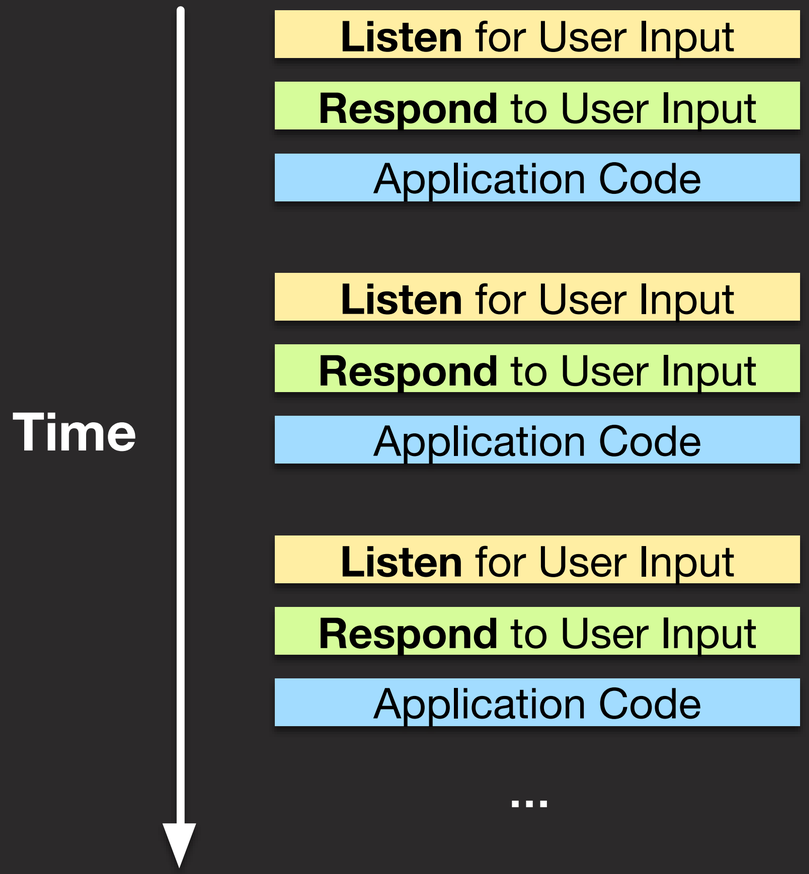
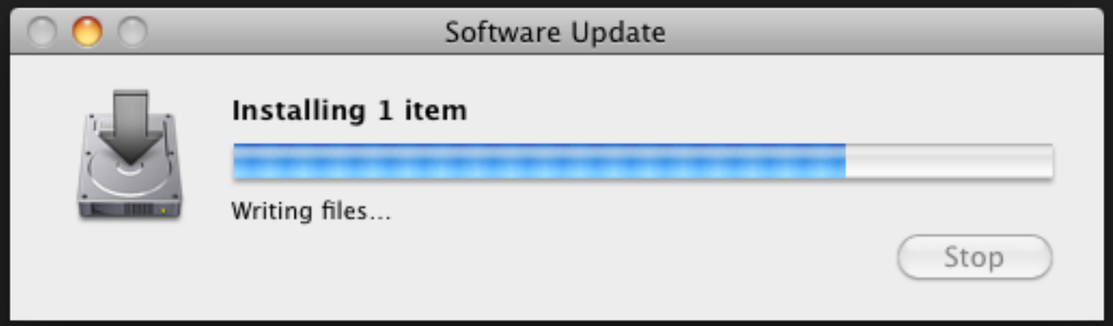
MAKE
SCHOOL

ASYNCHRONOUS CODE AND THREADING

WHY DO WE NEED THREADS?



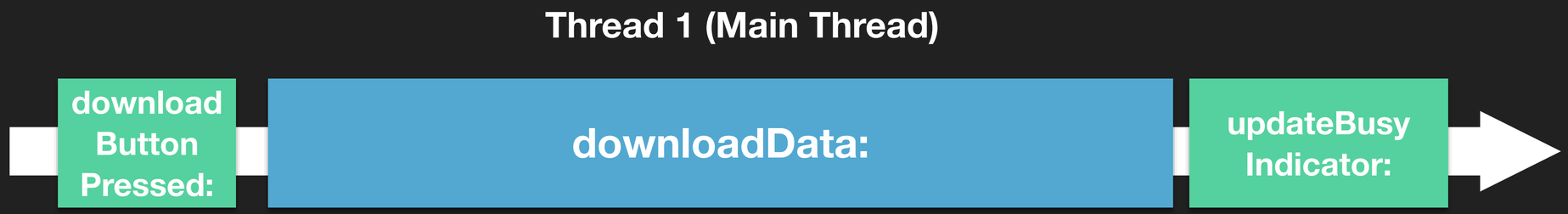
WHY DO WE NEED THREADS?



MAKE SCHOOL

MULTITHREADING BASICS

```
@IBAction func downloadButtonPressed(sender: AnyObject) {
  downloadData()
  updateBusyIndicator()
}
```



The entire program is blocked while one piece of code is running!

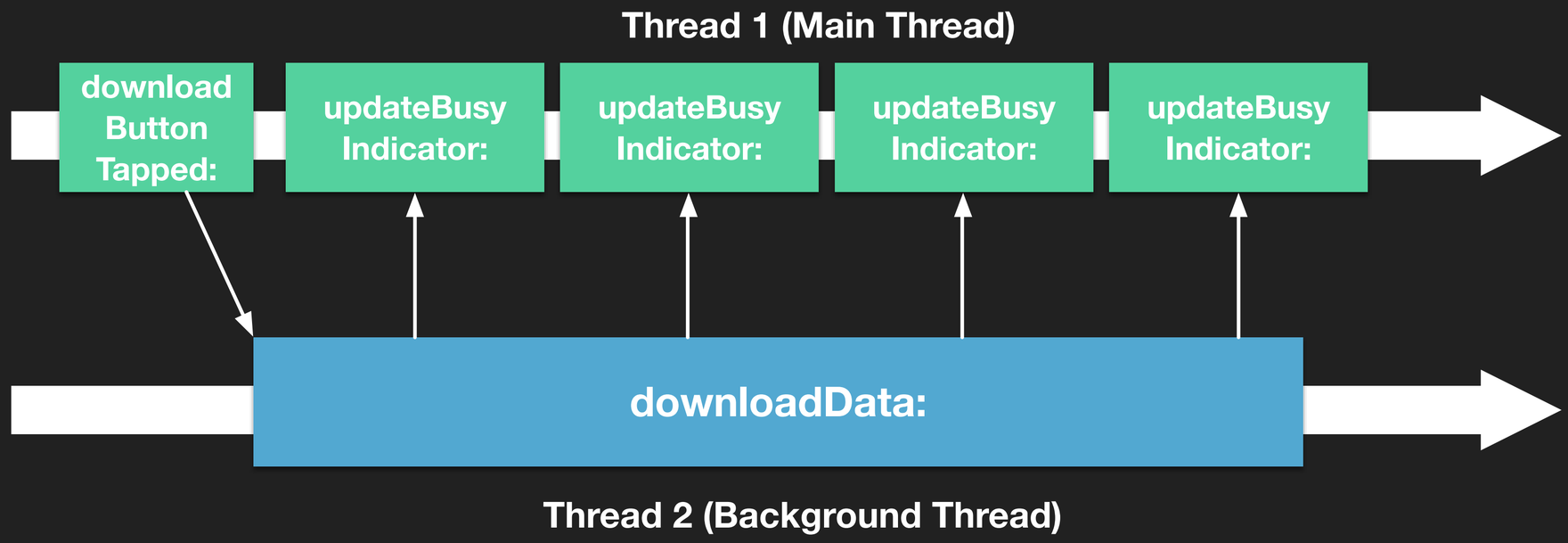
MULTITHREADING BASICS

Long running tasks block our program. The UI freezes!

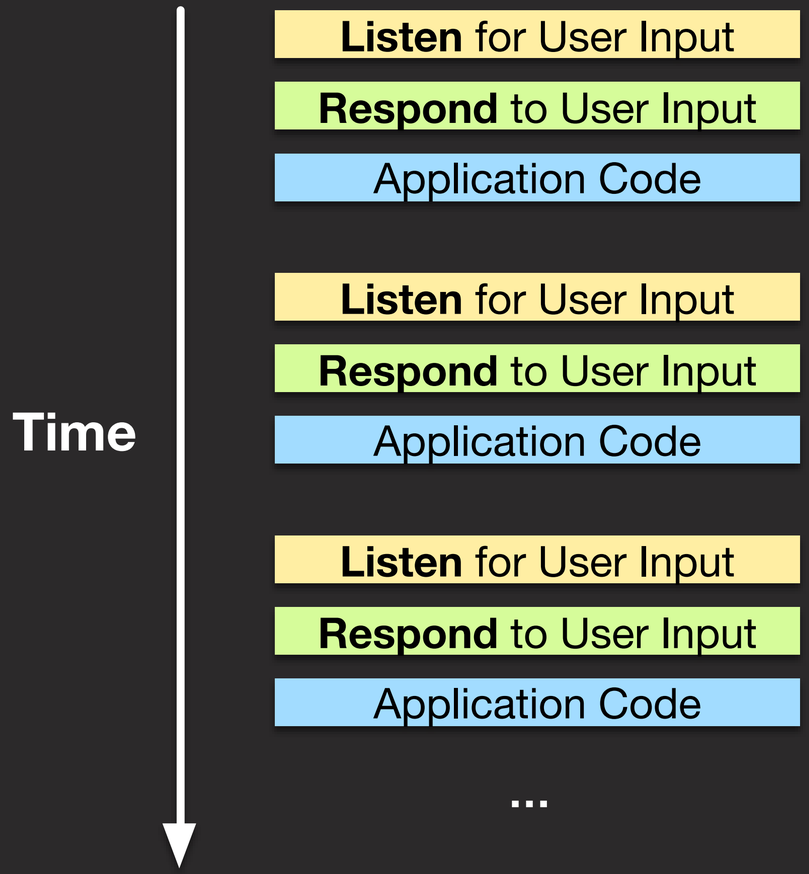
Threads allow us to run multiple tasks in **parallel**!

MULTITHREADING BASICS

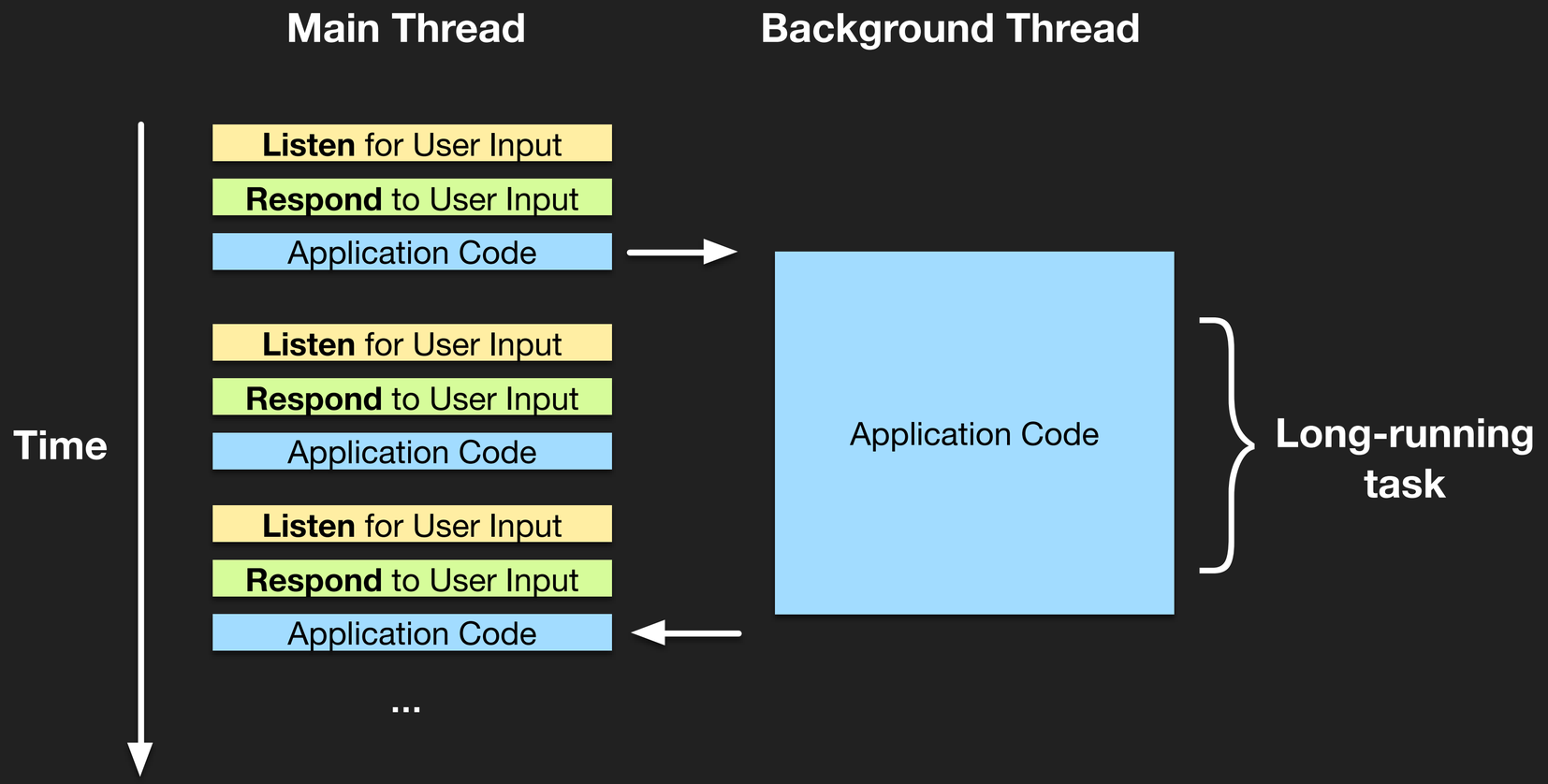
By using a background thread we can unblock the UI thread!



WHY DO WE NEED THREADS?



MULTITHREADING BASICS



OUR CODE IS NO LONGER SYNCHRONOUS!

SYNCHRONOUS CODE

```
let image = downloadImage("test")
printImageDescription(image)
print("image displayed!")


> Format: PNG, Size: 200x200
> image displayed
```

ASYNCHRONOUS CODE

```
downloadImageAsync("test")
   { (image: UIImage) -> () in
     printImageDescription(image)
   }

print(“image displayed!")
```

CLOSURE / CALLBACK

```
> image displayed
> Format: PNG, Size: 200x200
```

MAKE SCHOOL

ASYNCHRONOUS CODE

```
downloadImageAsync("test")
   { (image: UIImage) -> () in
     printImageDescription(image)
     print("image displayed!")
   }
```

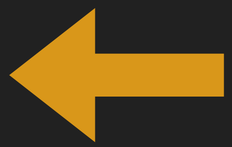
```
> Format: PNG, Size: 200x200
> image displayed
```

THREADING WITH PARSE

THREADING WITH PARSE

```
let query = PFQuery(className: ParseLikeClass)
query.whereKey(ParseLikeFromUser, equalTo: user)
query.whereKey(ParseLikeToPost, equalTo: post)
```

```
let likes = query.findObjects()
```



BLOCKS
MAIN THREAD

MAKE SCHOOL

THREADING WITH PARSE

```
let query = PFQuery(className: ParseLikeClass)
query.whereKey(ParseLikeFromUser, equalTo: user)
query.whereKey(ParseLikeToPost, equalTo: post)

query.findObjectsInBackgroundWithBlock( {
   (results: [PFObject]?, error: NSError?) -> Void in
      // …
})
```

MAKE SCHOOL

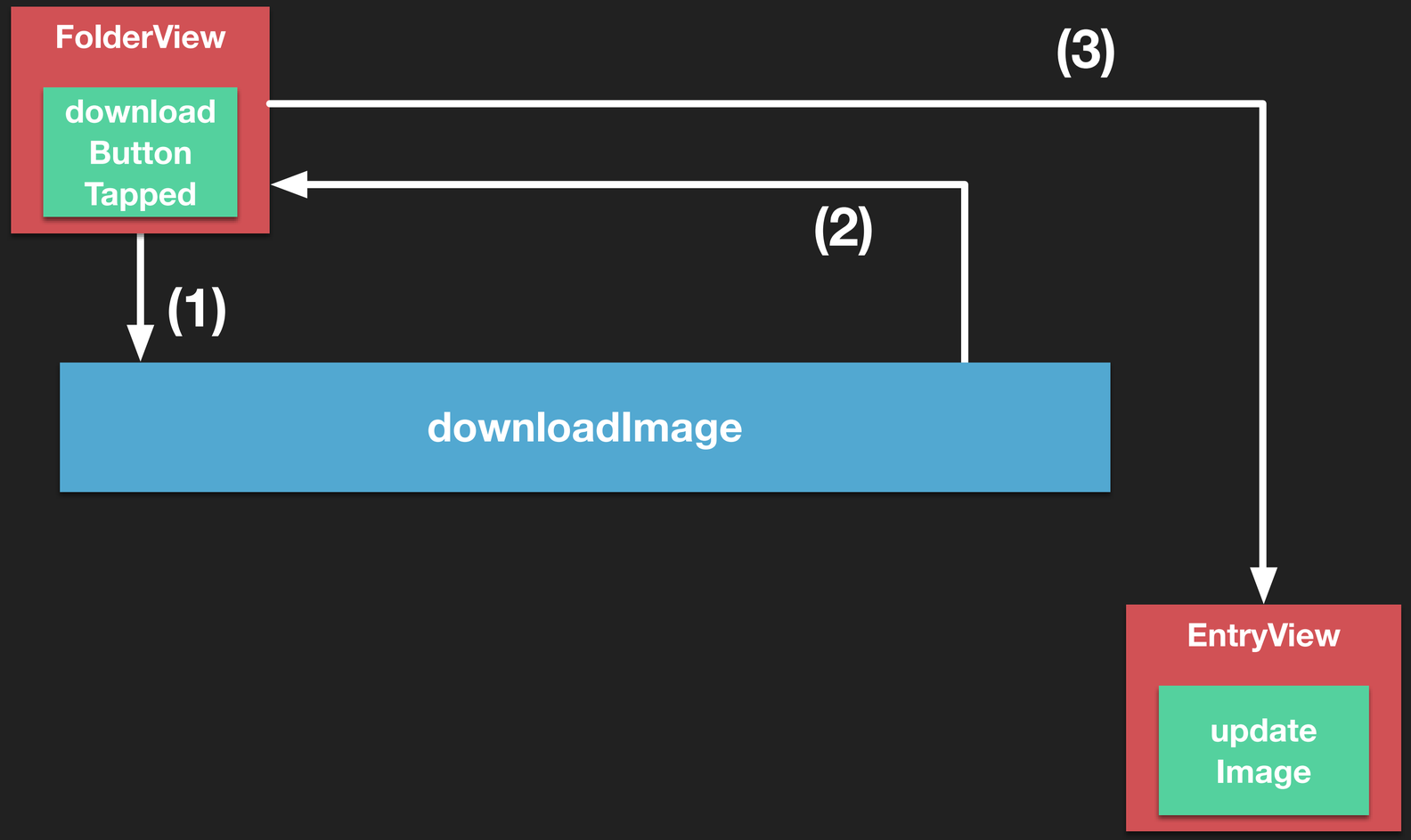
OBSERVABLES AND BONDS

OBSERVABLES AND BONDS

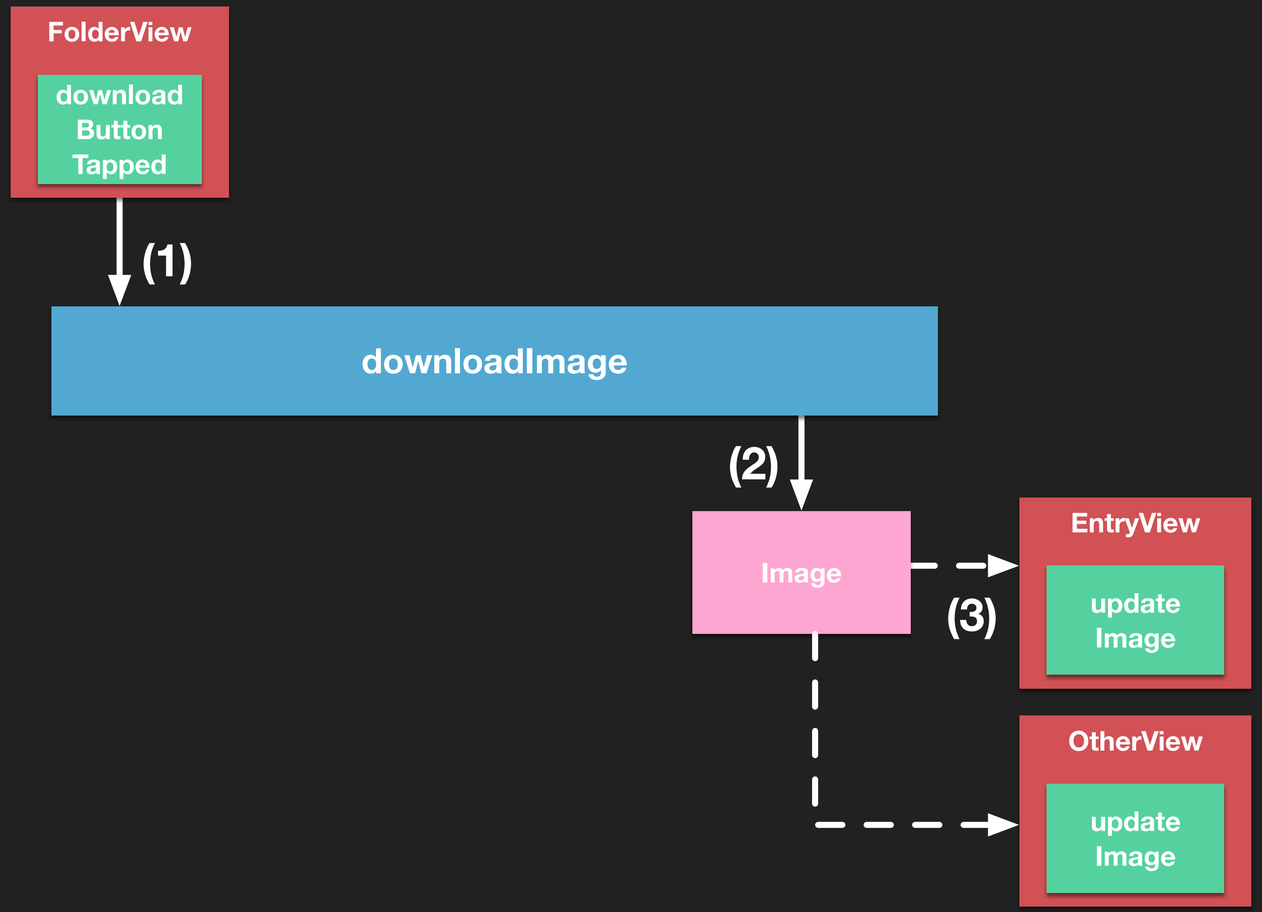
Callbacks are a very useful tool, but they have limitations

What if the code that triggers the network request is not the code that is interested in the result?

CLASSIC CALLBACK



OBSERVABLE AND BOND



OBSERVABLE PROPERTIES

```
// declaring Observable property
var image: Observable<UIImage?>

// Setting Observable property
image.value = UIImage()
```

BONDS

```
post.likes.observe {
(value: [PFUser]?) -> () in
    //...
}
```

MAKE SCHOOL

BONDS

```
post.image.bindTo(postImageView.bnd_image)
```

MAKE SCHOOL

SUMMARY

We perform code in the background to keep the UI Thread responsive

Performing code in the background leads to asynchronous program flow

Observables and Bonds are a great way to deal with asynchrony