
        cell.usernameLabel.text = post.username
    }
    return cell
}
```

Solution Code

Networking

Async vs Sync q1

Question: What's the difference between an asynchronous block of code and a synchronous block of code?

Async vs Sync q1

Question: What's the difference between an asynchronous block of code and a synchronous block of code?

Synchronous: waits until the task has completed
Asynchronous: completes a task in background and can notify you when complete

Async vs Sync q2

Question: Give an example of a scenario in which we would need to use a callback?

Async vs Sync q2 (ans)

API call

```
let client = TWTRAPIClient()
client.loadTweetWithID("20") { (tweet, error) -> Void in
    // handle the response or error
}
```

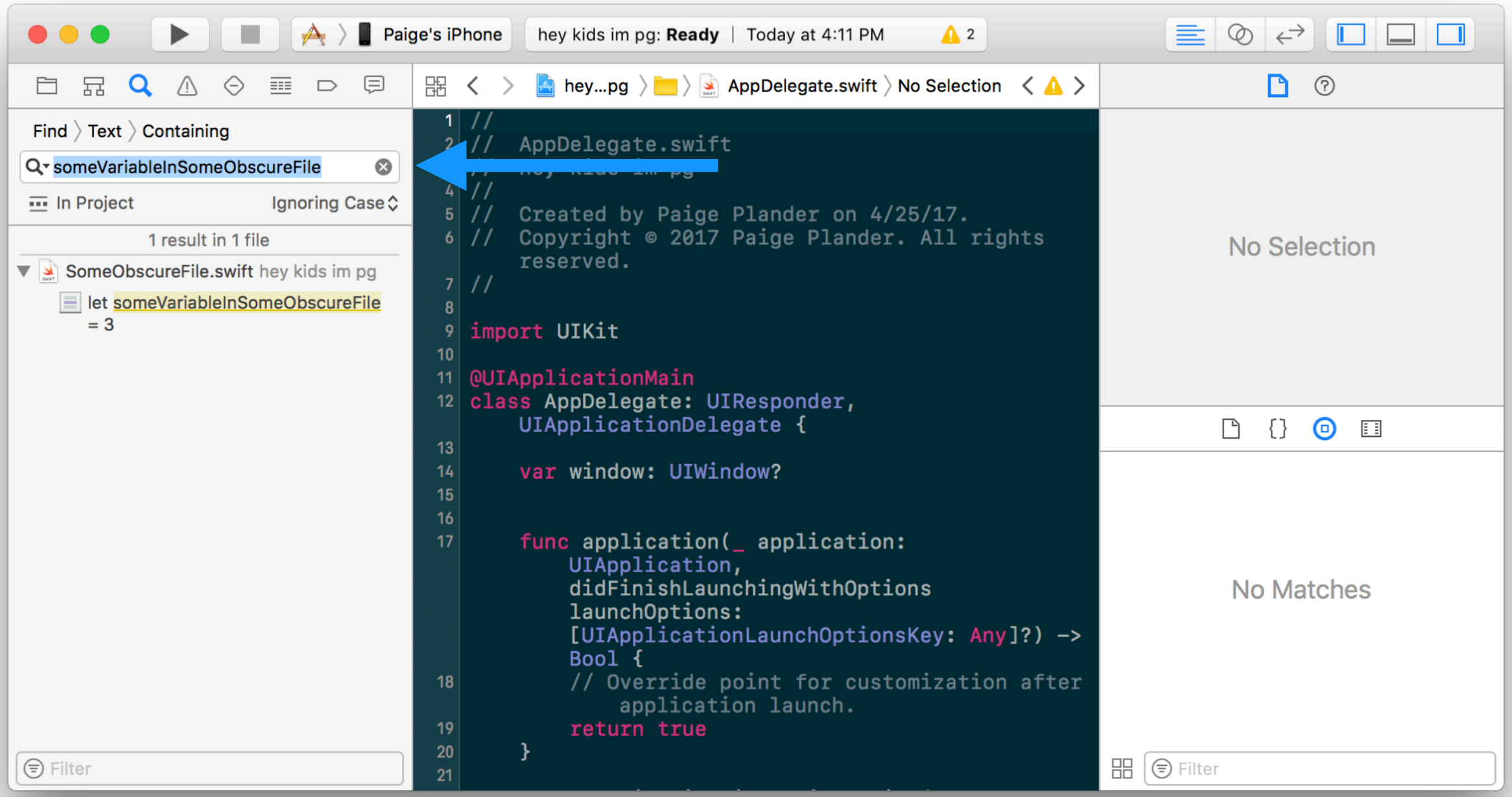
URLSession

```
func dataTask(with url: URL,
              completionHandler: @escaping (Data?,
                    URLResponse?, Error?) -> Void) ->
                    URLSessionDataTask
```

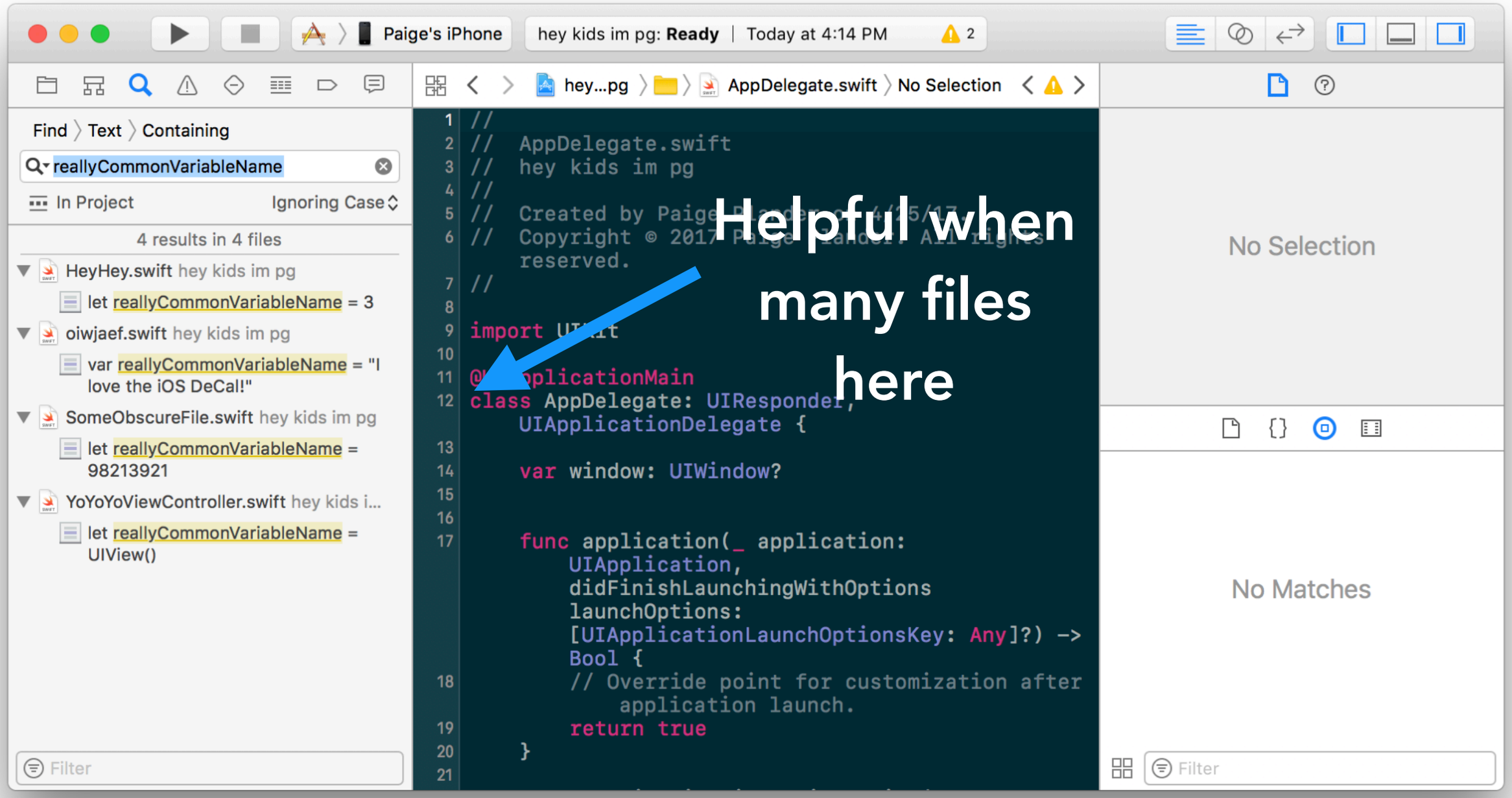
UIAlert

```
let alert = UIAlertController(title: "My Alert", message: “heyo!",
                  preferredStyle: .alert)

let okAction = UIAlertAction(title: "OK", style: .default, handler:
{(action: UIAlertAction) -> Void in
    // user pressed okay… let’s do something
})
alertController.addAction(OKAction)
```

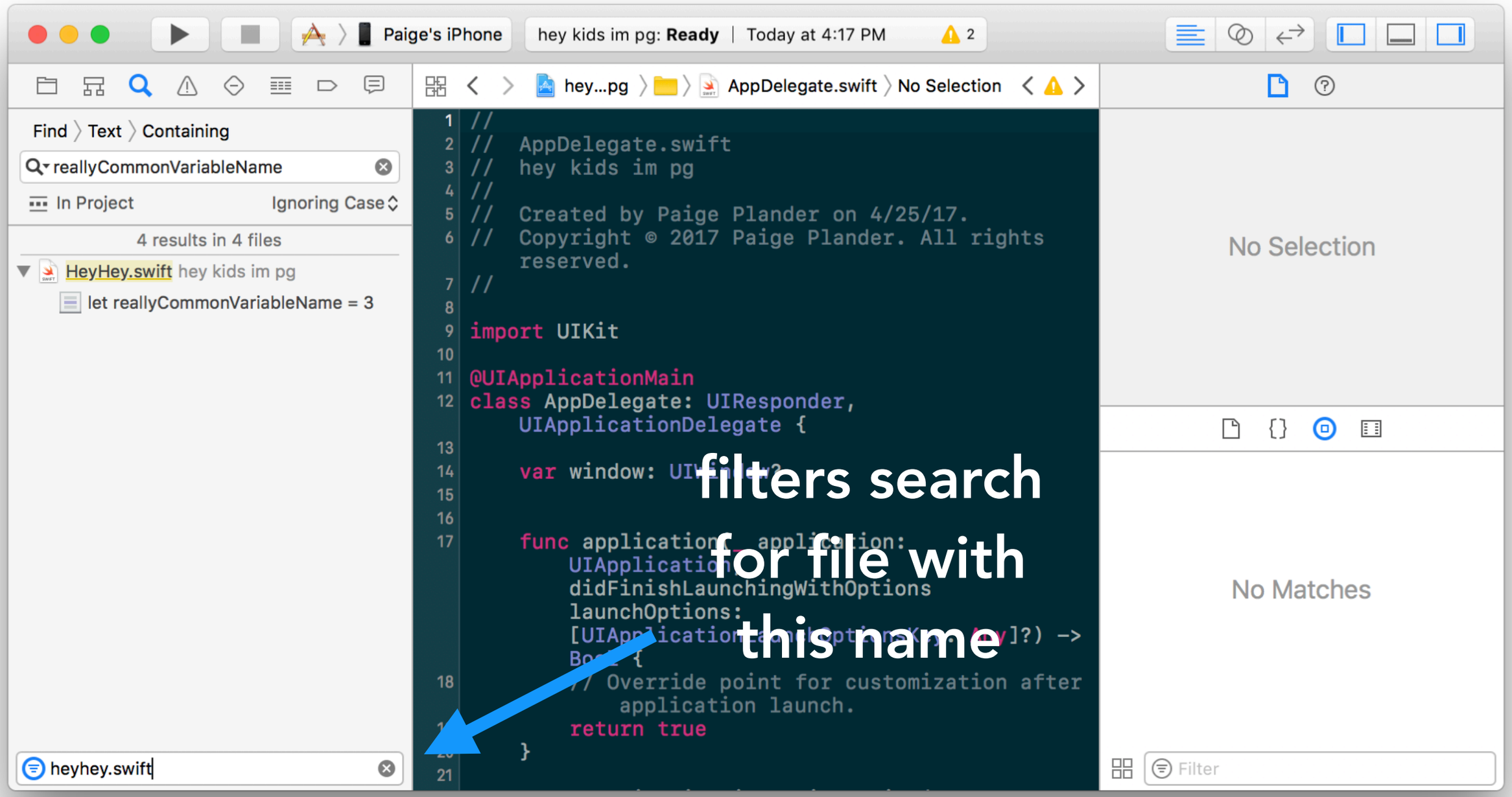
extra slides



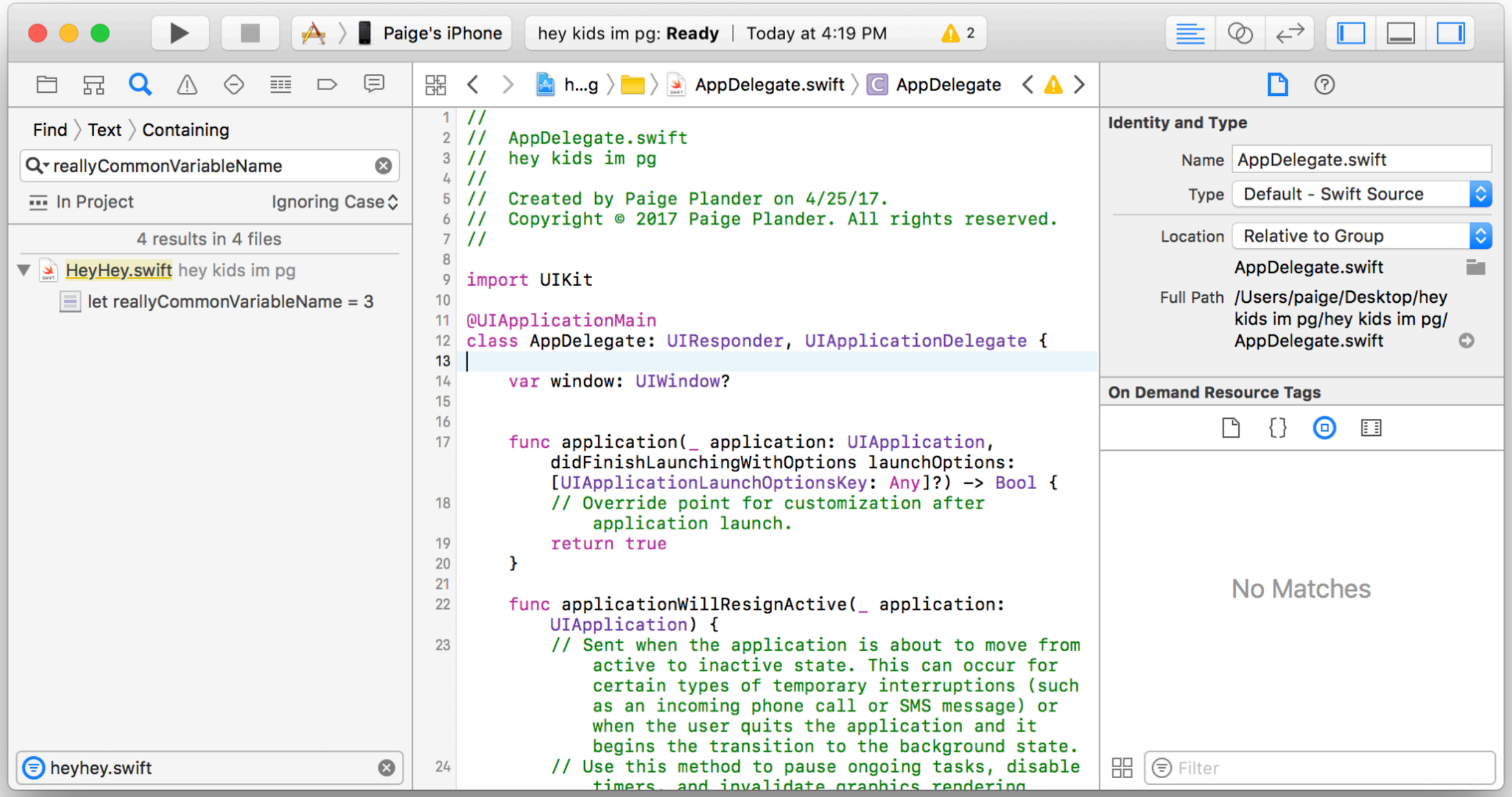
Searching for a specific variable / method / comment



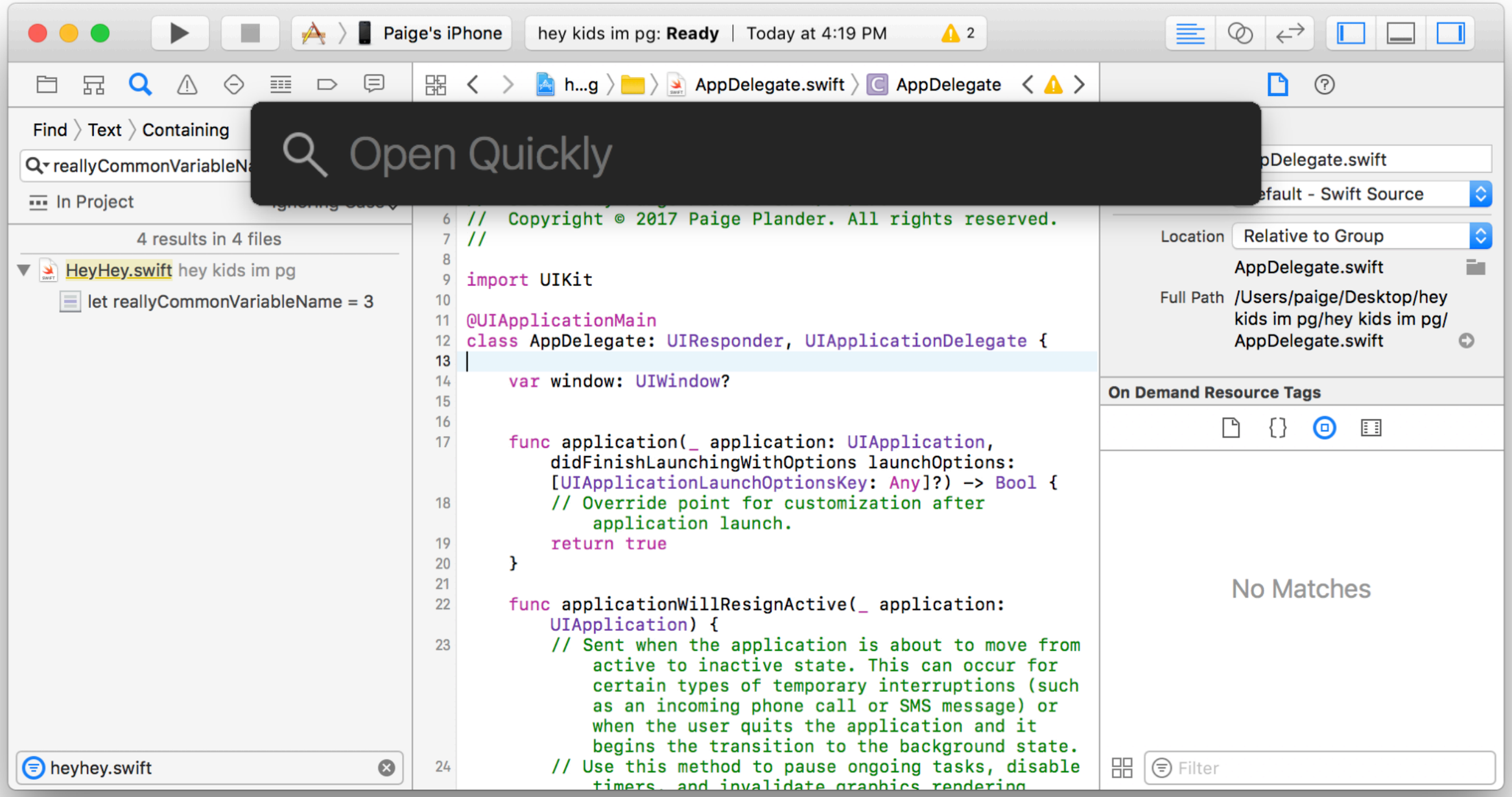
Search with filename filtering (saves a lot of time!)



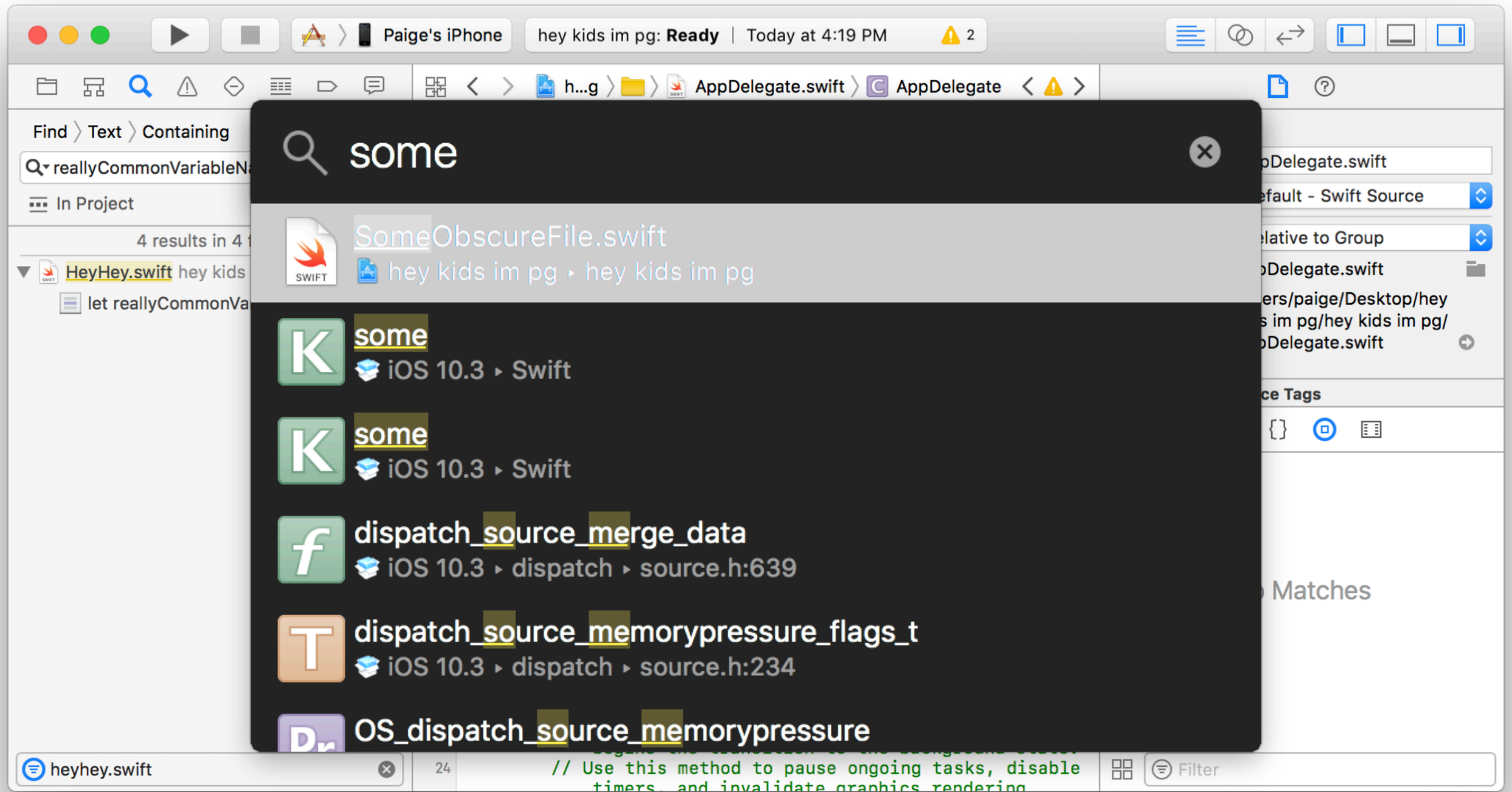
Search with filename filtering (saves a lot of time!)



Quick Search - Command + Shift + O



Quick Search - Command + Shift + O



Quick Search - Command + Shift + O

Strong vs Weak?

Strong - Two objects both increase each other's reference counts and are in memory forever.
Weak - Only one object increases reference count, so when one gets deallocated so does the other.

Retain Cycles

```
@class Child;
@interface Parent : NSObject {
    Child *child; //instance variables implicitly __strong
}
@end
@interface Child : NSObject {
    Parent *parent; //also implicitly __strong
}
@end
```