Developer Tools

#WWDC15

Improving your Existing Apps with Swift

Getting Swifty with It

Session 403

Woody L. 🐳 in the Sea of Swift

Demo

Set the Time Machine to 2012

The Elements has been restored from a vault

Agenda

The “before” app

Modernizing UI

Mix and match

Swift Structs

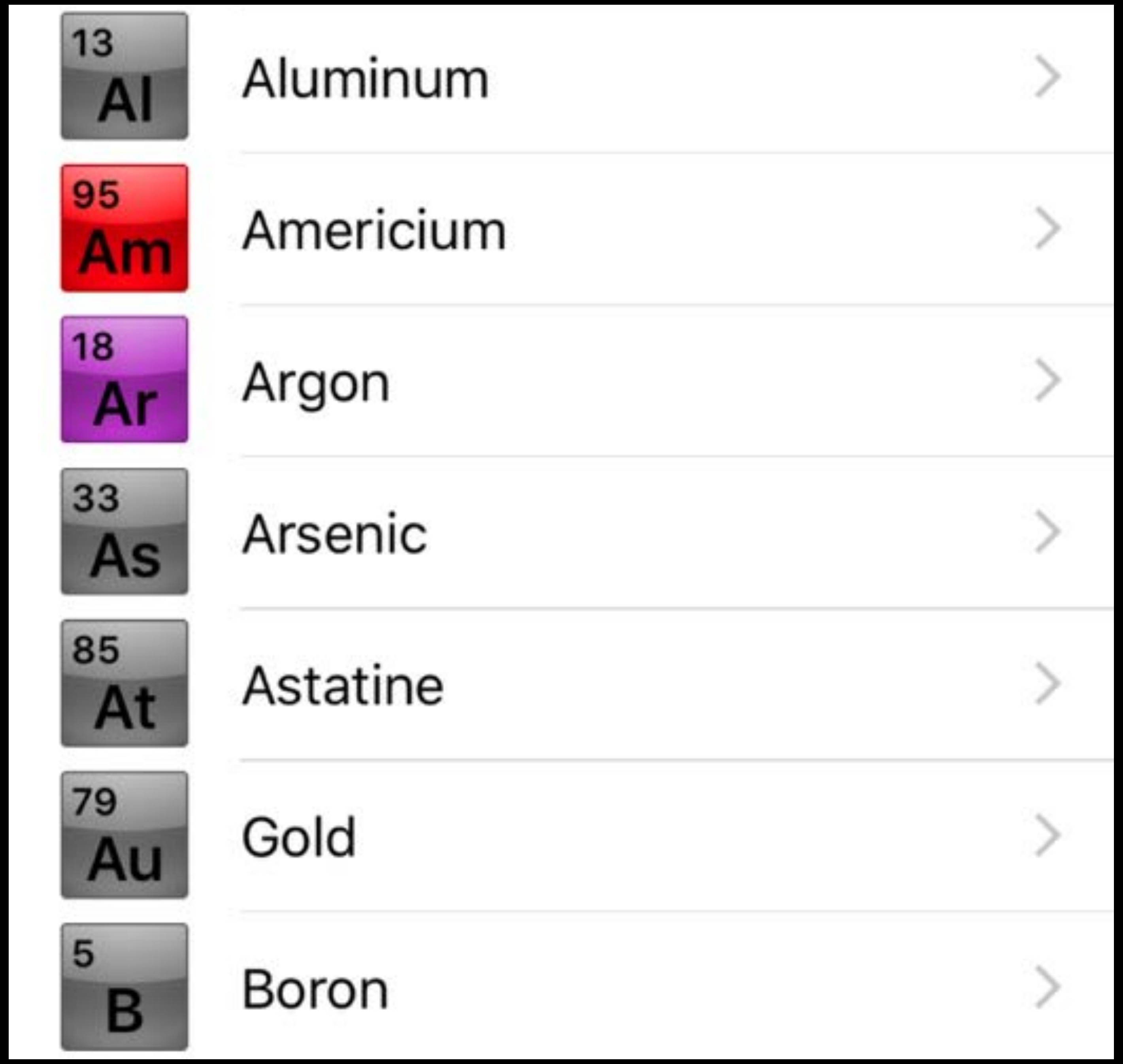
Availability

Live filtering

Modernizing UI

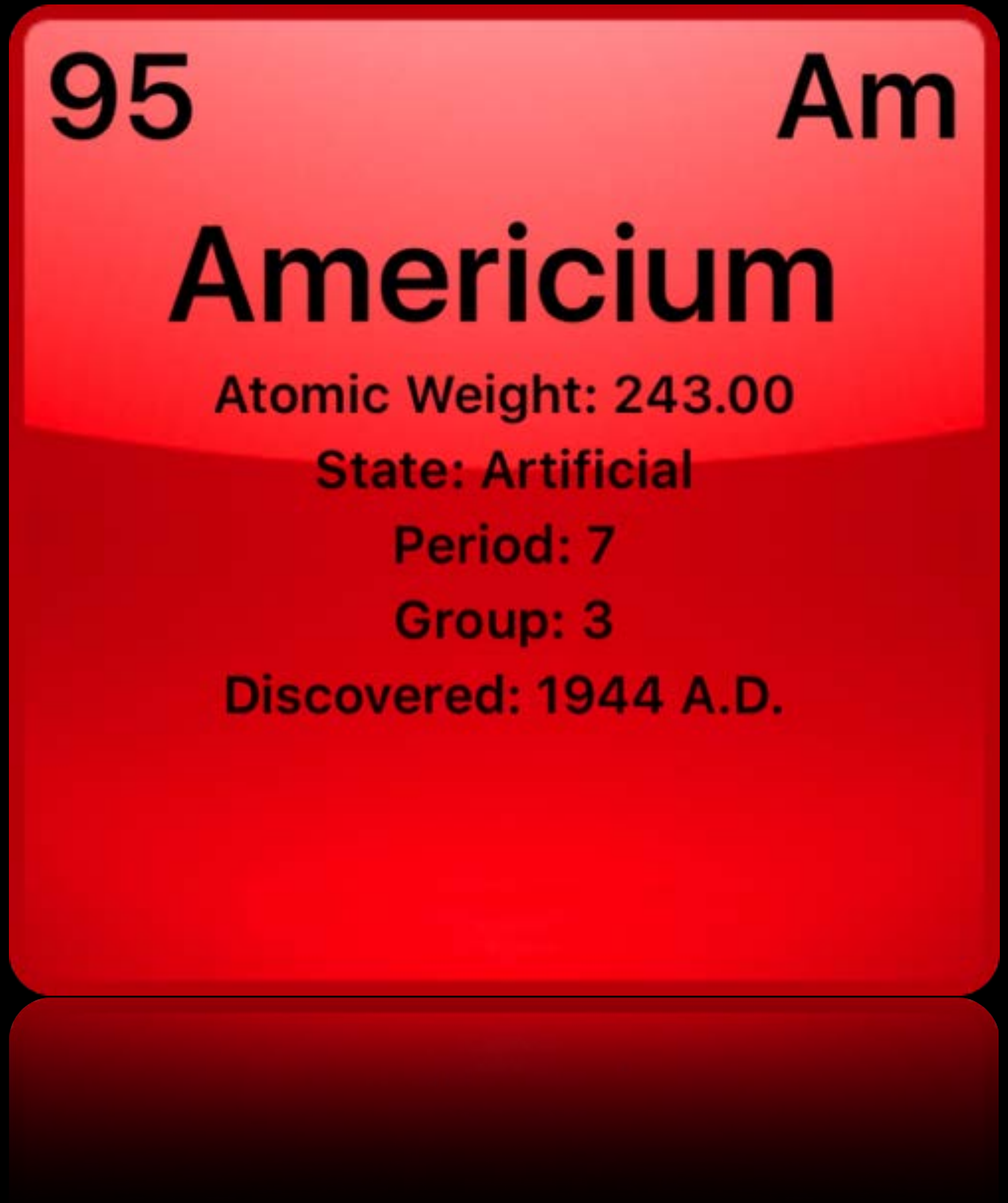
Walk. Walk. Fashion, Baby.

Tile Sizes



13 Al	Aluminum
95 Am	Americium
18 Ar	Argon
33 As	Arsenic
85 At	Astatine
79 Au	Gold
5 B	Boron

Small Tile



95 Am

Americium

Atomic Weight: 243.00
State: Artificial
Period: 7
Group: 3
Discovered: 1944 A.D.

Detail Tile

18 Ar

Argon

Atomic Weight: 39.95

State: Gas

Period: 3

Group: 18

Discovered: 1849 A.D.

18 **Ar**

Argon

Atomic Weight: 39.95

State: Gas

Period: 3

Group: 18

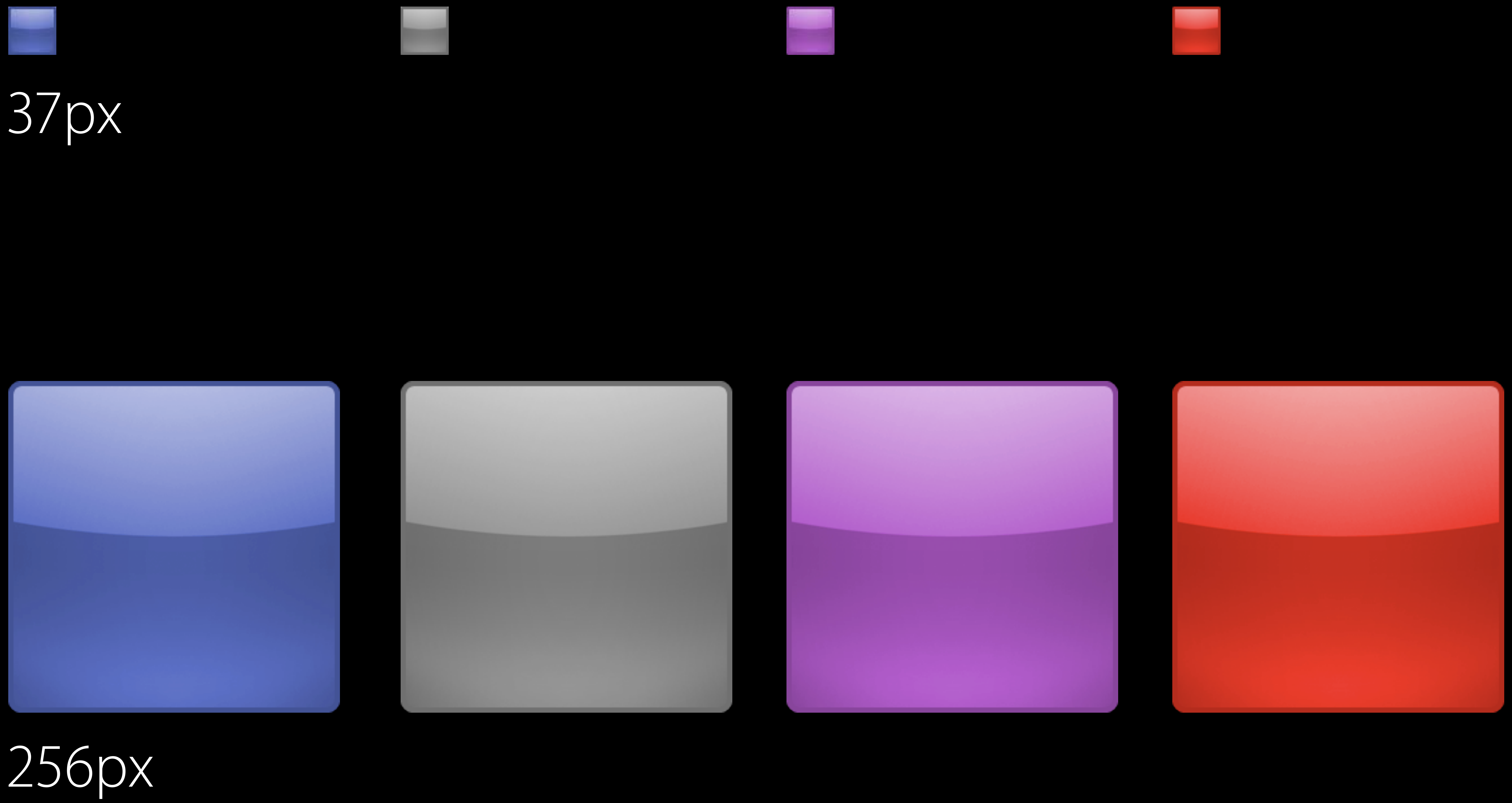
Discovered: 1849 A.D.

Pre-Rendered Backgrounds

37px

256px

Pre-Rendered Backgrounds



256px = 256pt @ 1x

512px = 256pt @ 2x

768px = 256pt @ 3x

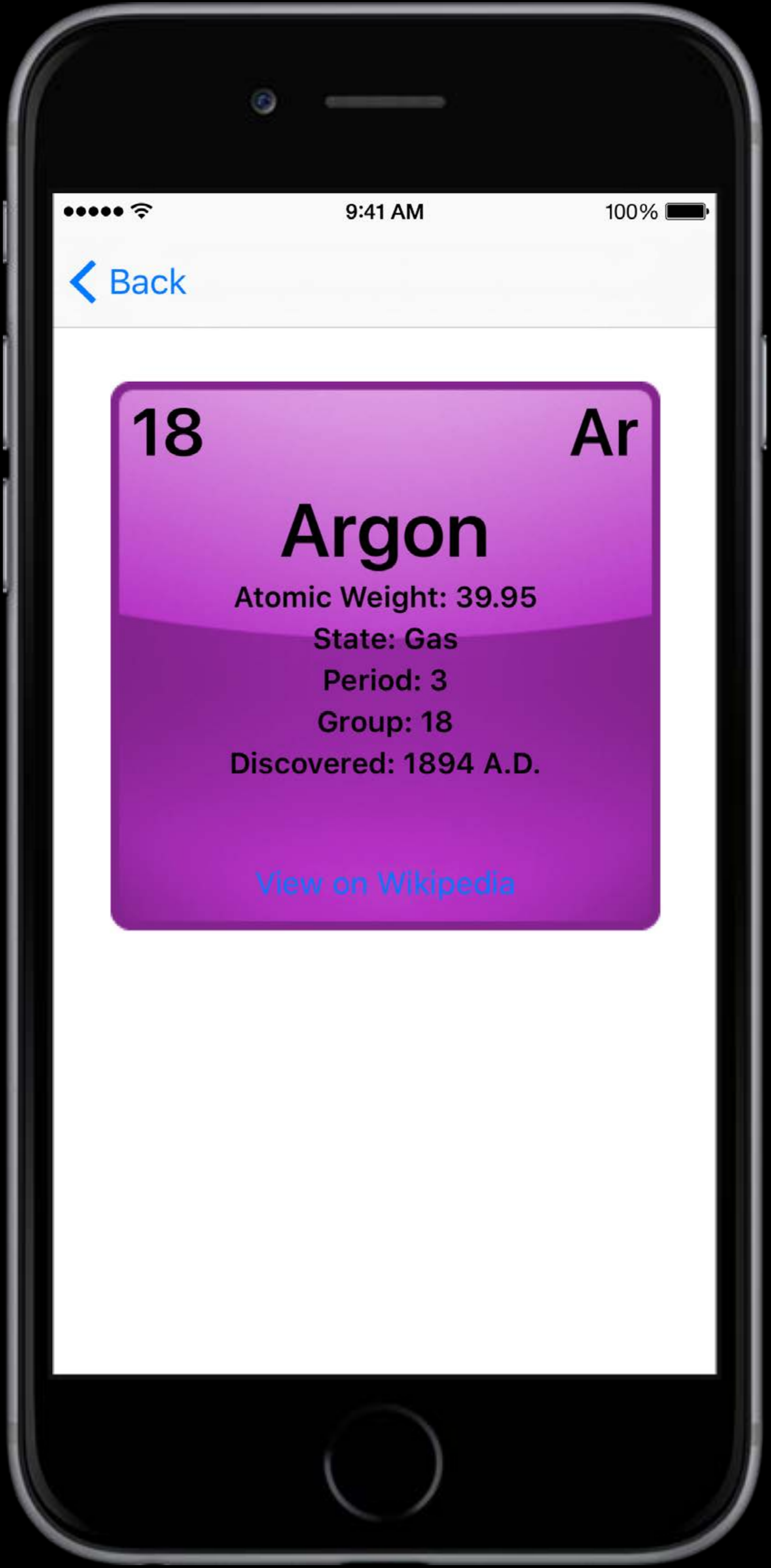
Does Not
Exist

Does Not
Exist

256px = 256pt @ 1x

512px = 256pt @ 2x

768px = 256pt @ 3x



18 Ar

Argon

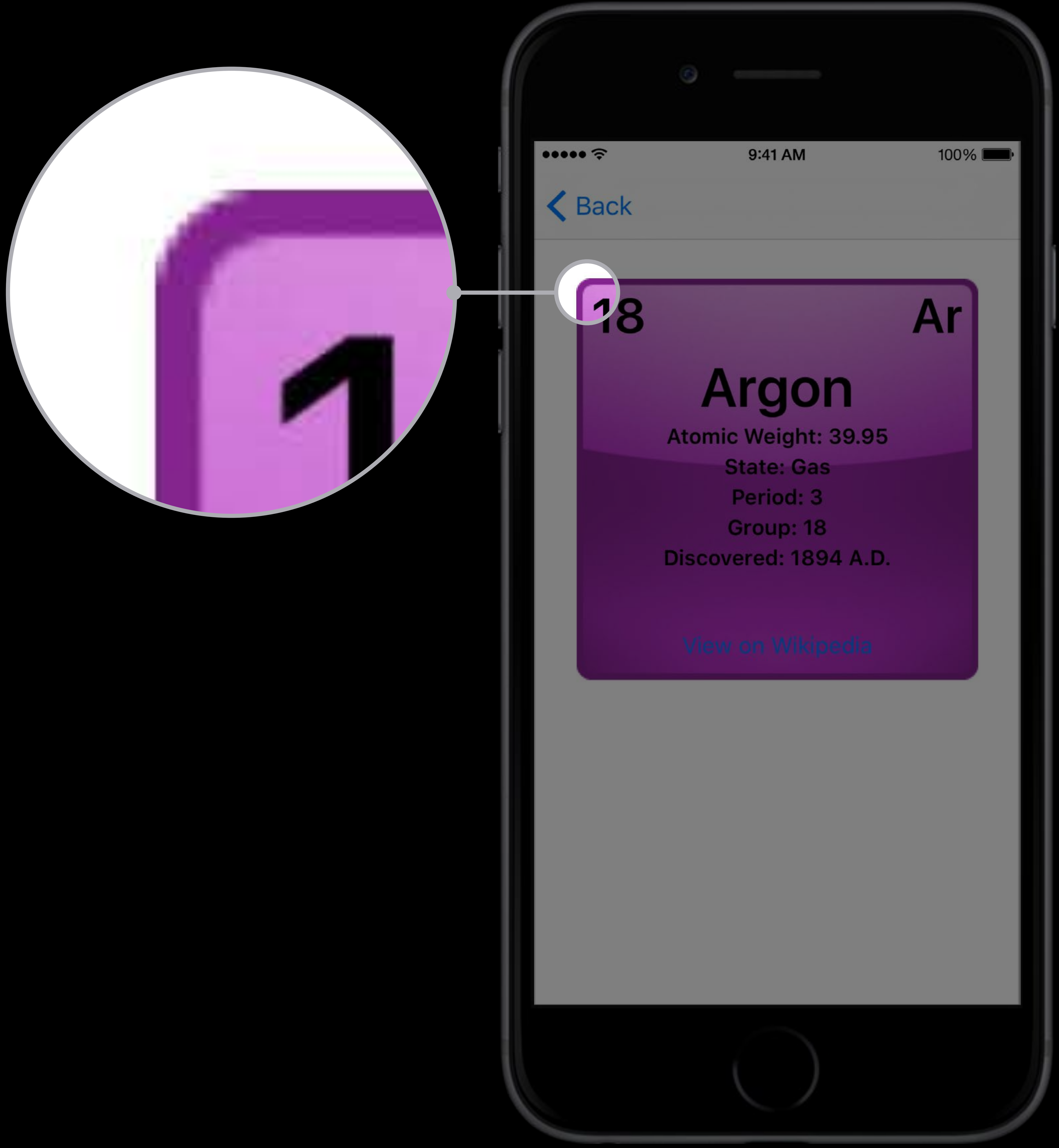
Atomic Weight: 39.95

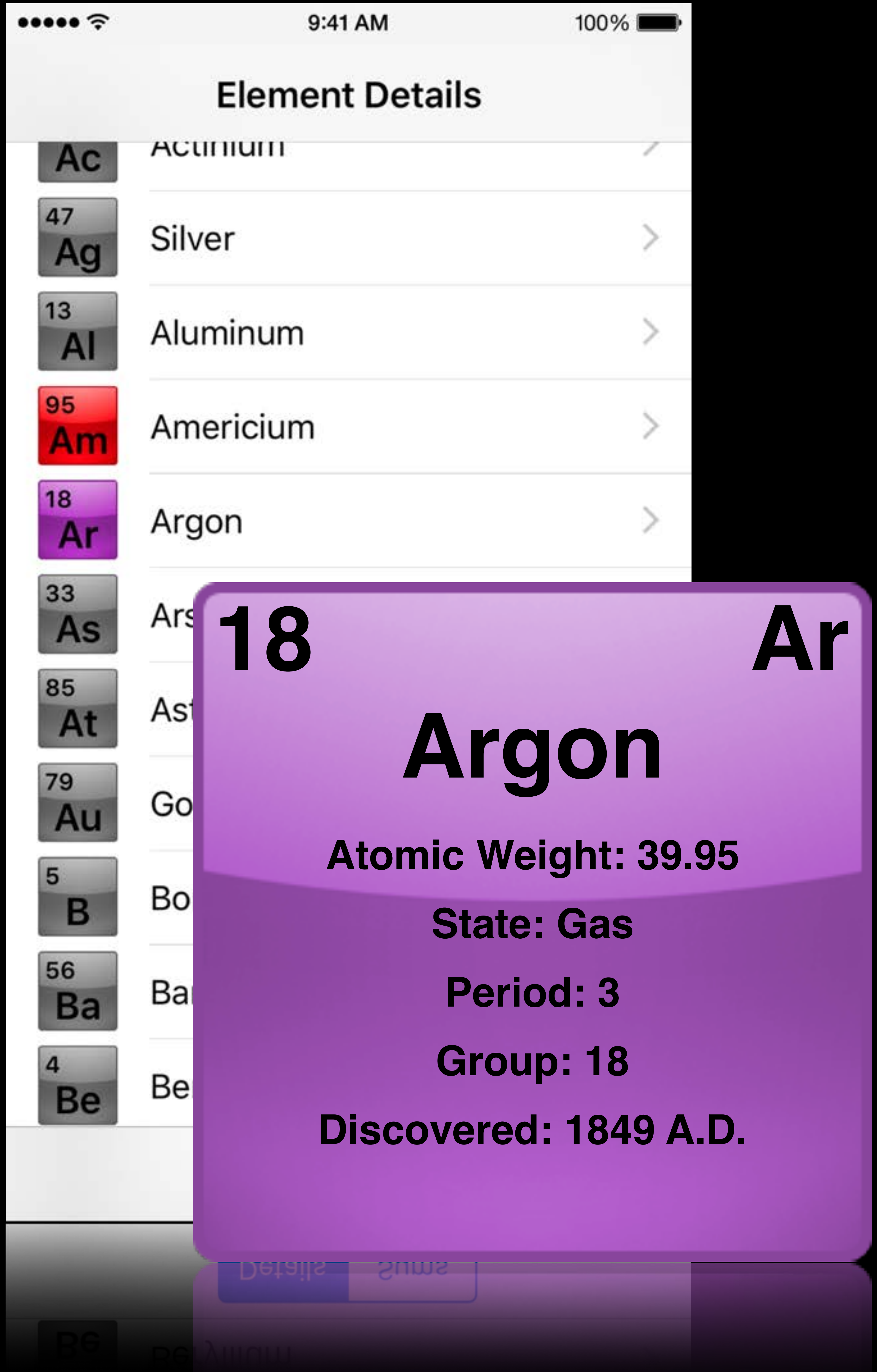
State: Gas

Period: 3

Group: 18

Discovered: 1894 A.D.


9:41 AM
100%
Back
18
Ar
Argon
Atomic Weight: 39.95
State: Gas
Period: 3
Group: 18
Discovered: 1894 A.D.
View on Wikipedia


9:41 AM 100%

Element Details

Ac	Actinium
47 Ag	Silver
13 Al	Aluminum
95 Am	Americium
18 Ar	Argon
33 As	Ars
85 At	Ast
79 Au	Go
5 B	Bo
56 Ba	Ba
4 Be	Be

18 Ar

Argon

Atomic Weight: 39.95

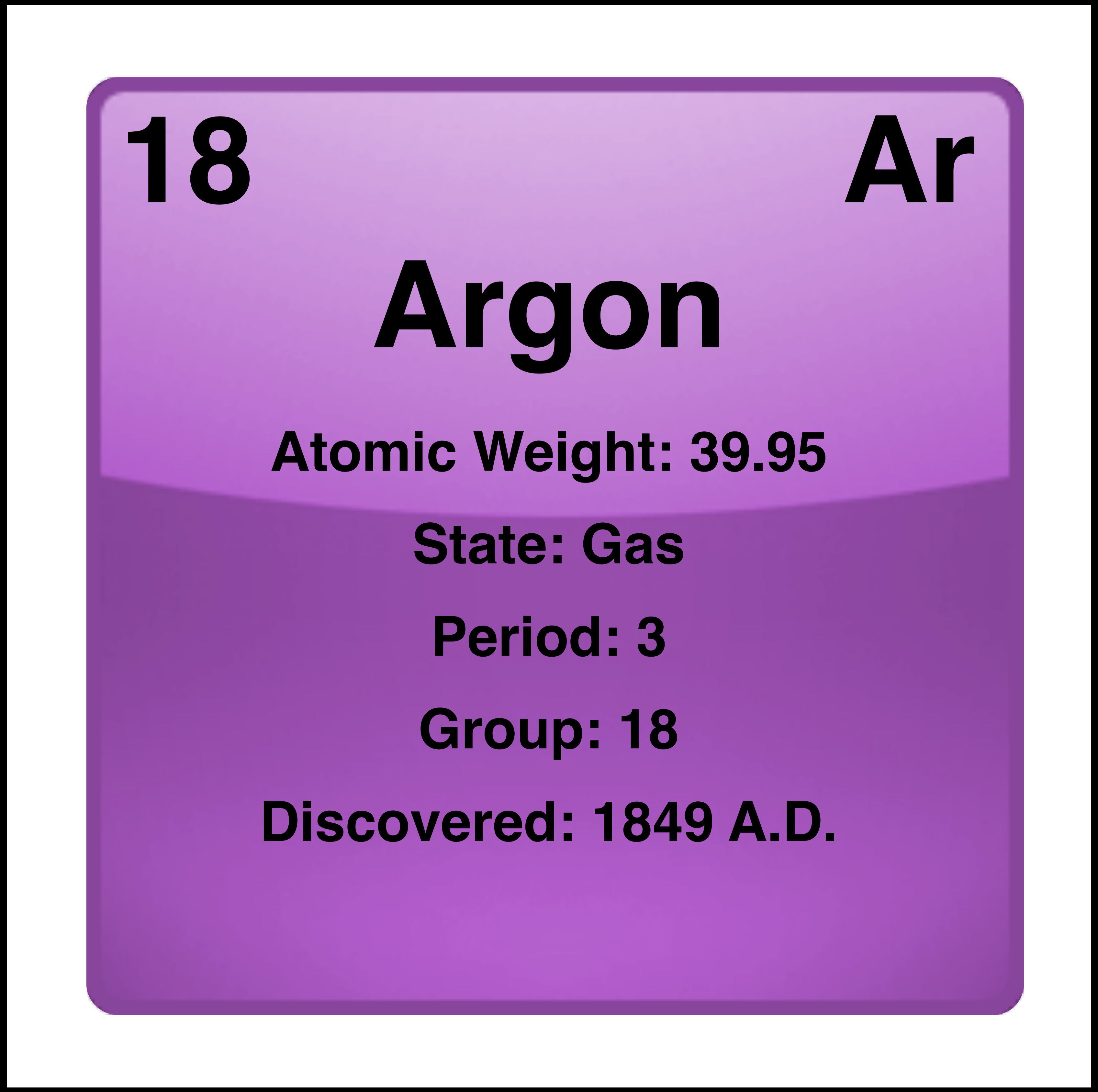
State: Gas

Period: 3

Group: 18

Discovered: 1849 A.D.

Tile Appearance


18

Ar

Argon

Atomic Weight: 39.95

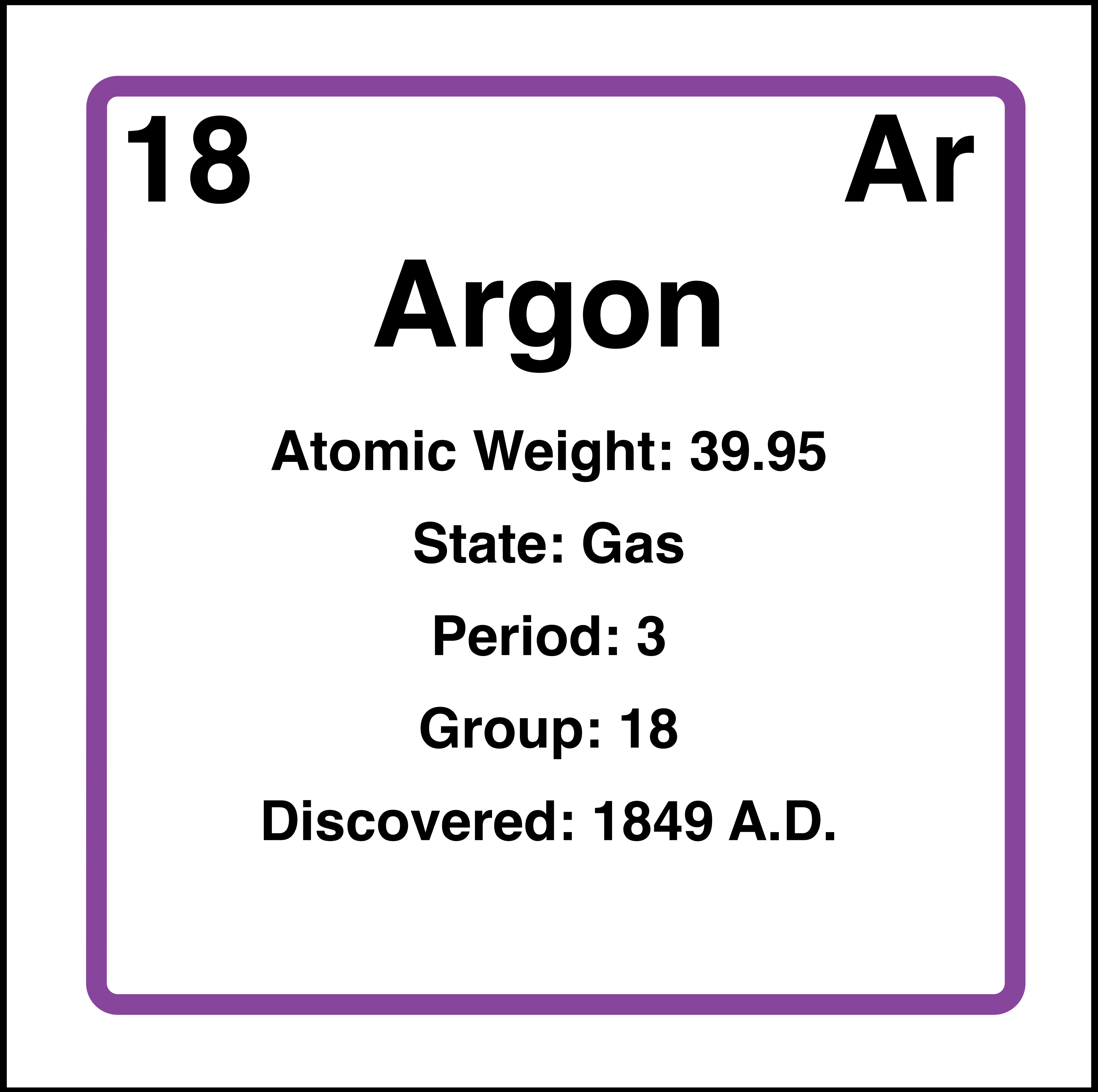
State: Gas

Period: 3

Group: 18

Discovered: 1849 A.D.

Tile Appearance



18 Ar

Argon

Atomic Weight: 39.95

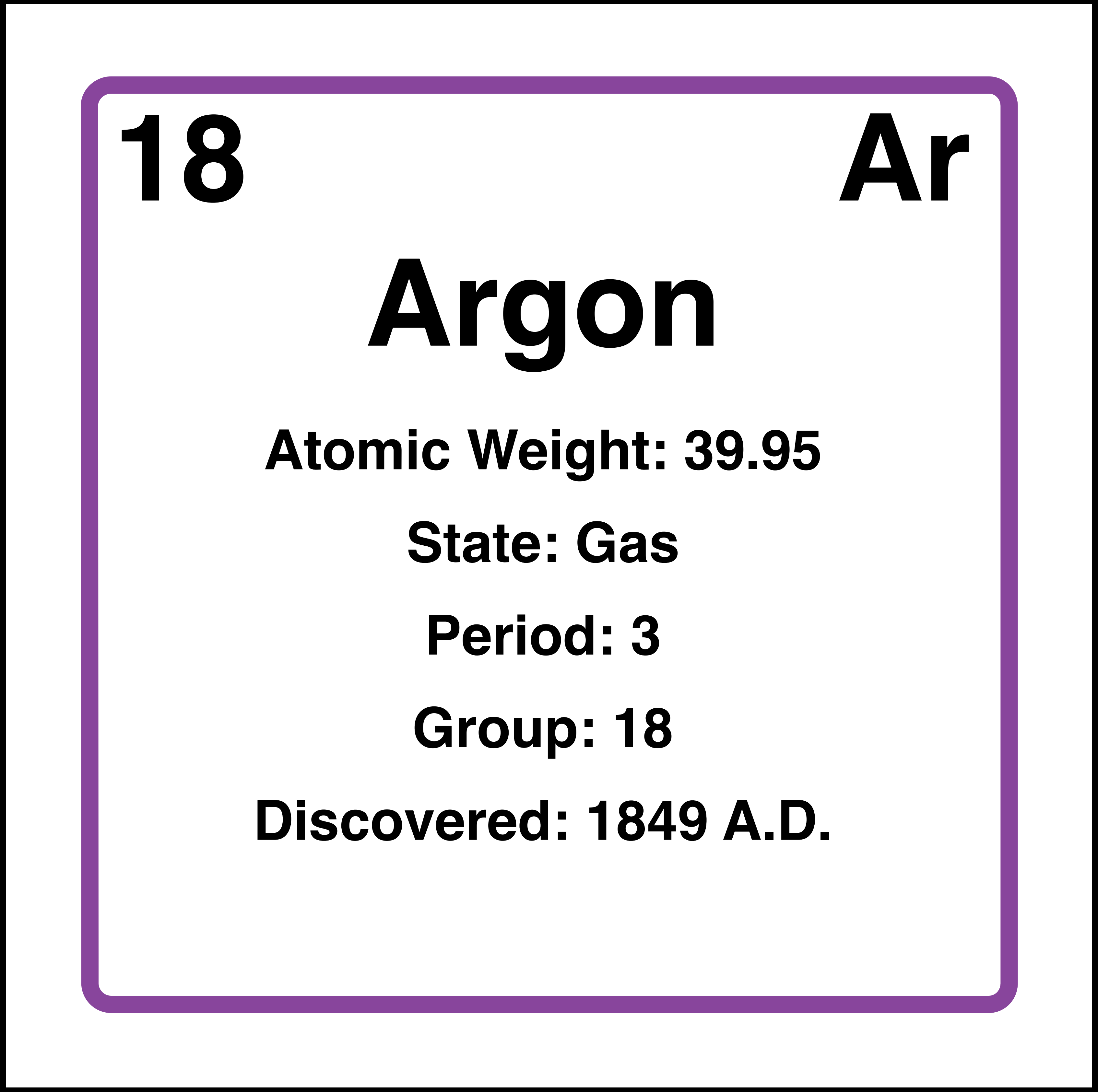
State: Gas

Period: 3

Group: 18

Discovered: 1849 A.D.

Tile Appearance


18 Ar

Argon

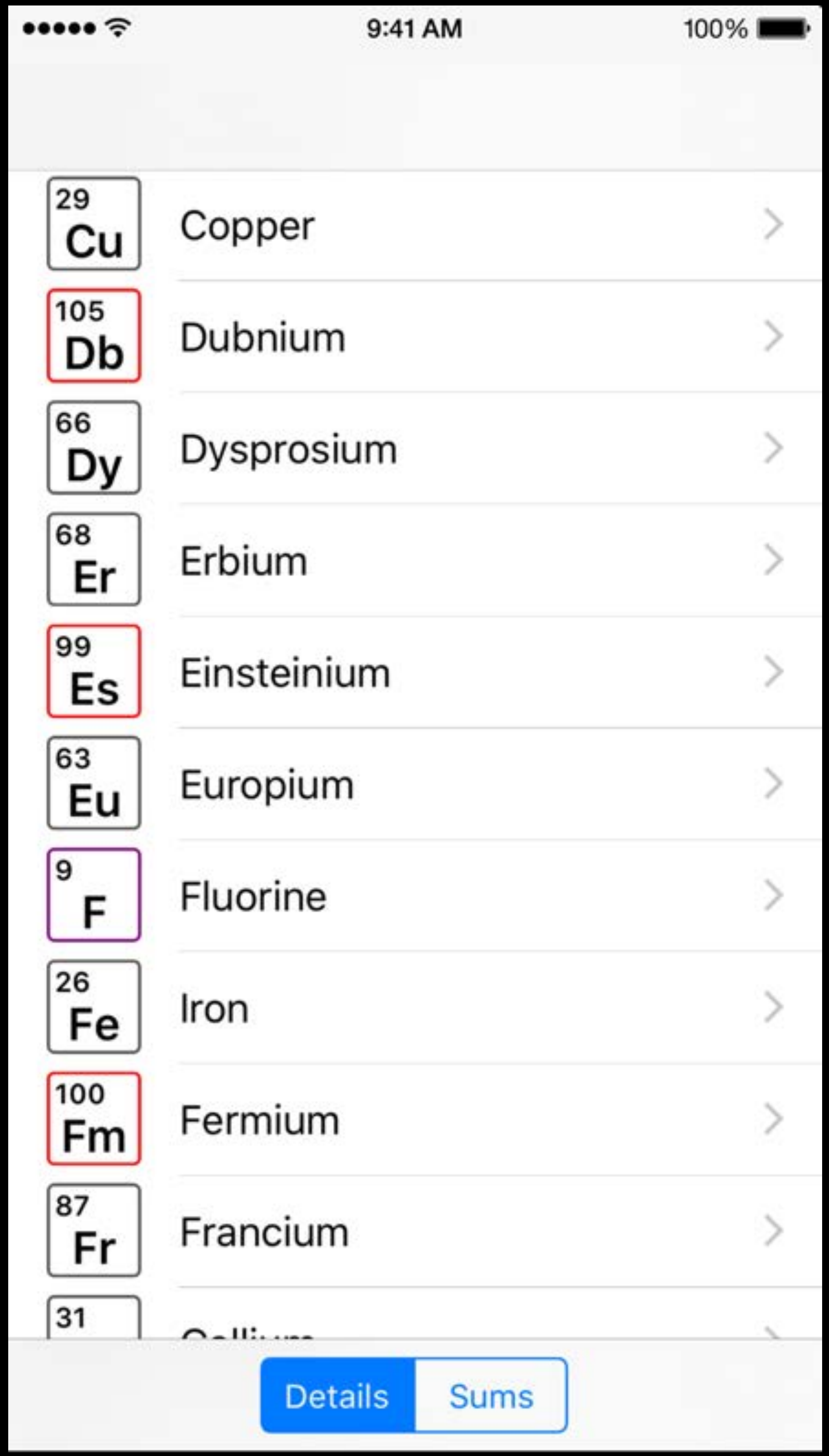
Atomic Weight: 39.95

State: Gas

Period: 3

Group: 18

Discovered: 1849 A.D.



Benefits of Custom Drawing Code

Goals	Custom Drawing Code
Modernize the Look	✓
Crisp, Sharp Corners	✓
Resolution Independence	✓

Mix and Match

Extending Objective-C Classes with Swift

An interoperability overture in the key of Swift.

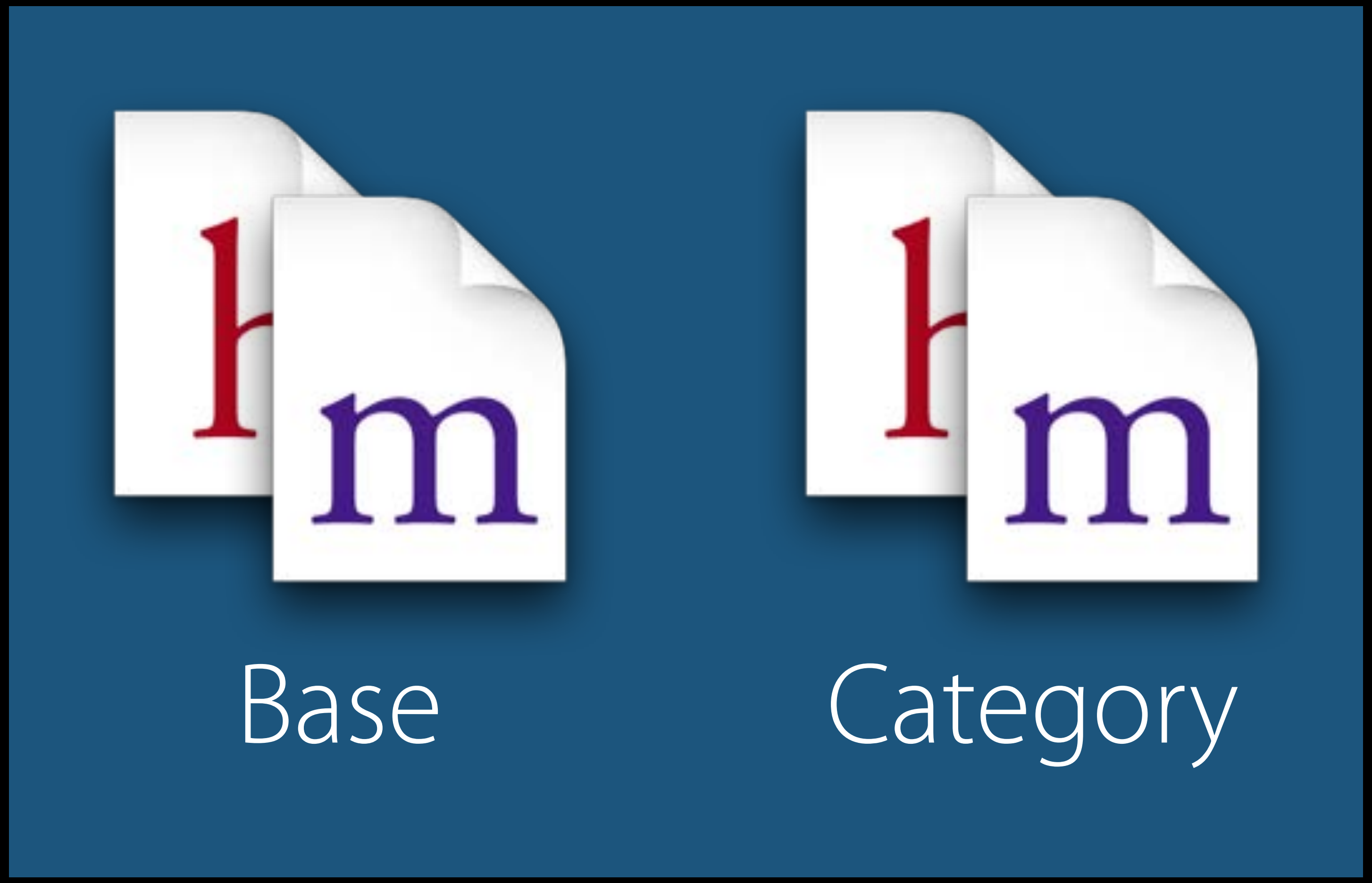
Class Design

Base Class

Class Definition

Class Design

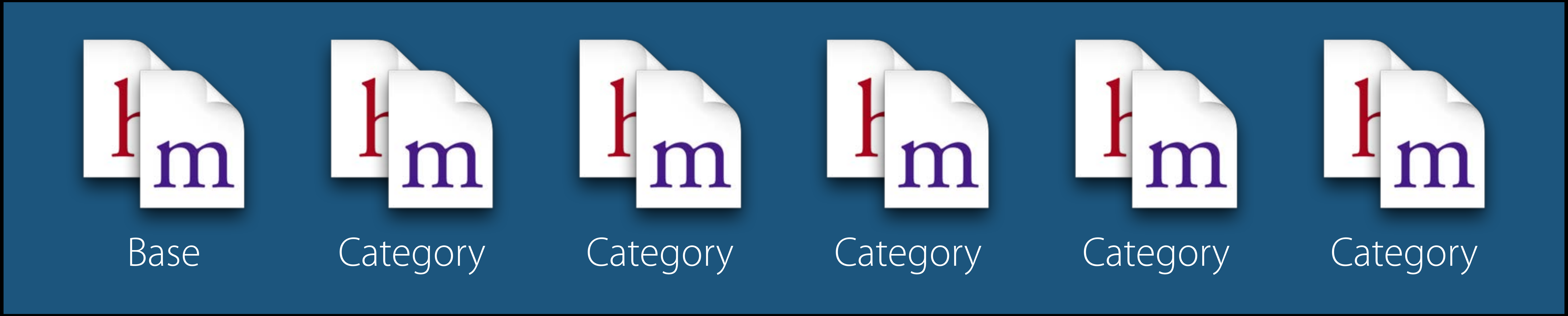
Base Class + Category



Class Definition

Class Design

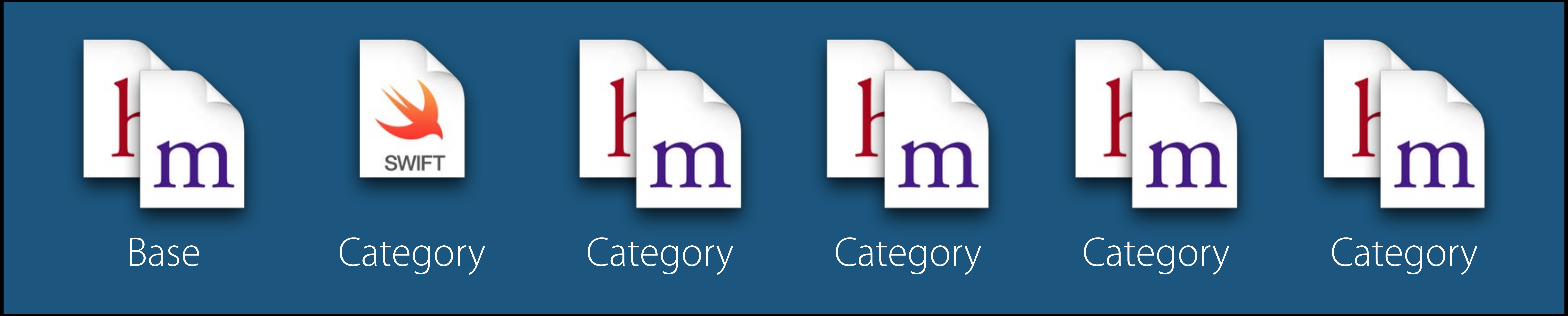
Base Class + Categories



Class Definition

Class Design

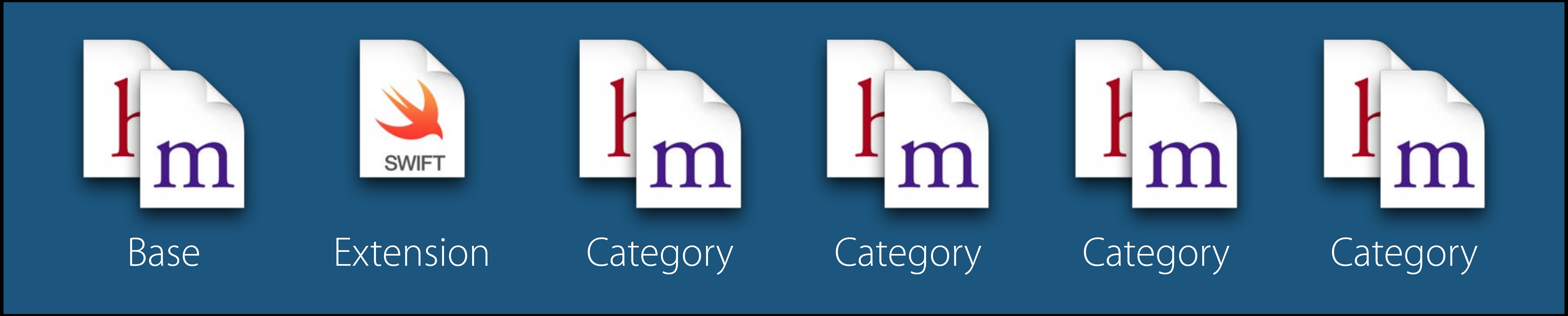
Base Class + Categories



Class Definition

Class Design

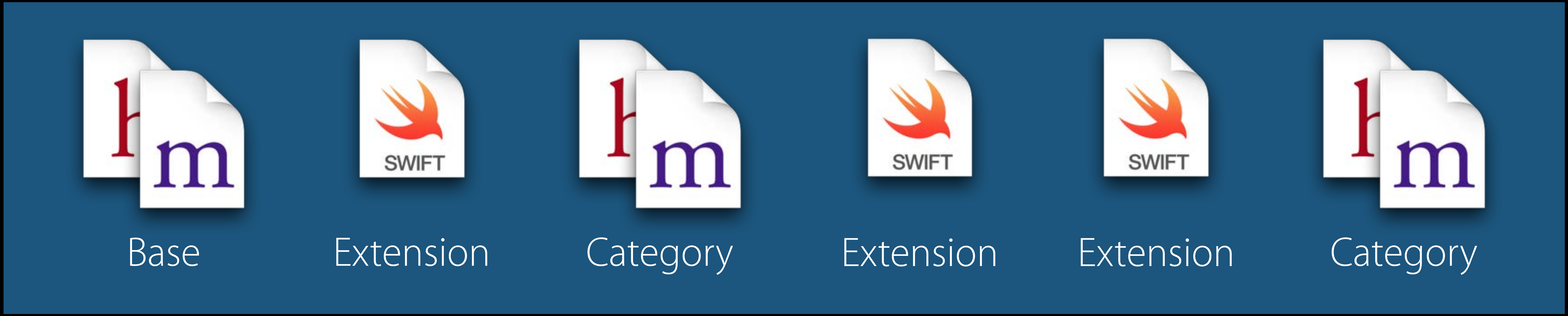
Base Class + Categories + Swift Extensions



Class Definition

Class Design

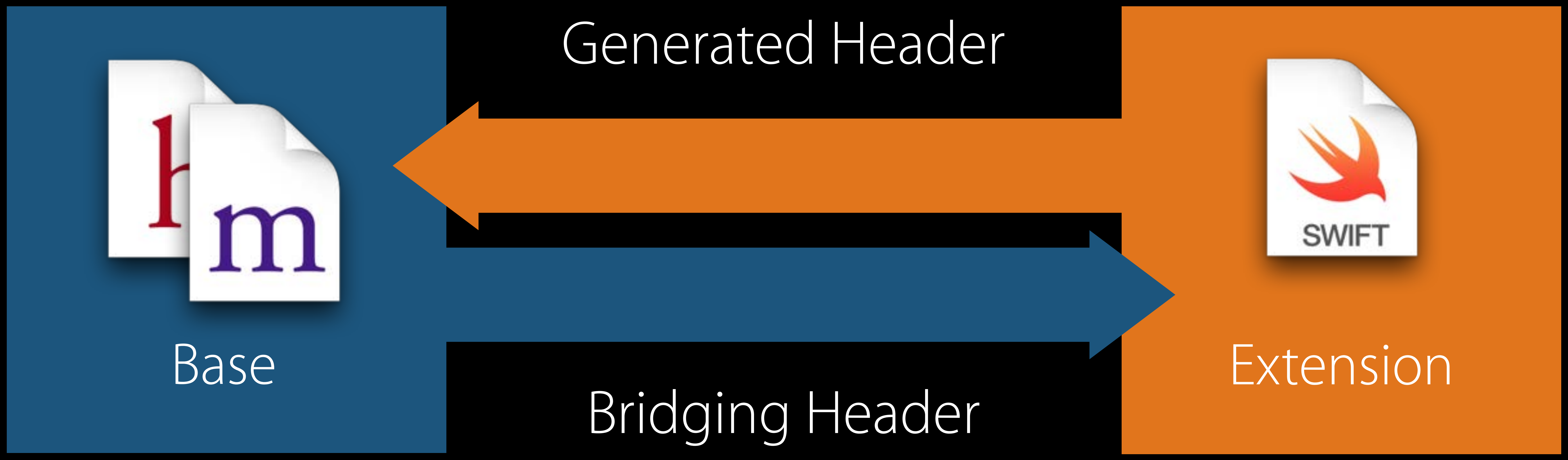
Base Class + Categories



Class Definition

Class Design

Mixed-Language Classes



Class Design

Mixed-Language Classes

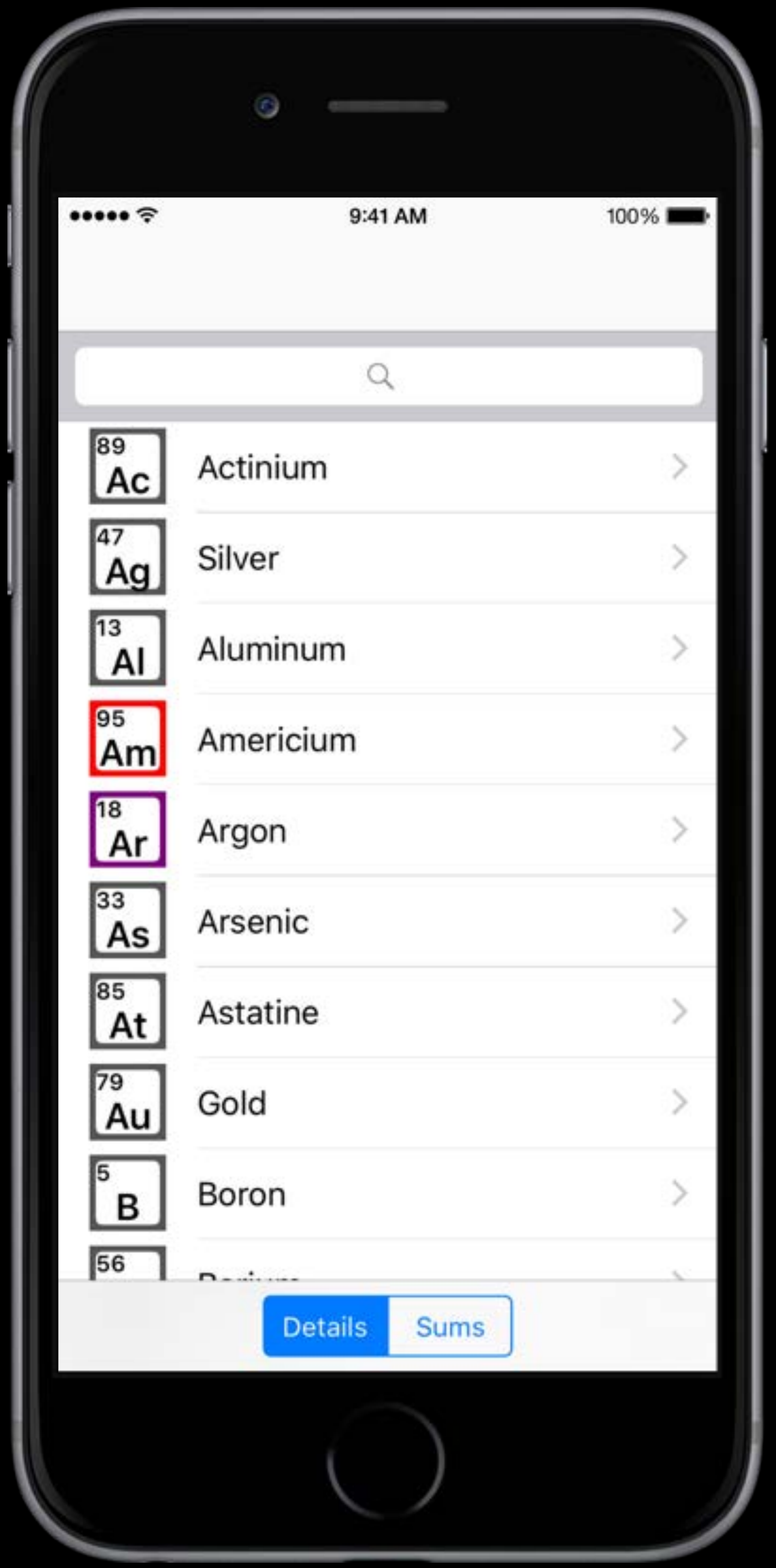
Bridging Header	Generated Header
Created by Xcode	Created by Compiler
Maintained by you	Maintained by Compiler
Contains #import "MyClass.h"	#import "Product-Swift.h" into MyClass.m
Exists in File Navigator	Exists in DerrivedData

Demo

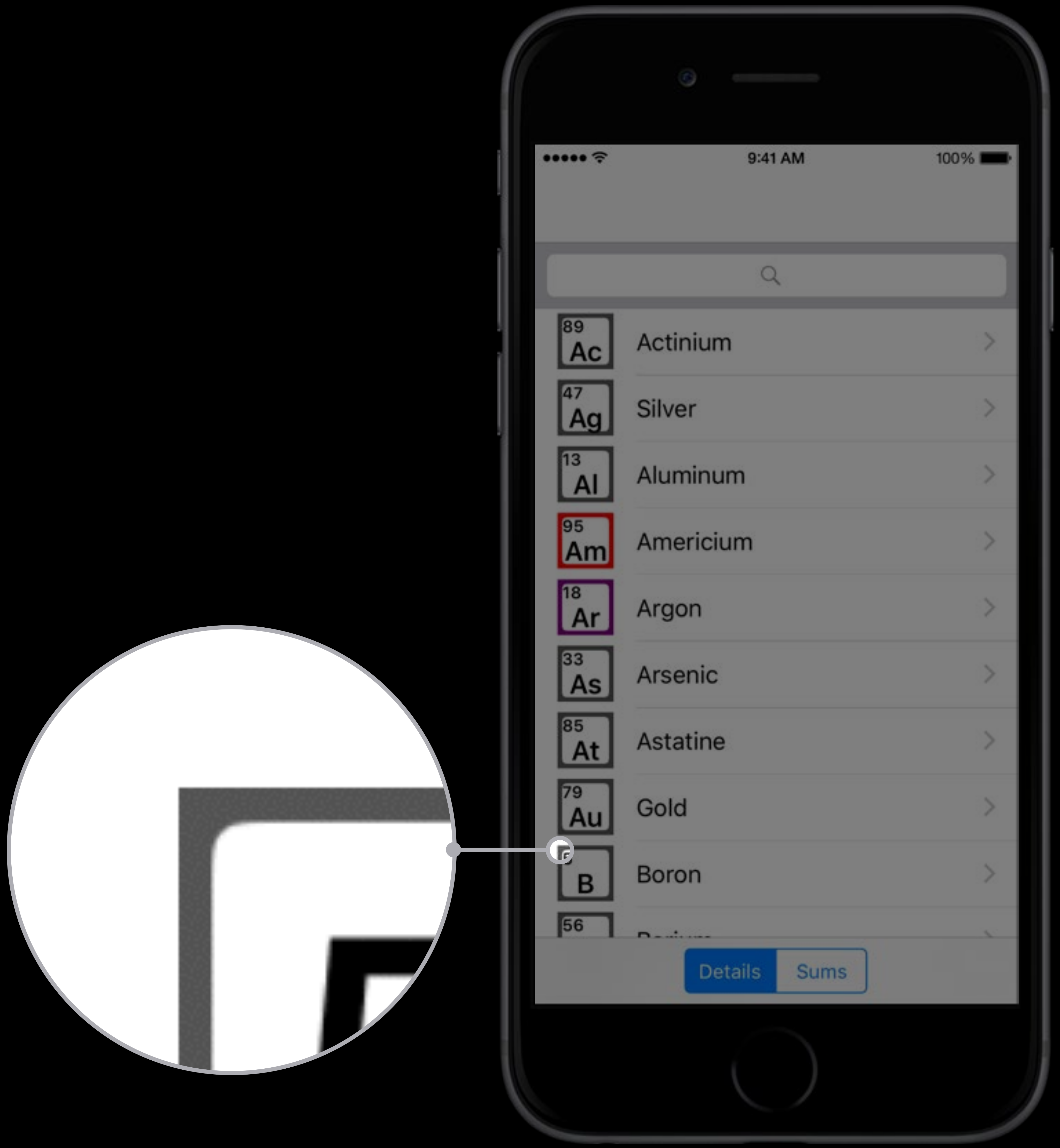
Modernizing The Elements UI

Boasting 120% daily intake of vitamin Swift.

Partially Rounded Corners



Partially Rounded Corners

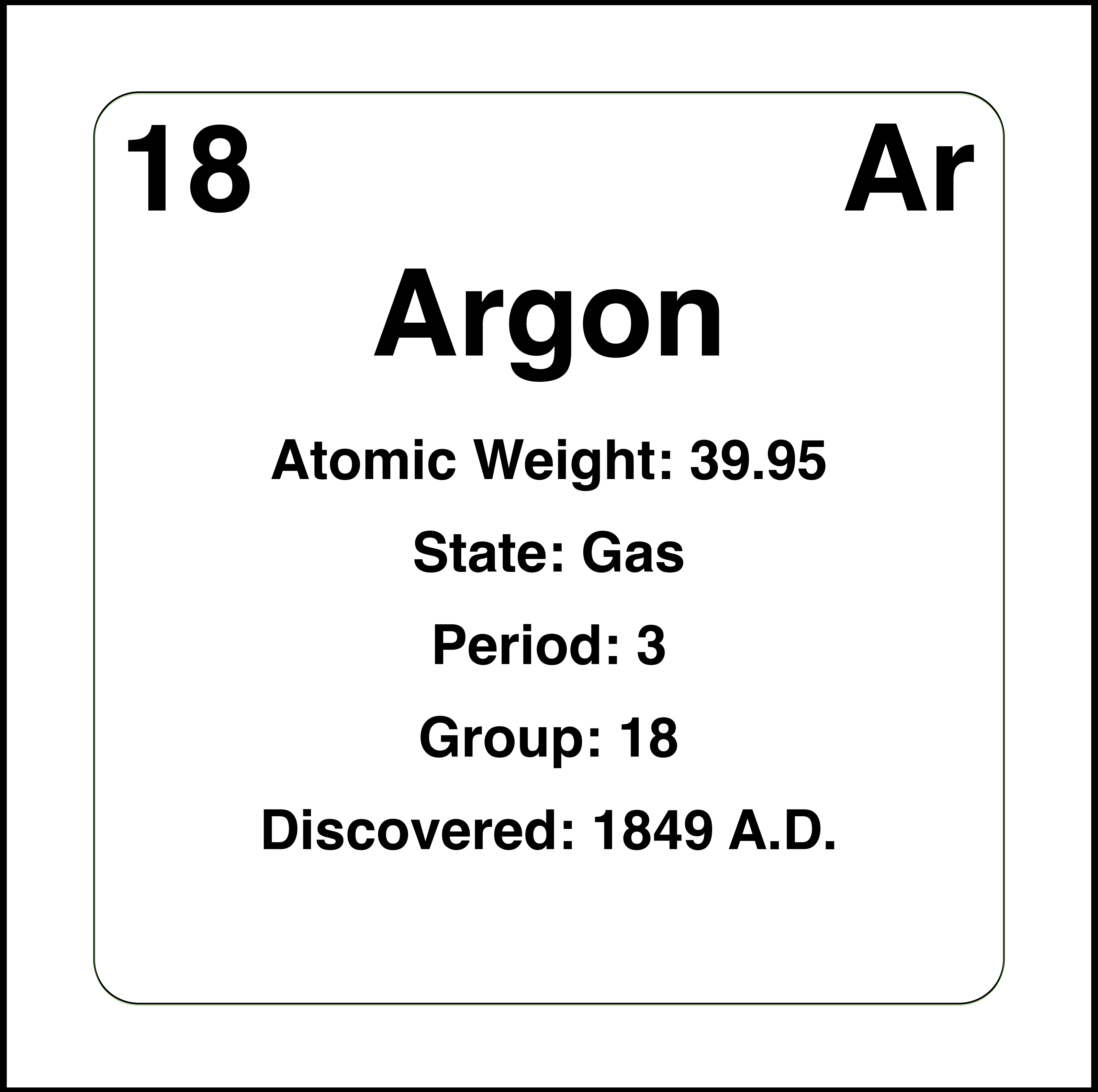


Swift Structs

With Playground prototyping

Global utility functions: shake ‘em off.

Bezier Path


18 Ar

Argon

Atomic Weight: 39.95

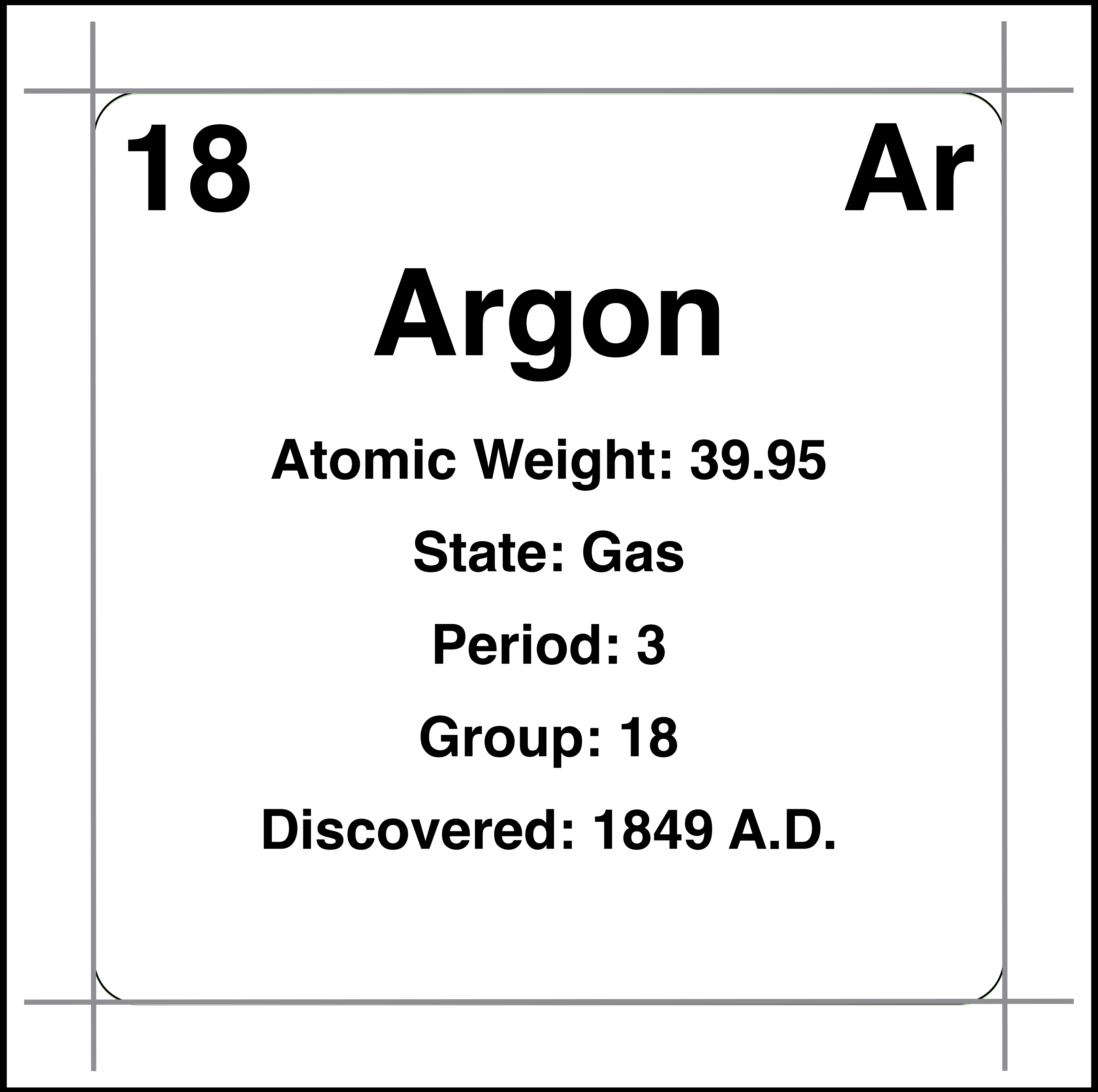
State: Gas

Period: 3

Group: 18

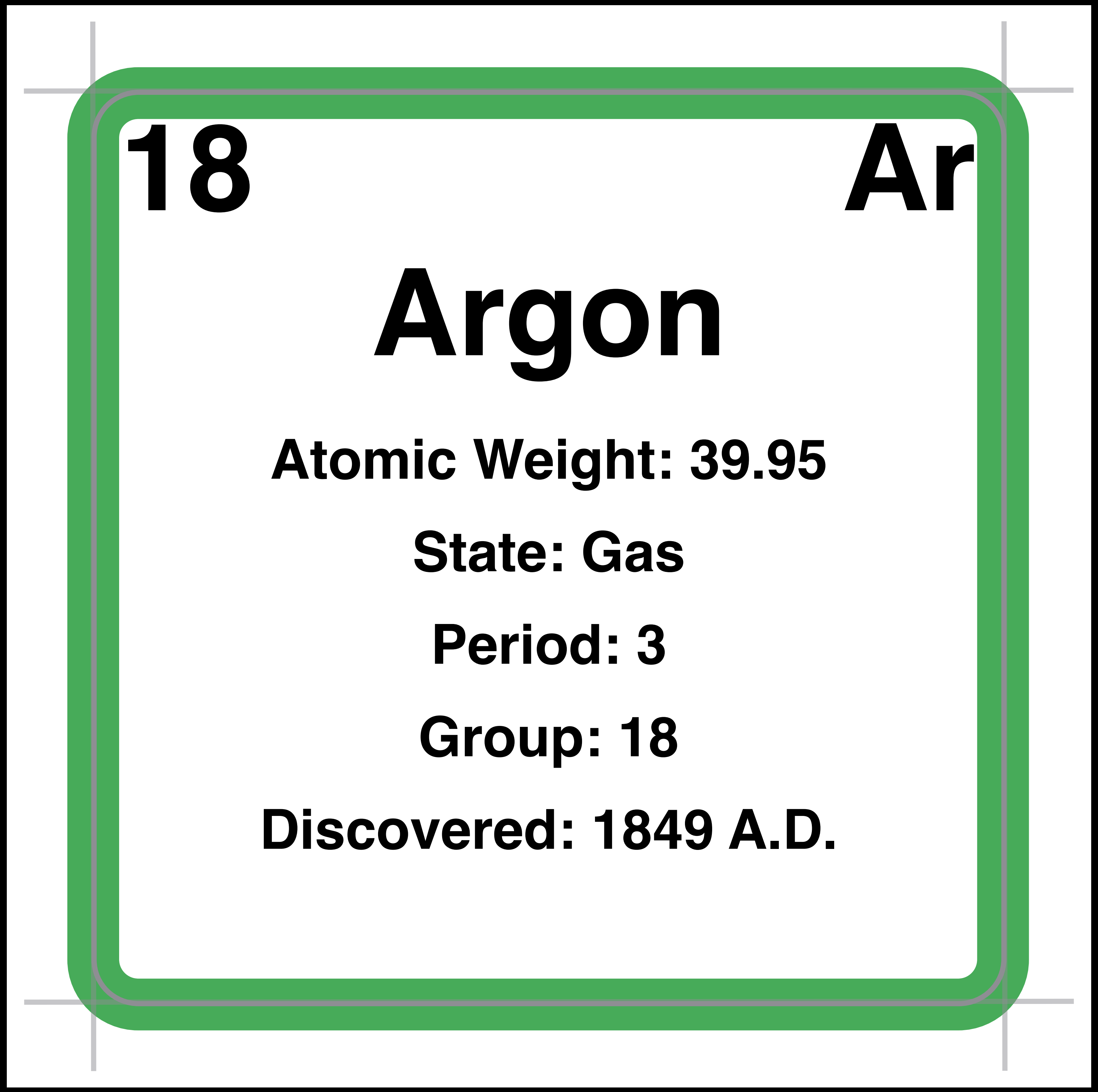
Discovered: 1849 A.D.

Bounding Rectangle



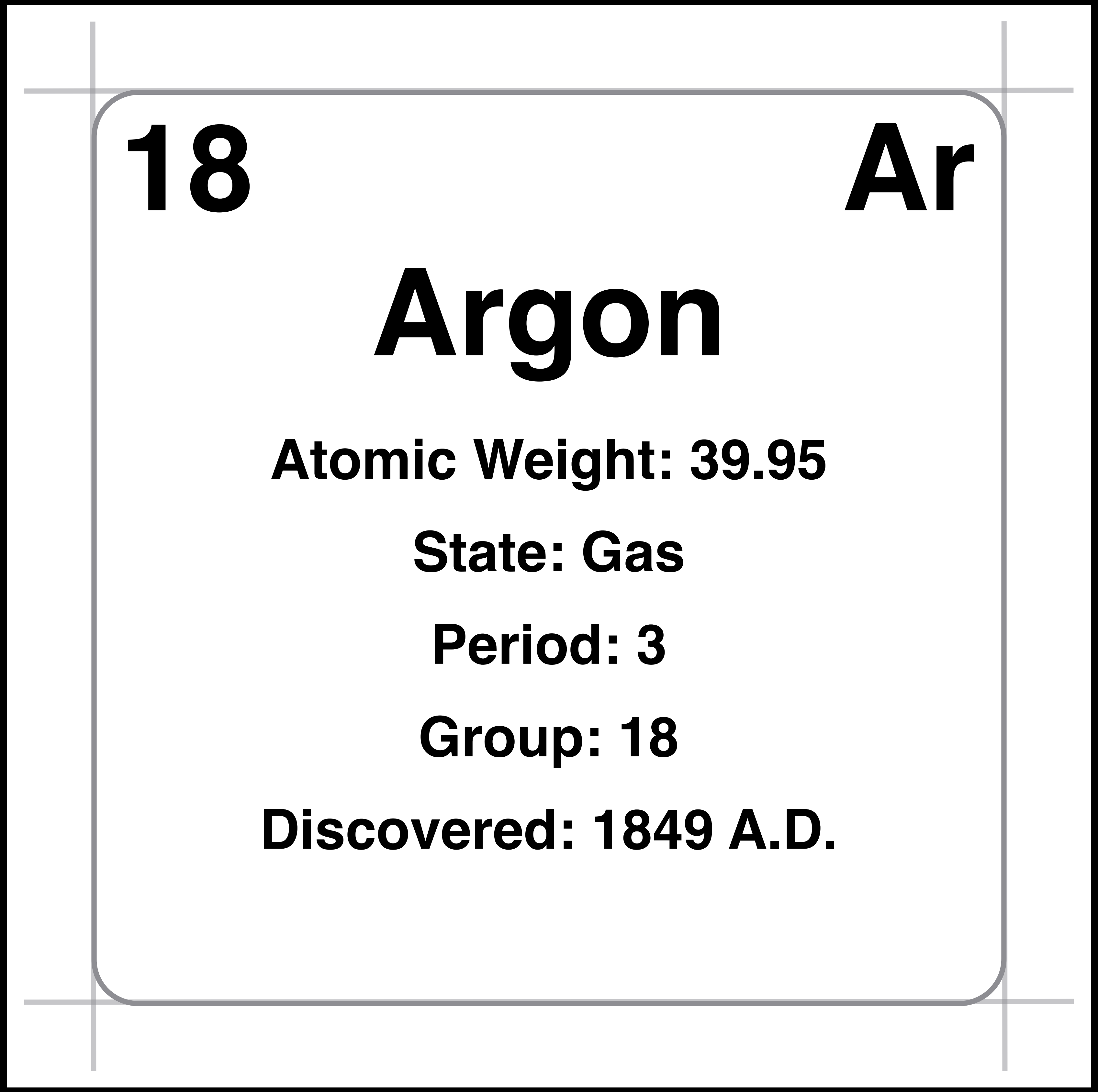
Draw Stroke

Stroke's midpoint is the Bezier Path



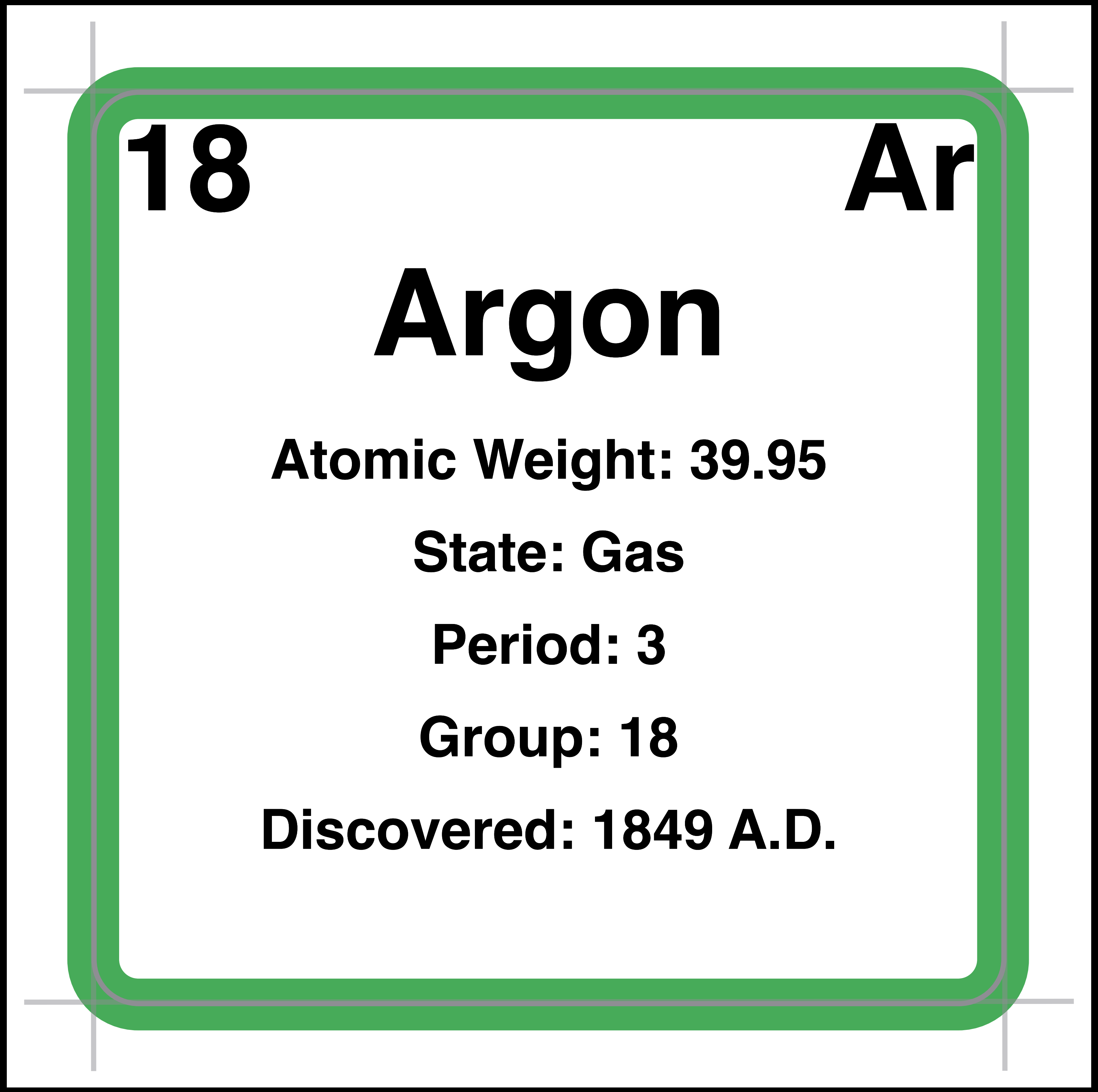
Draw Stroke

Stroke's midpoint is the Bezier Path



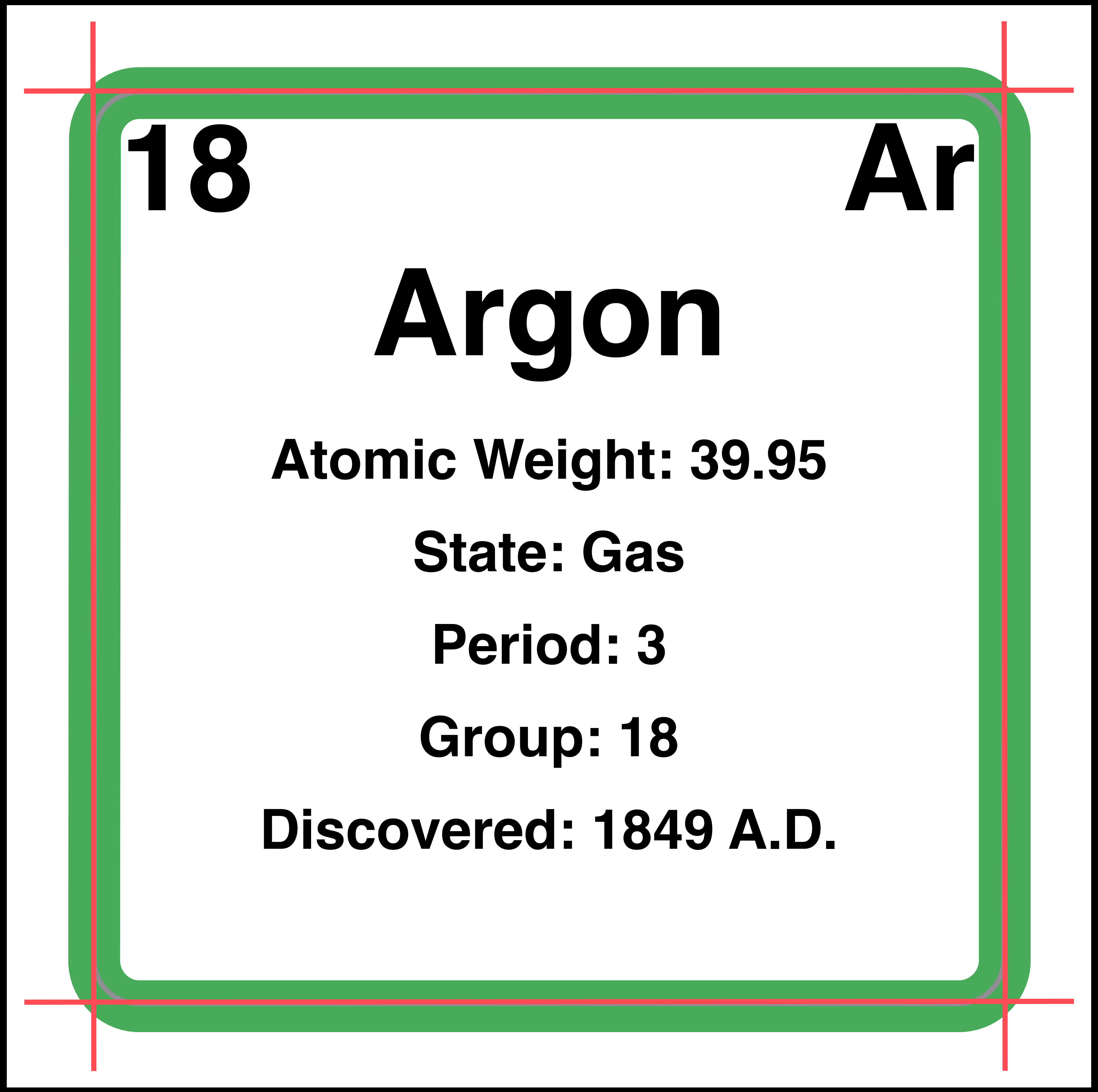
Draw Stroke

Clipping



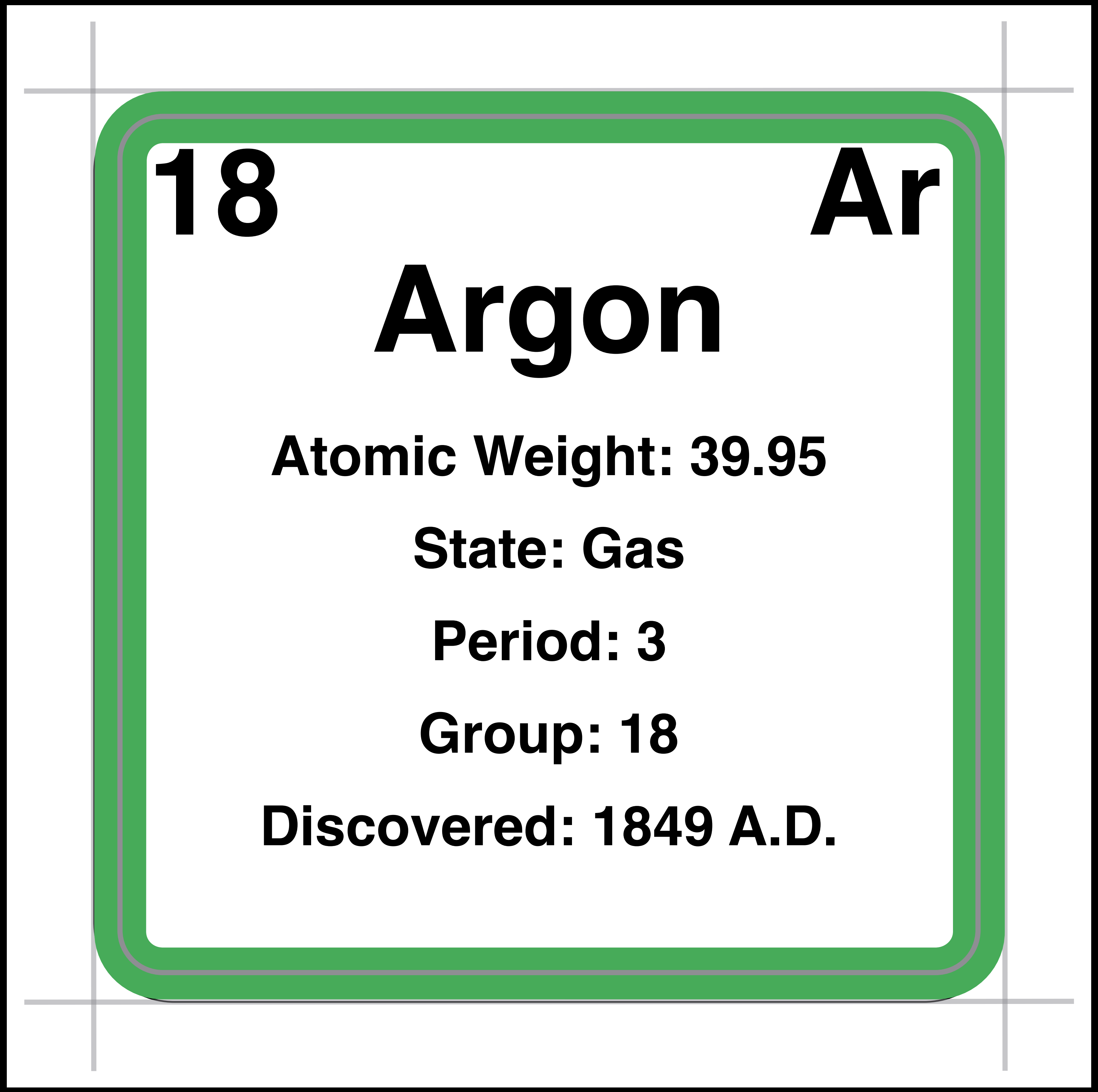
Draw Stroke

Clipping



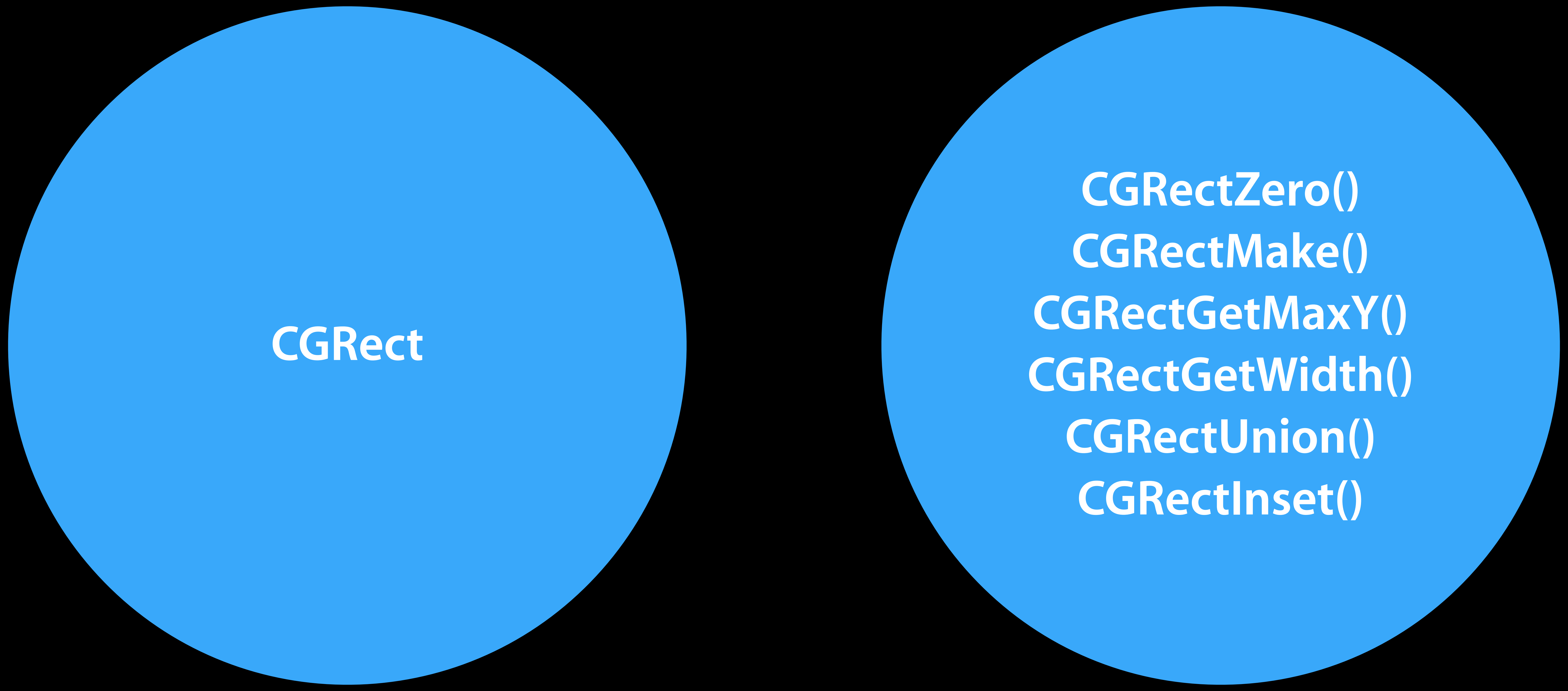
Solution

Inset the Bezier Path



Utility Functions

Core Graphics API

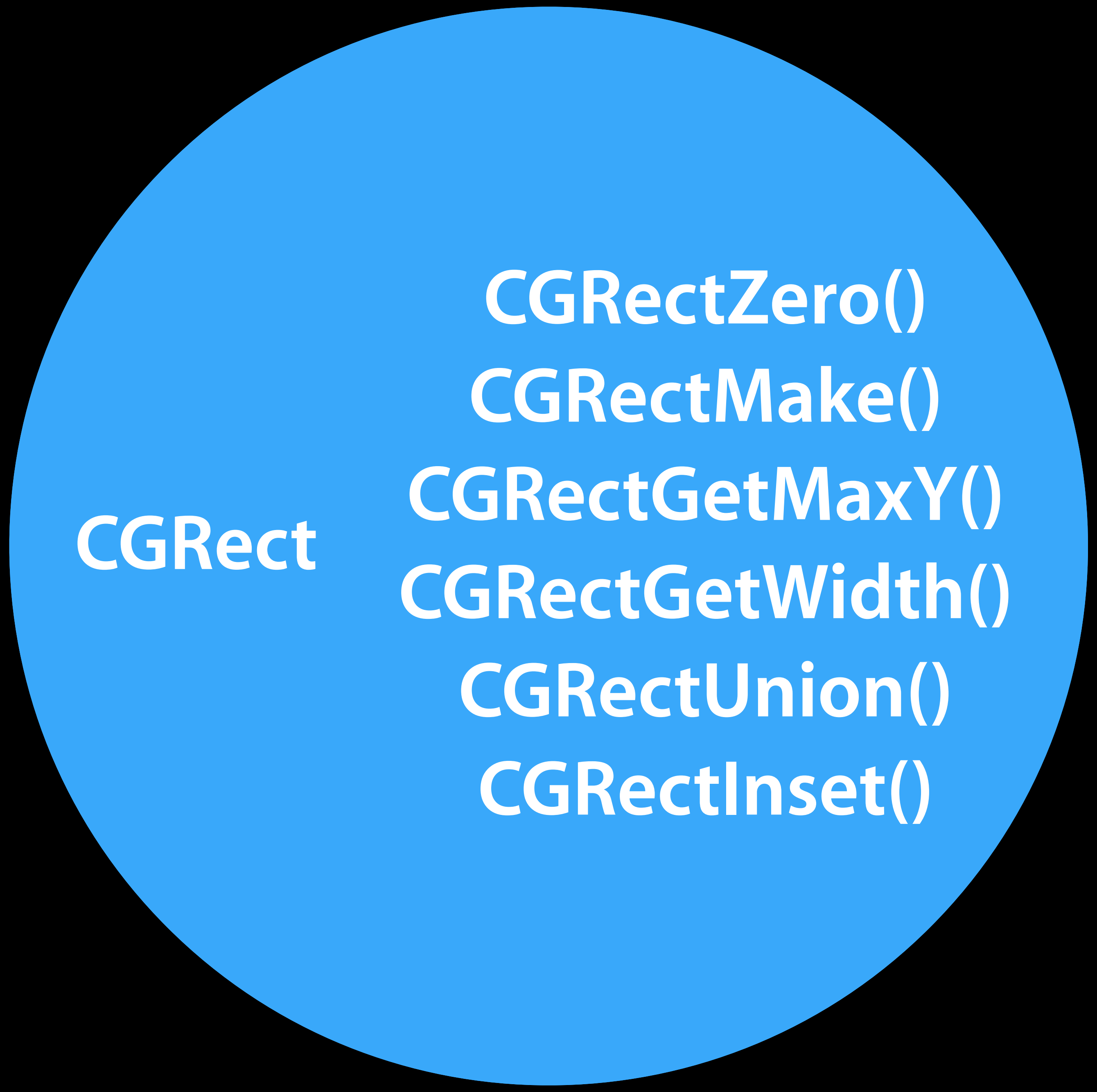


Struct

Global Utility Functions

Utility Functions

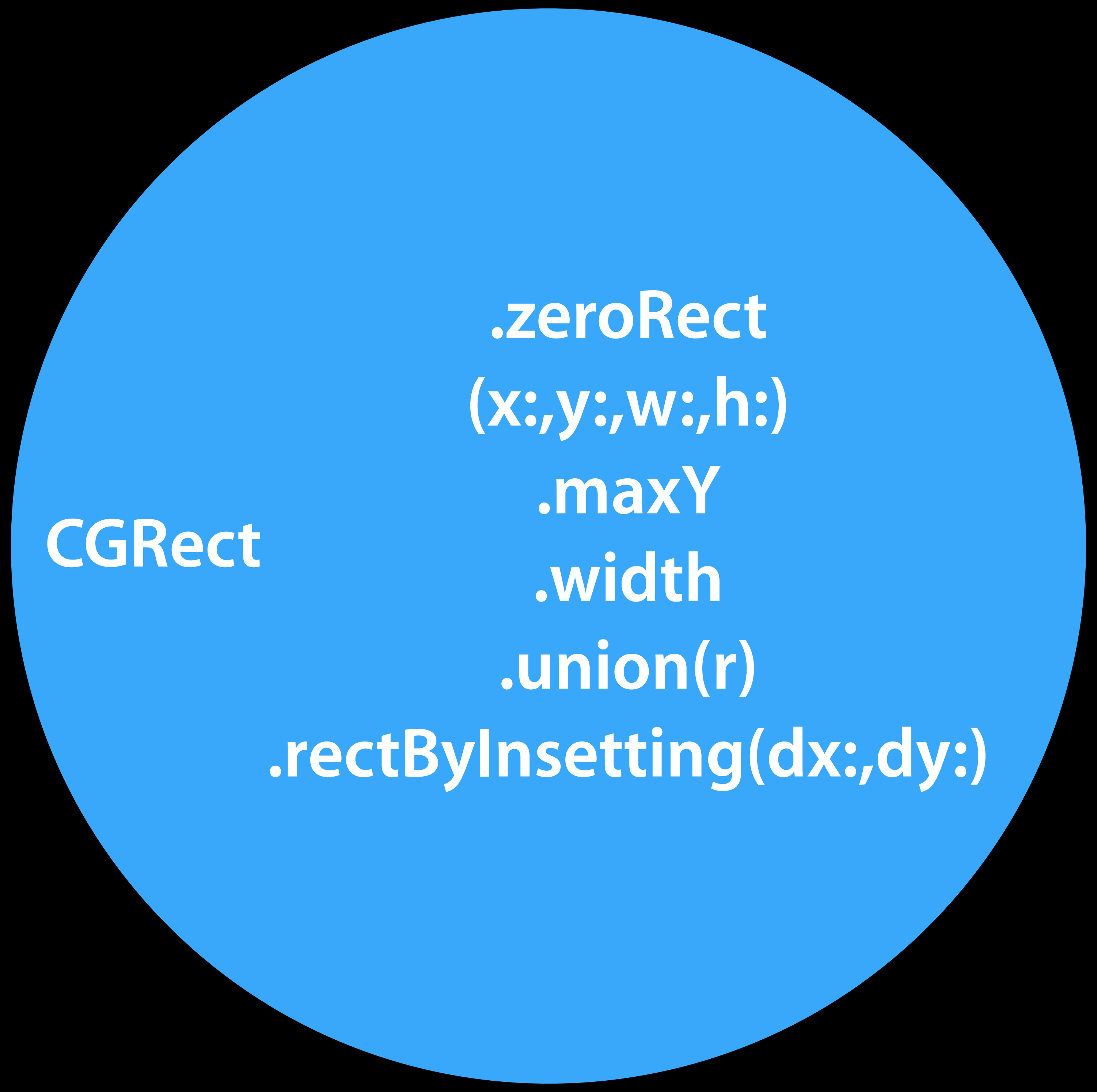
Core Graphics API



Struct Methods

Utility Functions

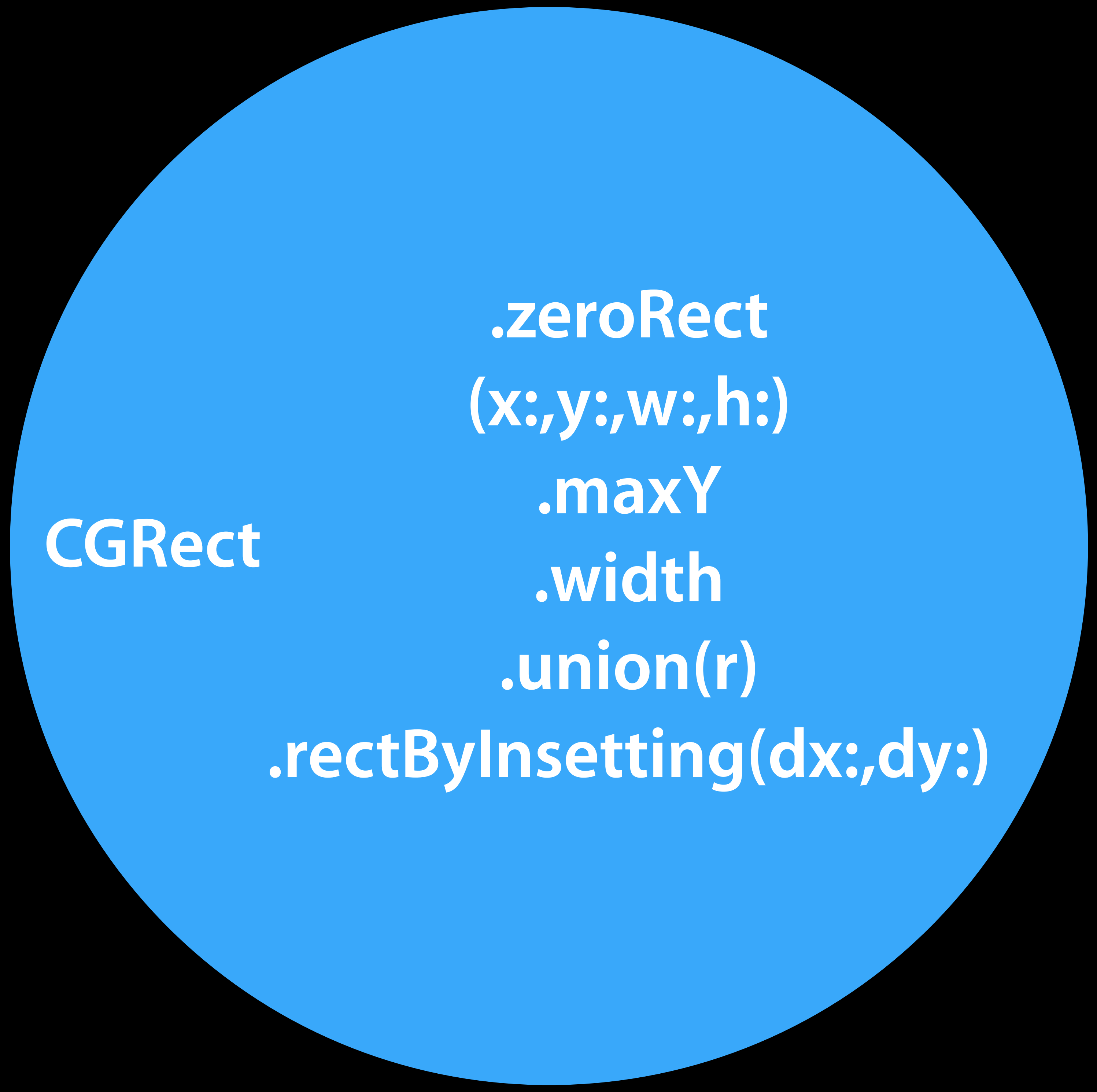
Core Graphics API



Struct   Initializers + Methods

Utility Functions

Core Graphics API

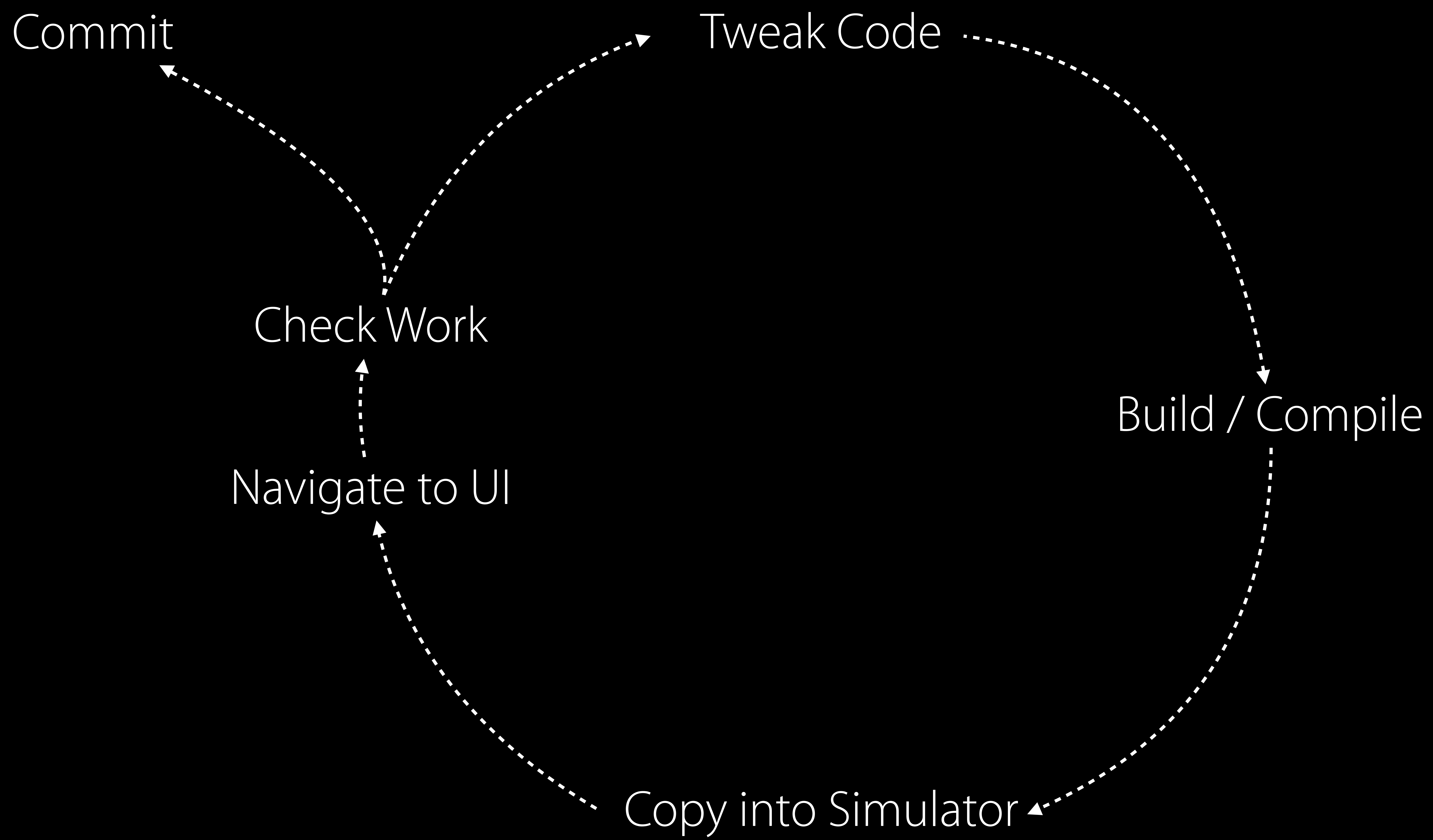


Struct Initializers + Methods

- Consistent
- Initializer Syntax
- Encapsulation
- Code Completion
- API Discovery

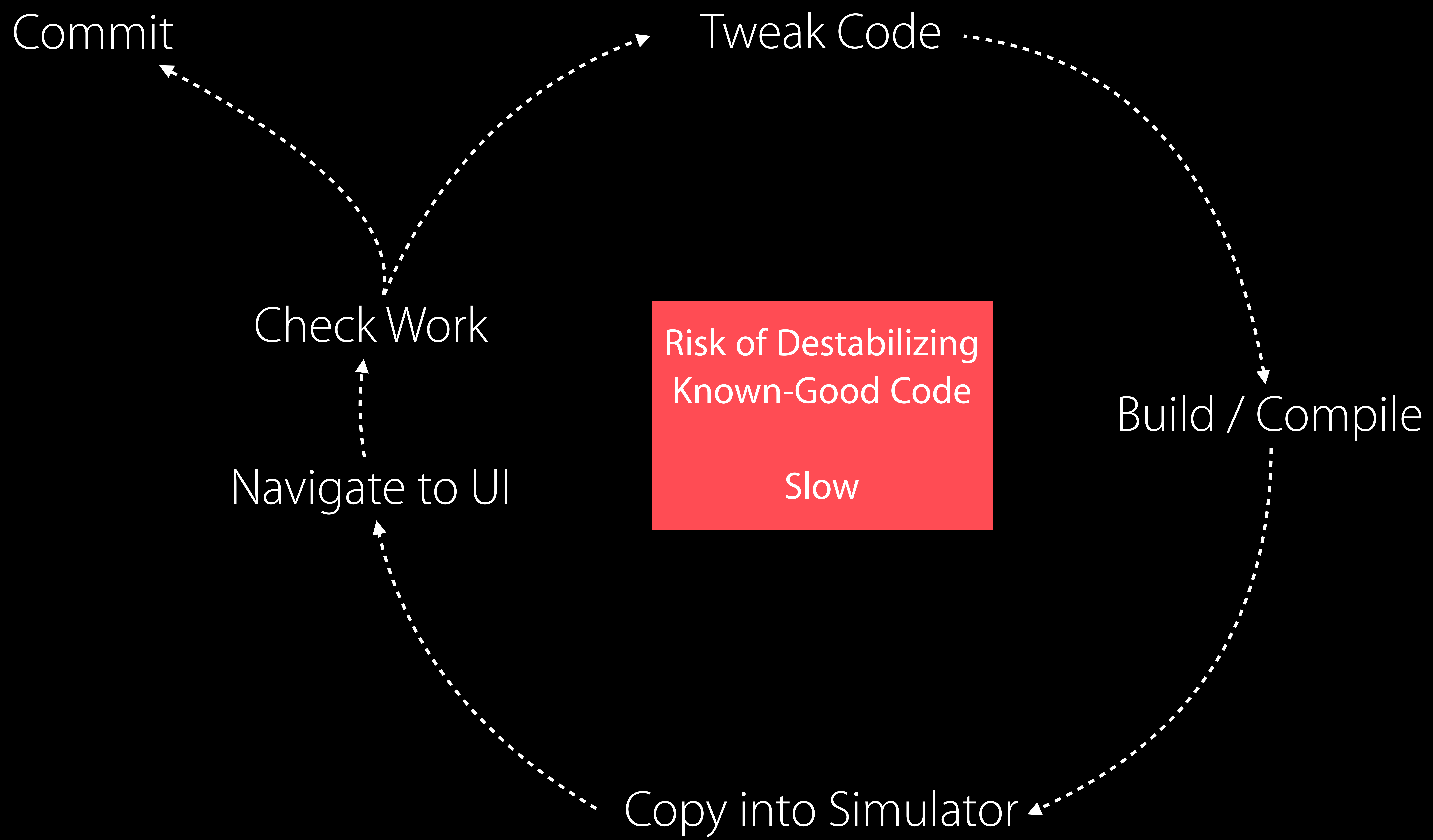
Playgrounds for Prototyping

Tweak-build loop



Playgrounds for Prototyping

Tweak-build loop



Risk of Destabilizing Known-Good Code

Slow

Playgrounds for Prototyping

The plan

Reduce roundtrip time

Iterative + experimental development

$$\begin{tikzcd}
& \text{Tweak Code} \arrow[d, shift left, dashed] \\
& \text{Check Work} \arrow[u, shift left, dashed] \arrow[dl, dashed] \\
{} &
\end{tikzcd}$$

Demo

Struct Methods

With Playground Prototyping

Availability: the Next Generation

Put down your `if respondsToSelector()` and step away from Xcode.

Runtime Interrogation

The classic way

```
if ([UIImagePickerController instancesRespondToSelector:
      @selector(availableCaptureModesForCameraDevice:)]) {
  // Method is available for use.
}
else {
  // Method is not available.
}
```

Availability Based on OS Version

The officially endorsed, modern way

```
if #available(iOS 8.3, *) {
     viewController.modalPresentationStyle = .Popover
     if let popoverPC = viewController.popoverPresentationController {
         let cell = tableView.cellForRowAtIndexPath(indexPath)
         popoverPC.sourceView = cell
         popoverPC.delegate = self
     }

     presentViewController(viewController, animated: true, completion: nil)
 } else {
     // Earlier than iOS 8.3 APIs
     navigationController?.pushViewController(viewController, animated: true)
 }
```

Compile-Time Availability Checking

```
viewController.modalPresentationStyle = .Popover
        if let popoverPC = viewController.popoverPresentationController {
            let cell = tableView.cellForRowAtIndexPath(indexPath)
            popoverPC.sourceView = cell
            popoverPC.delegate = self
        }
```

Compile-Time Availability Checking

```
error: ‘popoverPresentationController’ is
   only available on iOS 8.0 or newer
```

```
viewController.modalPresentationStyle = .Popover
        if let popoverPC = viewController.popoverPresentationController {
            let cell = tableView.cellForRowAtIndexPath(indexPath)
            popoverPC.sourceView = cell
            popoverPC.delegate = self
        }
```

Demo

Availability

APIs must be this tall to ride.

Availability

New in Swift 2

Compilation-time checking of API Availability

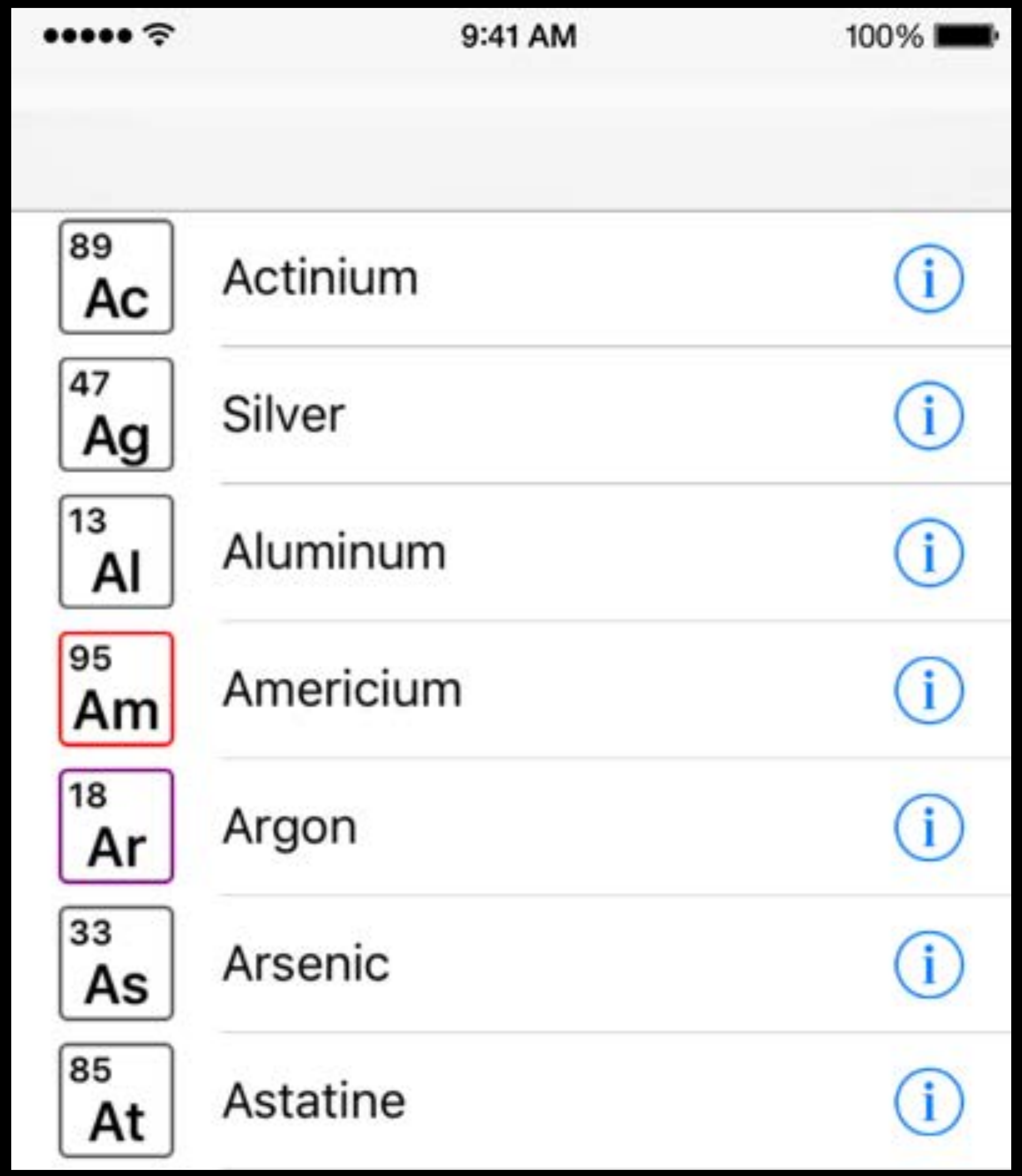
Runtime checks inserted automatically

Swift in Practice Presidio Thursday 2:30PM

Live Search

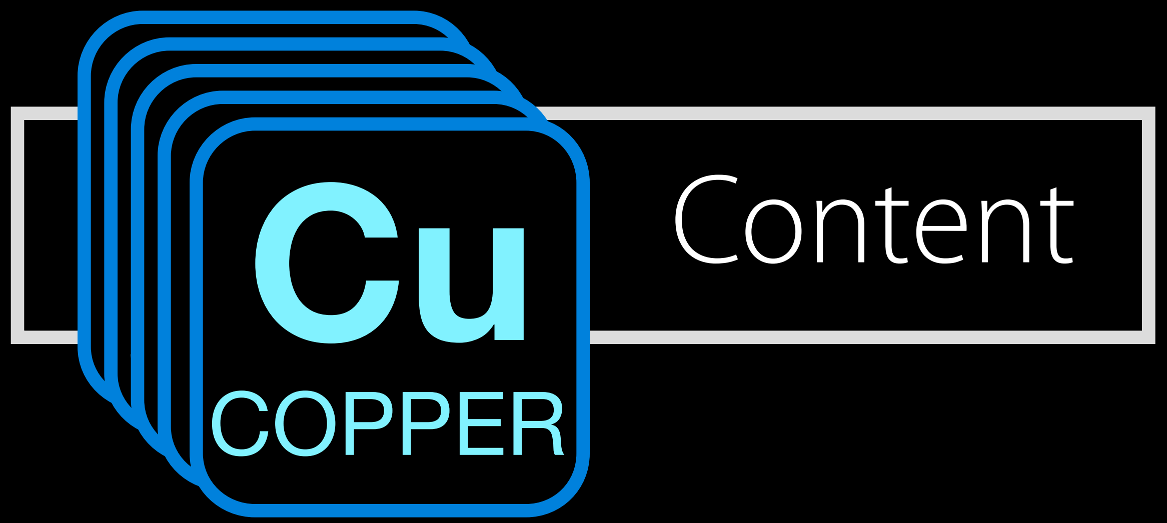
Filtering elements since antiquity.

TableView

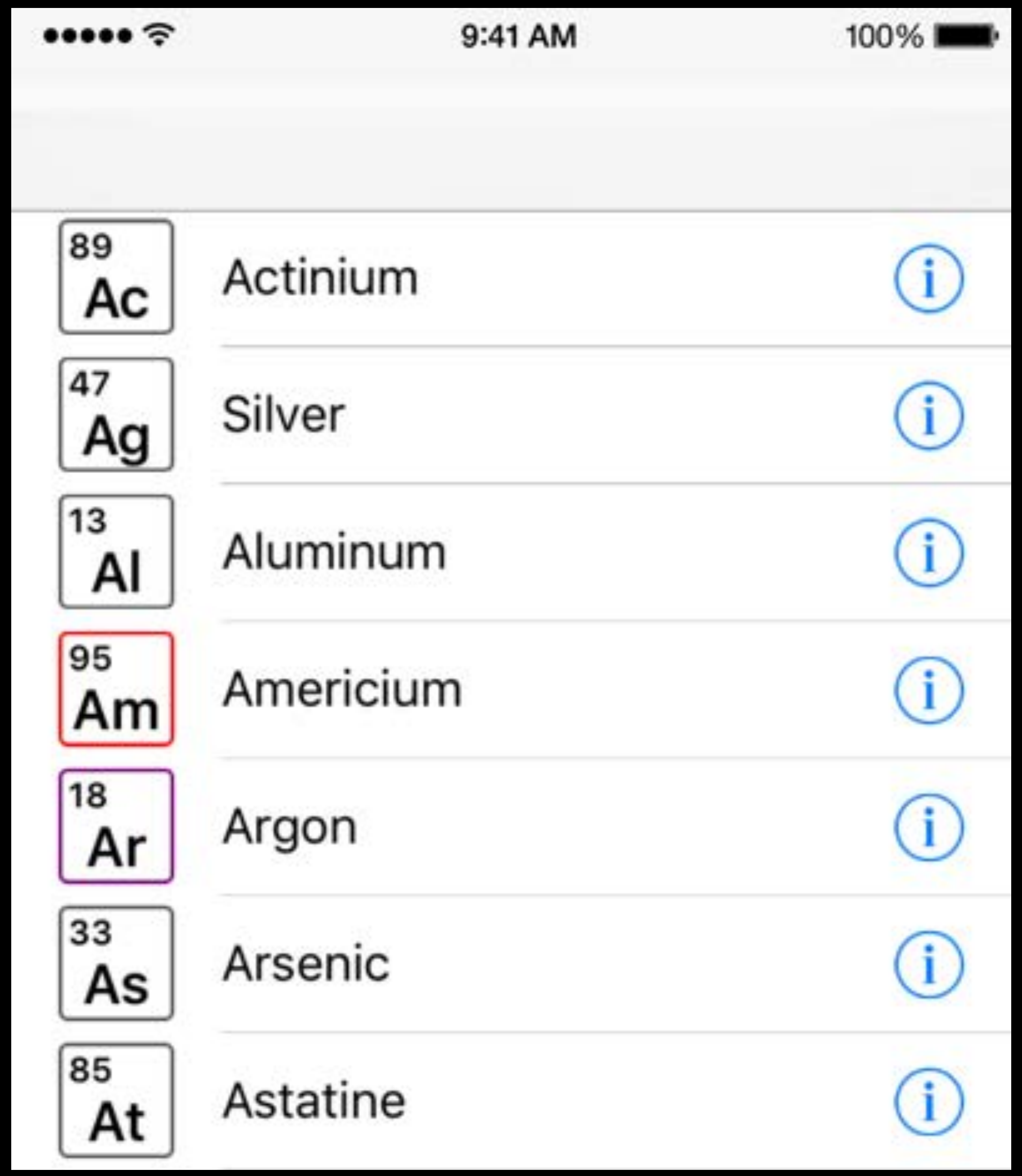


89 Ac	Actinium
47 Ag	Silver
13 Al	Aluminum
95 Am	Americium
18 Ar	Argon
33 As	Arsenic
85 At	Astatine

View Controller



TableView



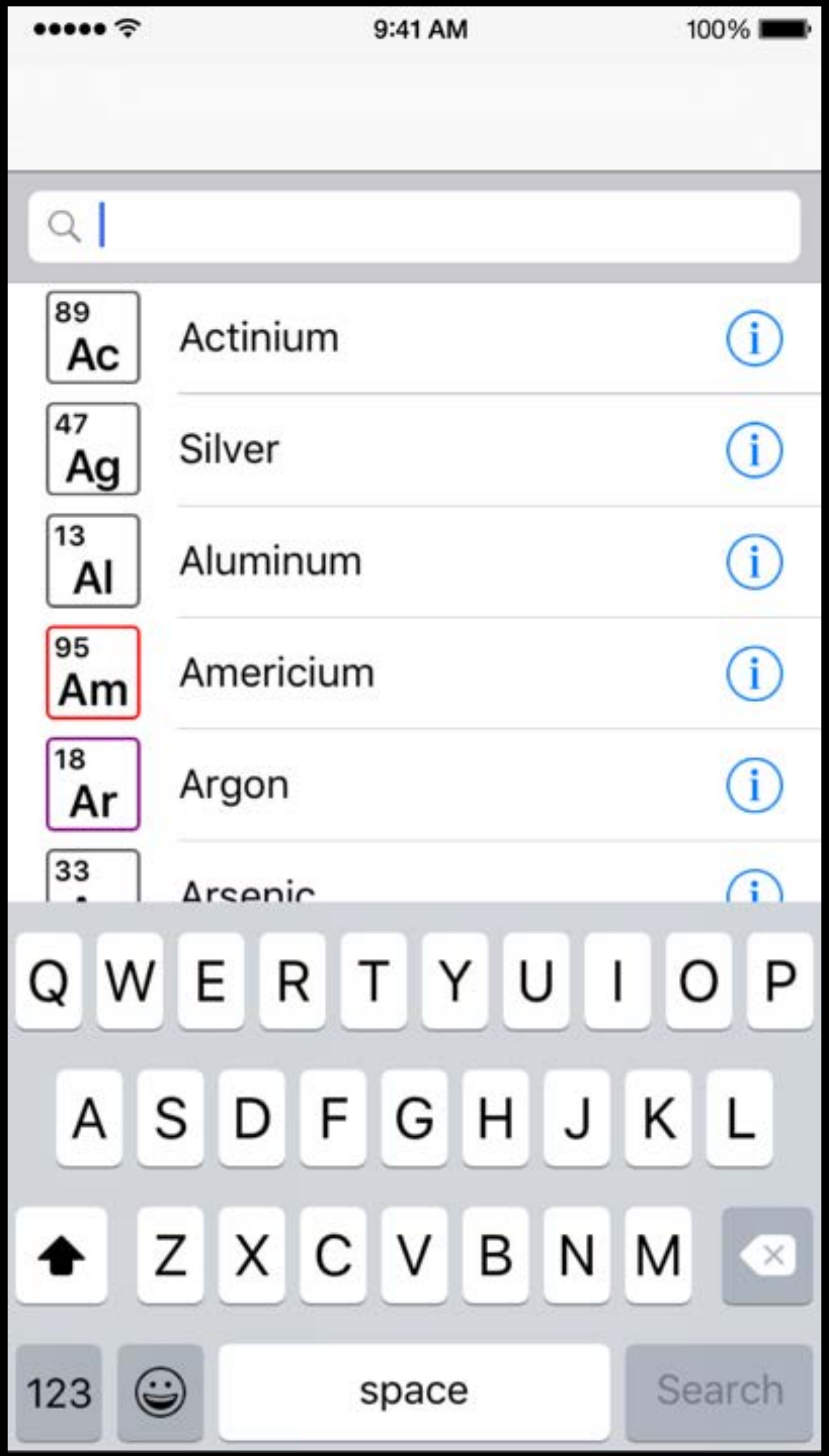
9:41 AM 100%

89 Ac	Actinium	ⓘ
47 Ag	Silver	ⓘ
13 Al	Aluminum	ⓘ
95 Am	Americium	ⓘ
18 Ar	Argon	ⓘ
33 As	Arsenic	ⓘ
85 At	Astatine	ⓘ

View Controller
<UITableViewDataSource>

Content

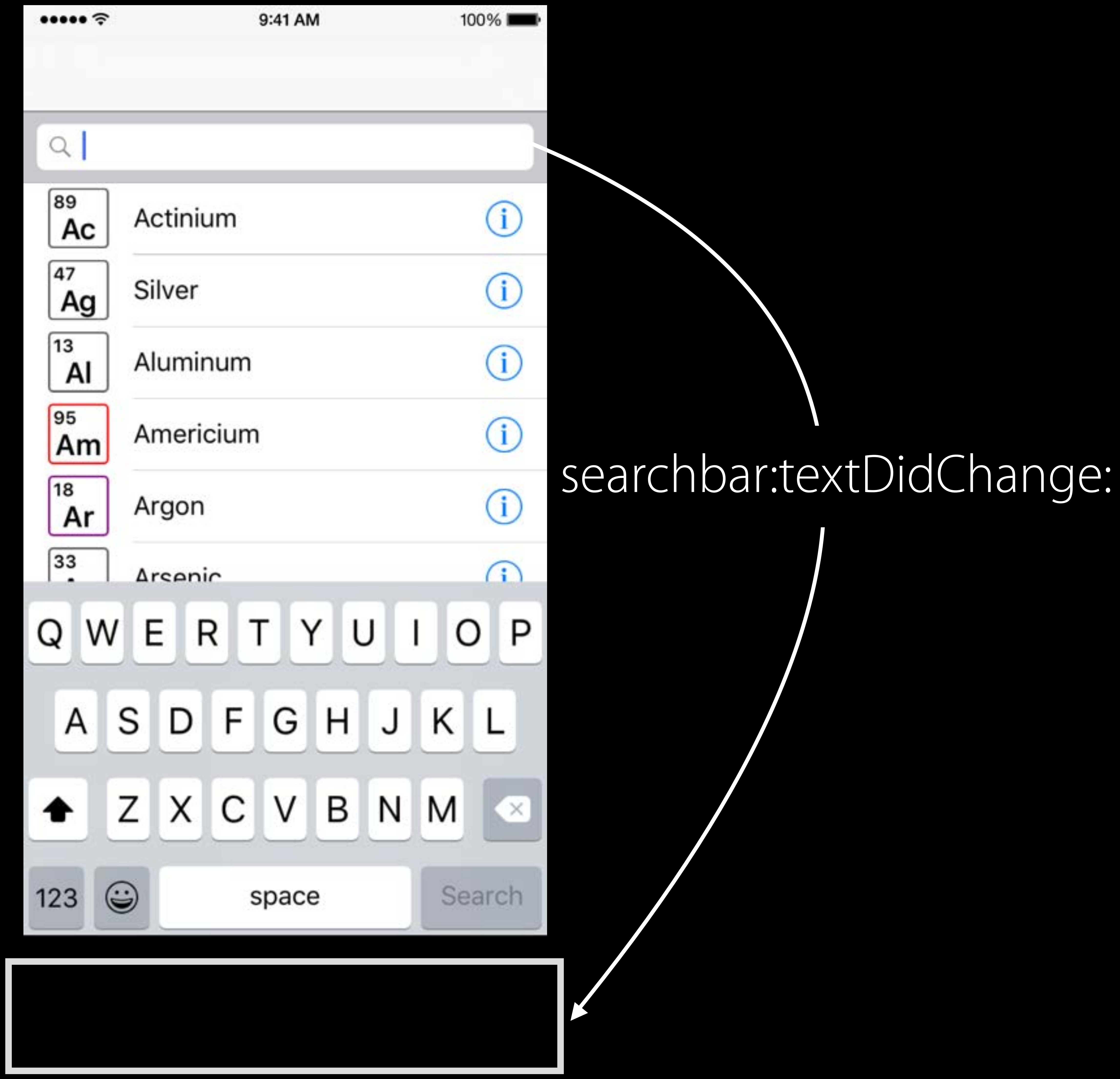
TableView



View Controller
<UITableViewDataSource, UISearchBarDelegate>

Content

TableView



View Controller
<UITableViewDataSource, UISearchBarDelegate>

Content

Filter

searchBar:textDidChange:

```
func searchBar(searchBar: UISearchBar, textDidChange searchText: String) {
   if searchText.isEmpty {
            content = atomicElements
         } else {
            content = atomicElements.filter{$0.name.hasPrefix(searchText)}
         }
      tableView?.reload()
 }
```

Filter

searchBar:textDidChange:

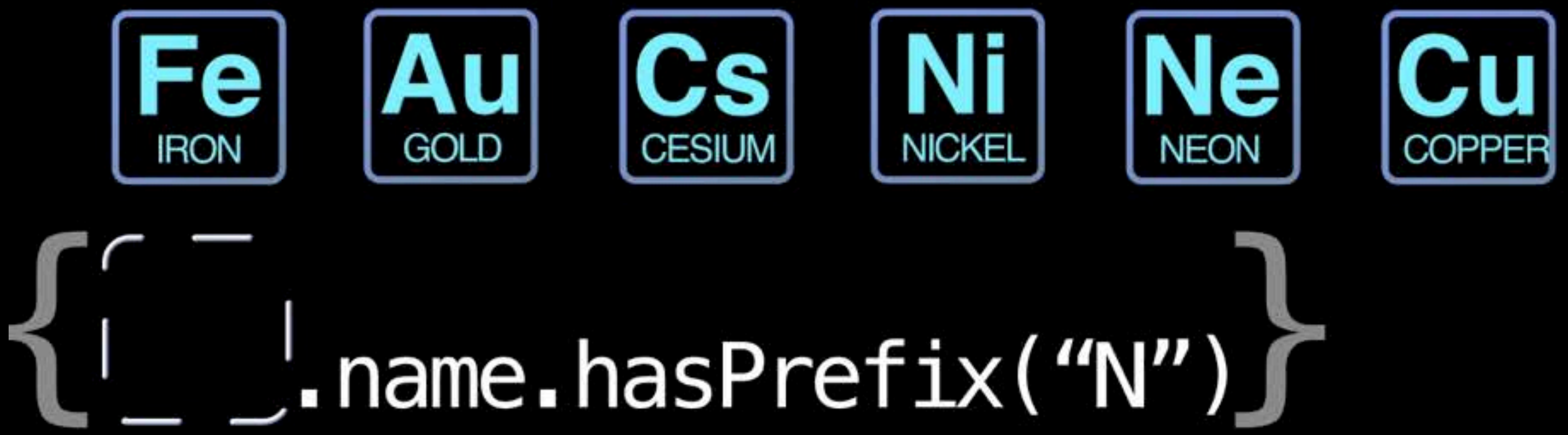
```
func searchBar(searchBar: UISearchBar, textDidChange searchText: String) {
   if searchText.isEmpty {
            content = atomicElements
        } else {
            content = atomicElements.filter{$0.name.hasPrefix(searchText)}
        }
      tableView?.reload()
 }
```

Filter

searchBar:textDidChange:

```
func searchBar(searchBar: UISearchBar, textDidChange searchText: String) {
   if searchText.isEmpty == true {
           content = atomicElements
        } else {
           content = atomicElements.filter{$0.name.hasPrefix(searchText)}
        }
      tableView?.reload()
 }
```

Original Array:



Closure:

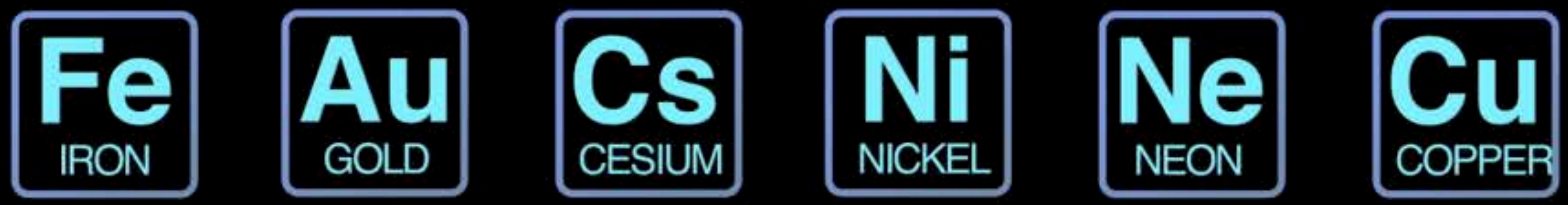
Filtered Array:

Filter

searchBar:textDidChange:

```
func searchBar(searchBar: UISearchBar, textDidChange searchText: String) {
  if searchText.isEmpty == true {
        content = atomicElements
      } else {
        content = atomicElements.filter{$0.name.hasPrefix(searchText)}
      }
    tableView?.reload()
 }
```

Original Array:



Closure:

```
{ .name.hasPrefix("N") }
```

Filtered Array:

Demo

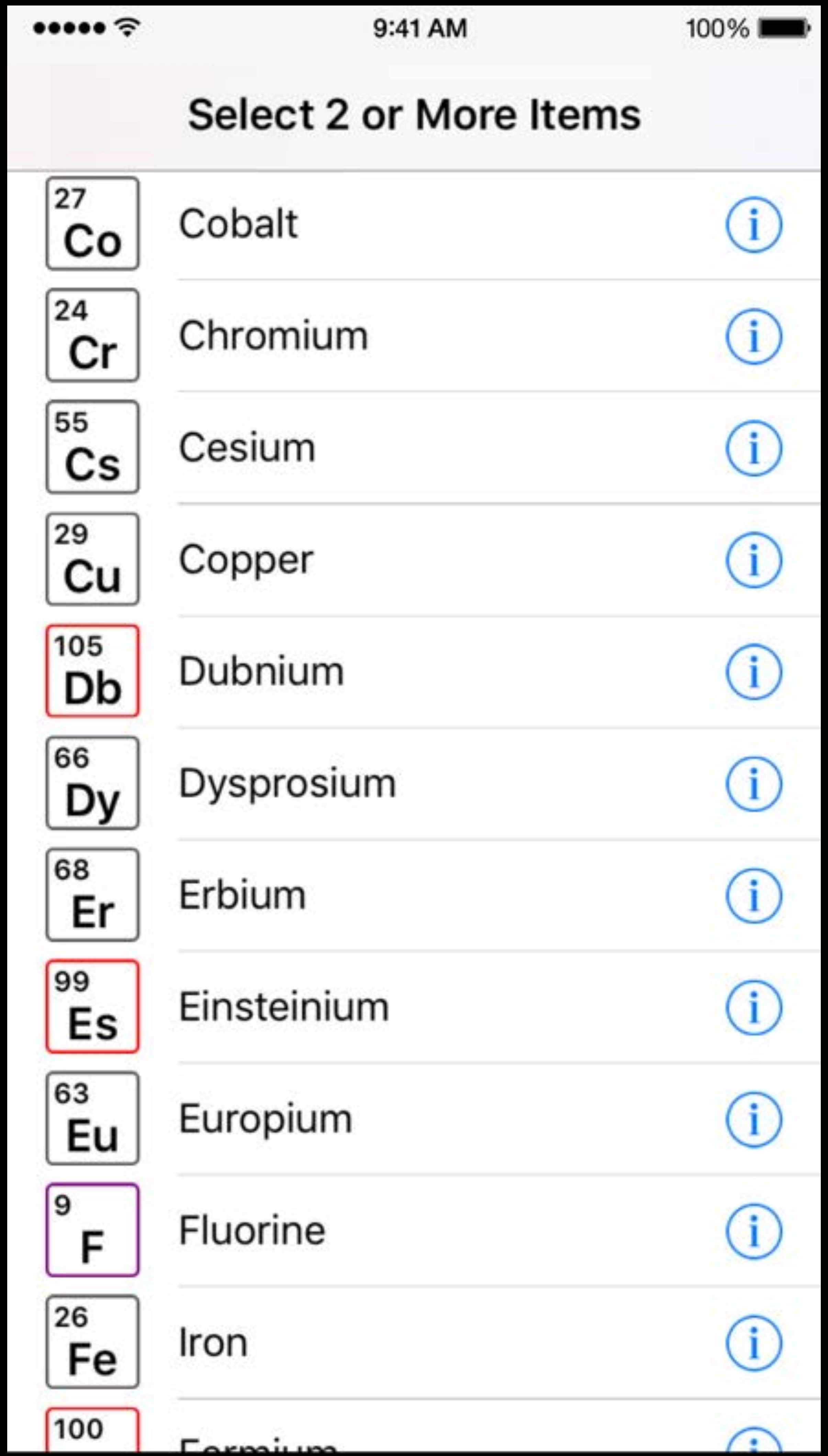
Filtering

Only the worthy may pass.

Sums of Atomic Weights

Introducing Map & Reduce, in harmony like Ebony & Ivory.

Sum of Atomic Weights



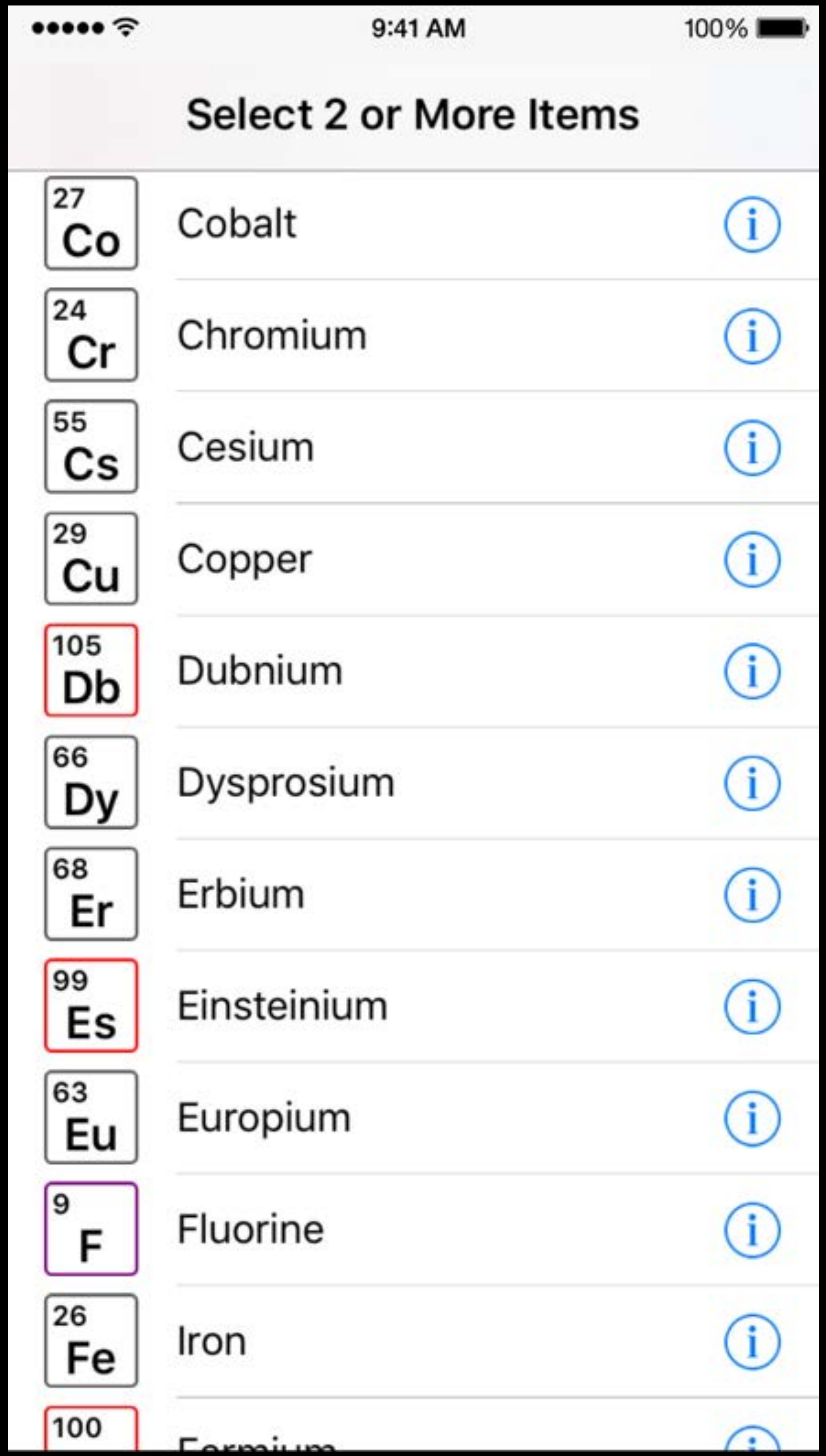
9:41 AM 100%

Select 2 or More Items

27 Co	Cobalt
24 Cr	Chromium
55 Cs	Cesium
29 Cu	Copper
105 Db	Dubnium
66 Dy	Dysprosium
68 Er	Erbium
99 Es	Einsteinium
63 Eu	Europium
9 F	Fluorine
26 Fe	Iron
100	

Sum of Atomic Weights

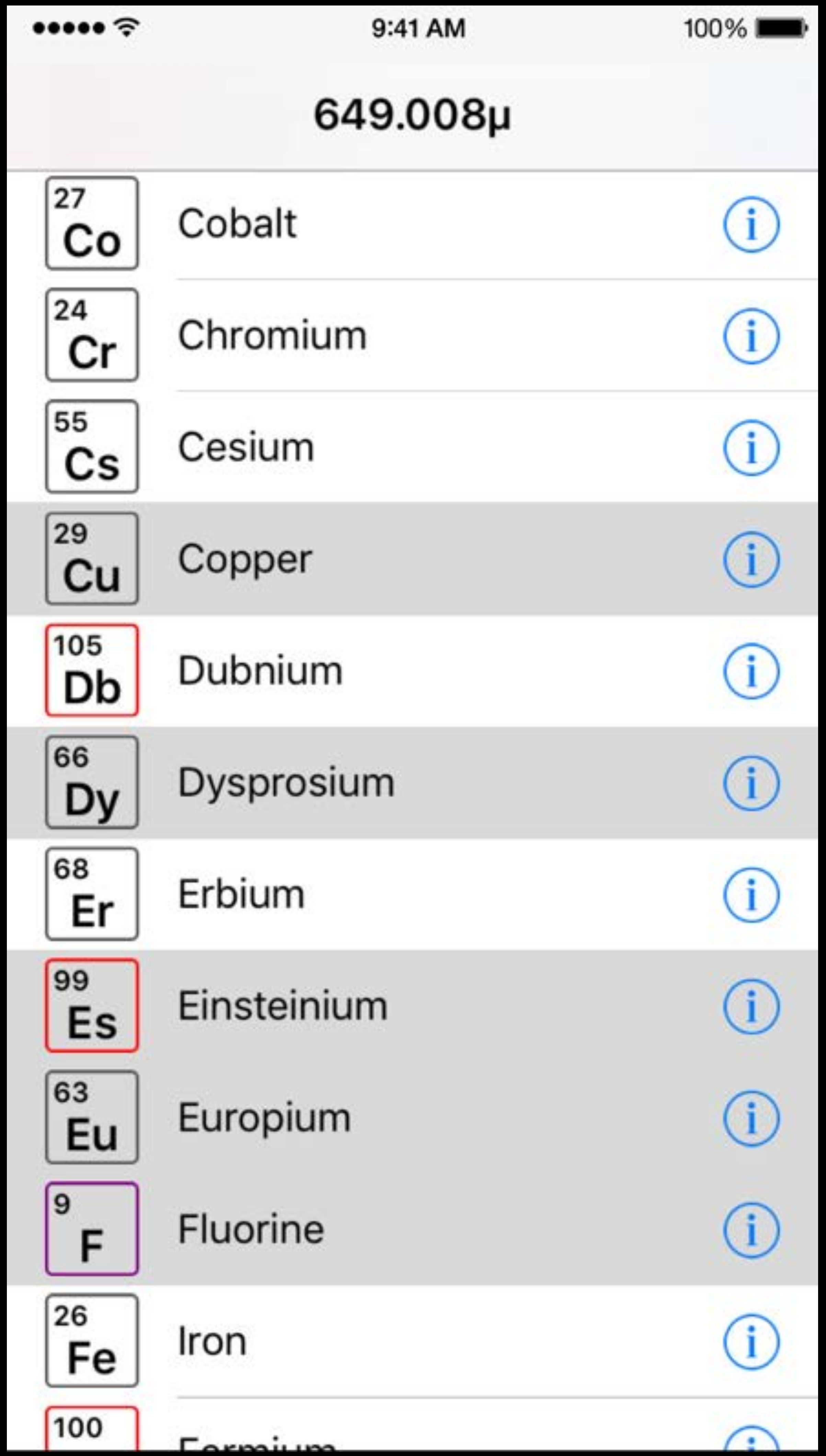
Two or more rows selected



9:41 AM 100%

Select 2 or More Items

27 Co	Cobalt	ⓘ
24 Cr	Chromium	ⓘ
55 Cs	Cesium	ⓘ
29 Cu	Copper	ⓘ
105 Db	Dubnium	ⓘ
66 Dy	Dysprosium	ⓘ
68 Er	Erbium	ⓘ
99 Es	Einsteinium	ⓘ
63 Eu	Europium	ⓘ
9 F	Fluorine	ⓘ
26 Fe	Iron	ⓘ
100		



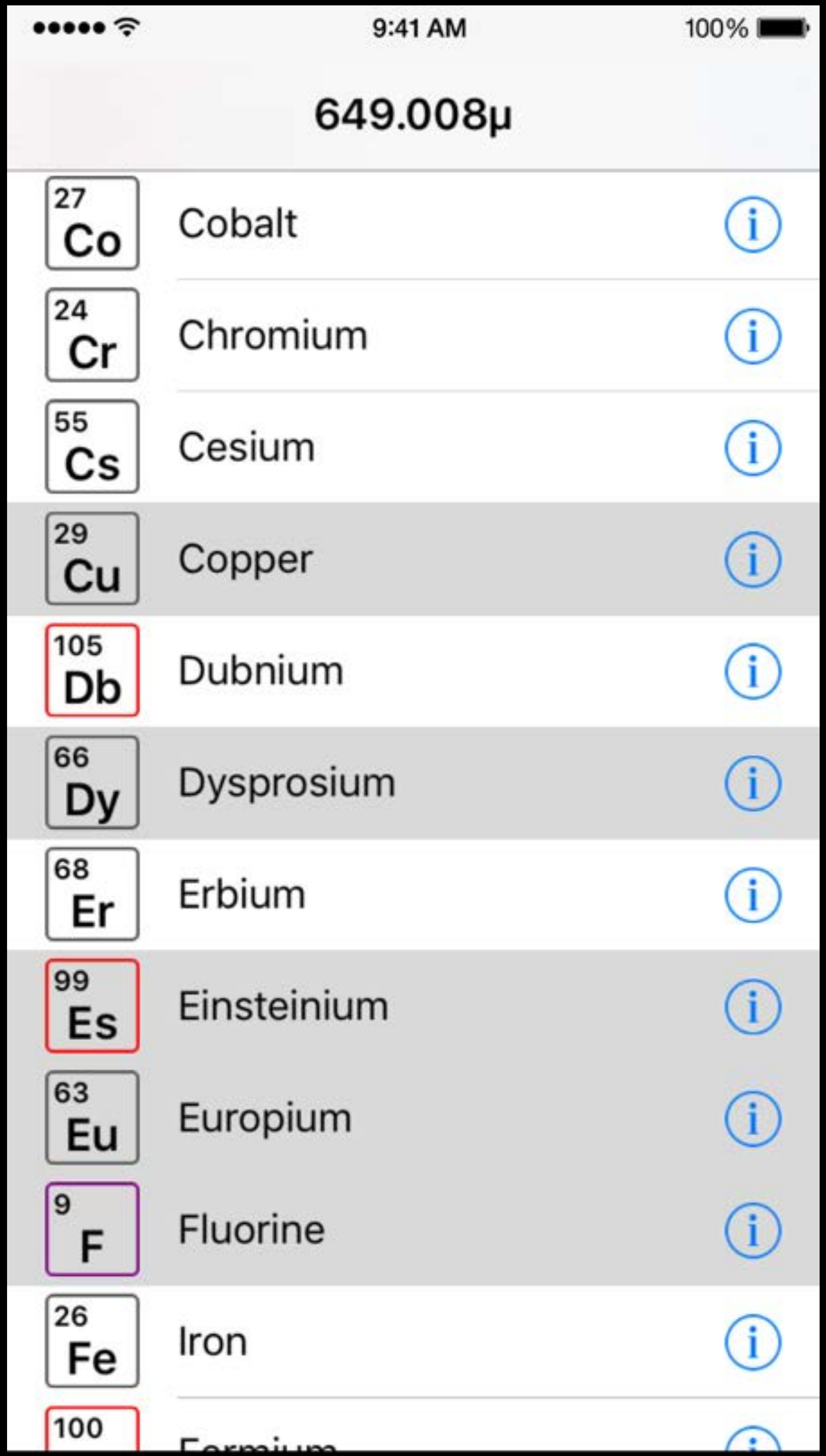
9:41 AM 100%

649.008μ

27 Co	Cobalt	ⓘ
24 Cr	Chromium	ⓘ
55 Cs	Cesium	ⓘ
29 Cu	Copper	ⓘ
105 Db	Dubnium	ⓘ
66 Dy	Dysprosium	ⓘ
68 Er	Erbium	ⓘ
99 Es	Einsteinium	ⓘ
63 Eu	Europium	ⓘ
9 F	Fluorine	ⓘ
26 Fe	Iron	ⓘ
100		

Intermediate Objective

Array of selected atomic elements

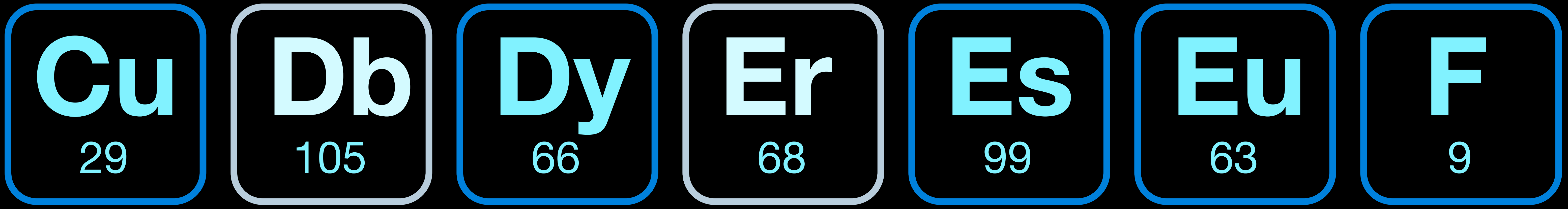


9:41 AM 100%

649.008μ

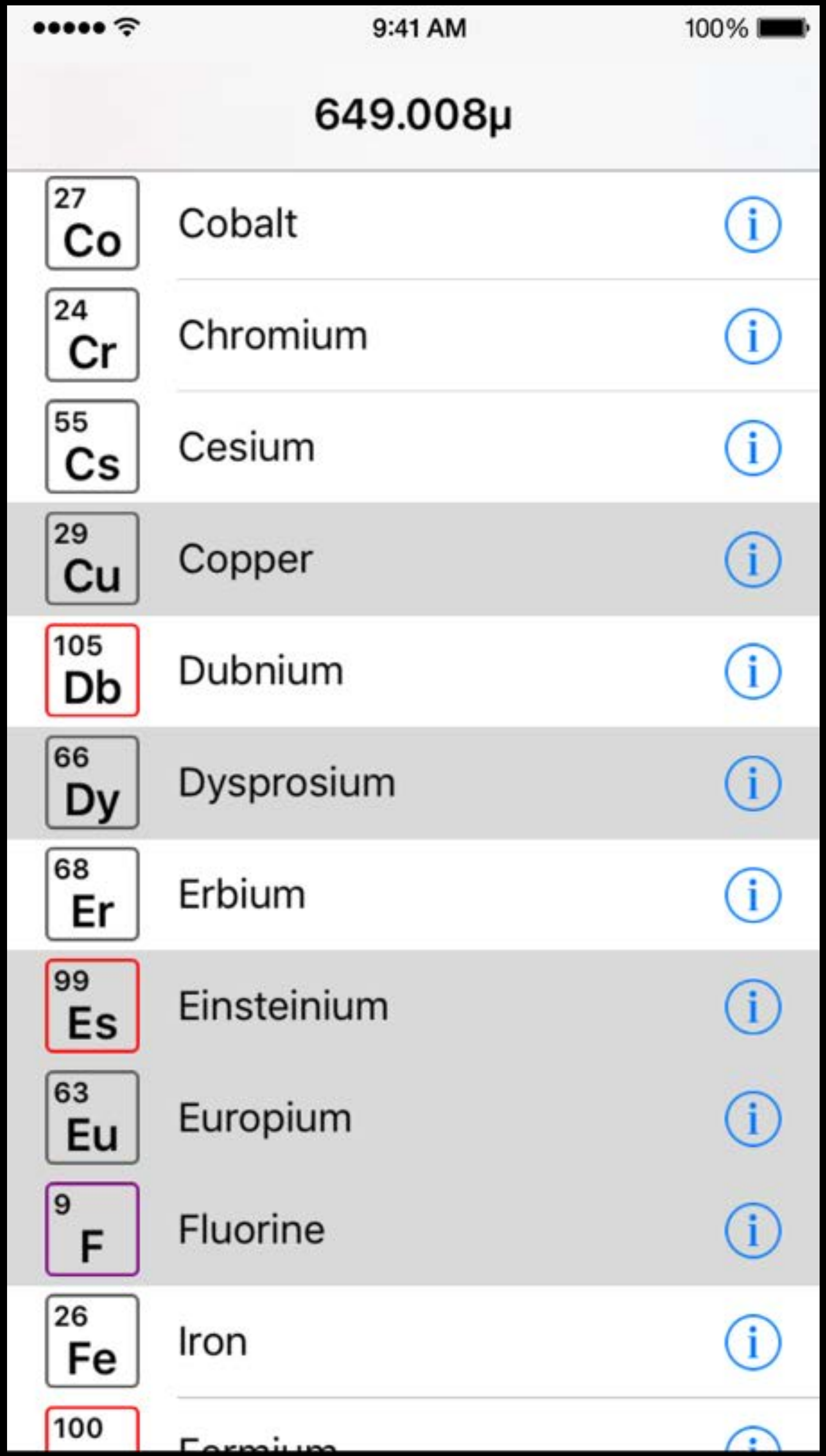
27 Co	Cobalt	ⓘ
24 Cr	Chromium	ⓘ
55 Cs	Cesium	ⓘ
29 Cu	Copper	ⓘ
105 Db	Dubnium	ⓘ
66 Dy	Dysprosium	ⓘ
68 Er	Erbium	ⓘ
99 Es	Einsteinium	ⓘ
63 Eu	Europium	ⓘ
9 F	Fluorine	ⓘ
26 Fe	Iron	ⓘ
100		

```
content
```



Intermediate Objective

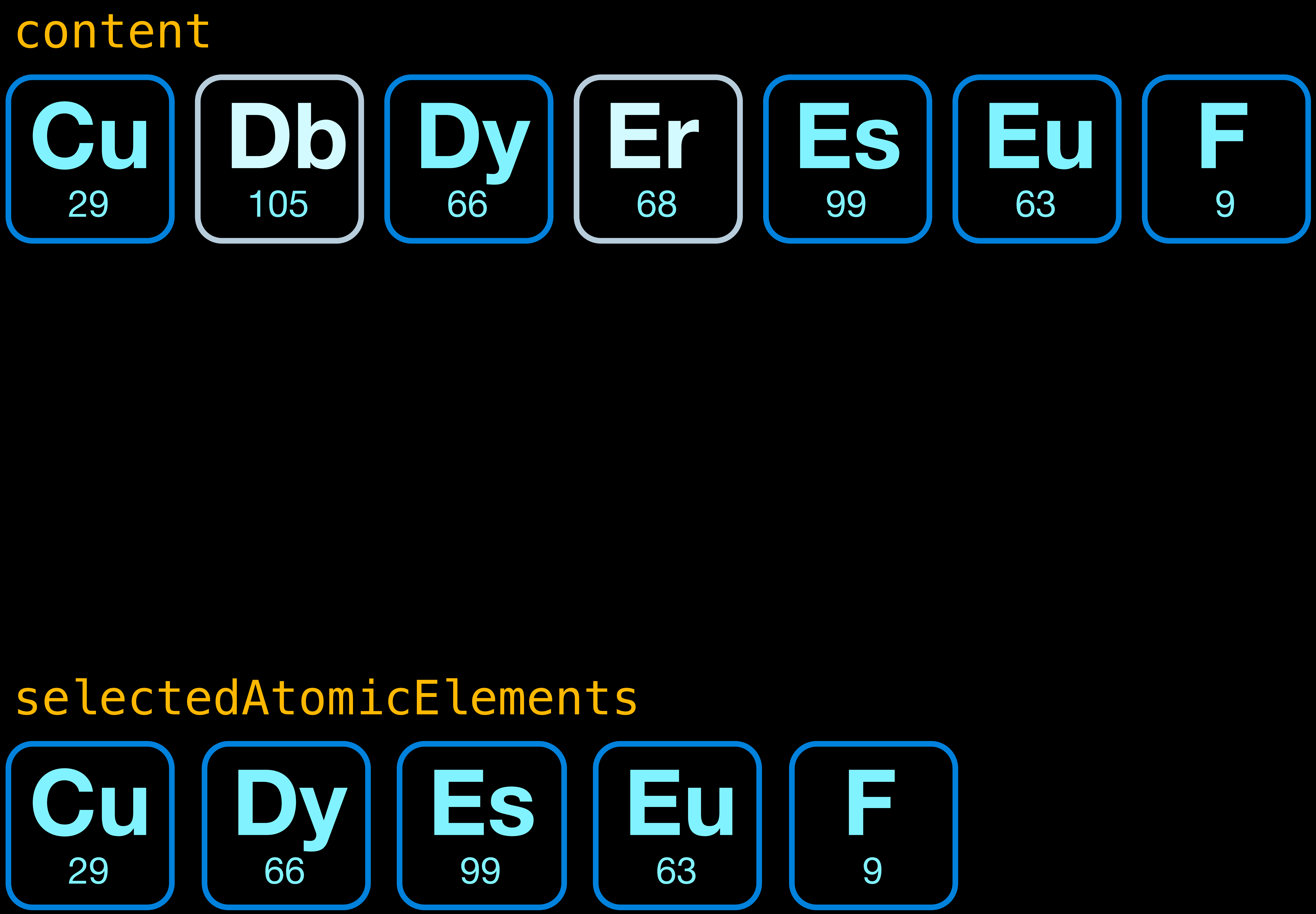
Array of selected atomic elements



9:41 AM 100%

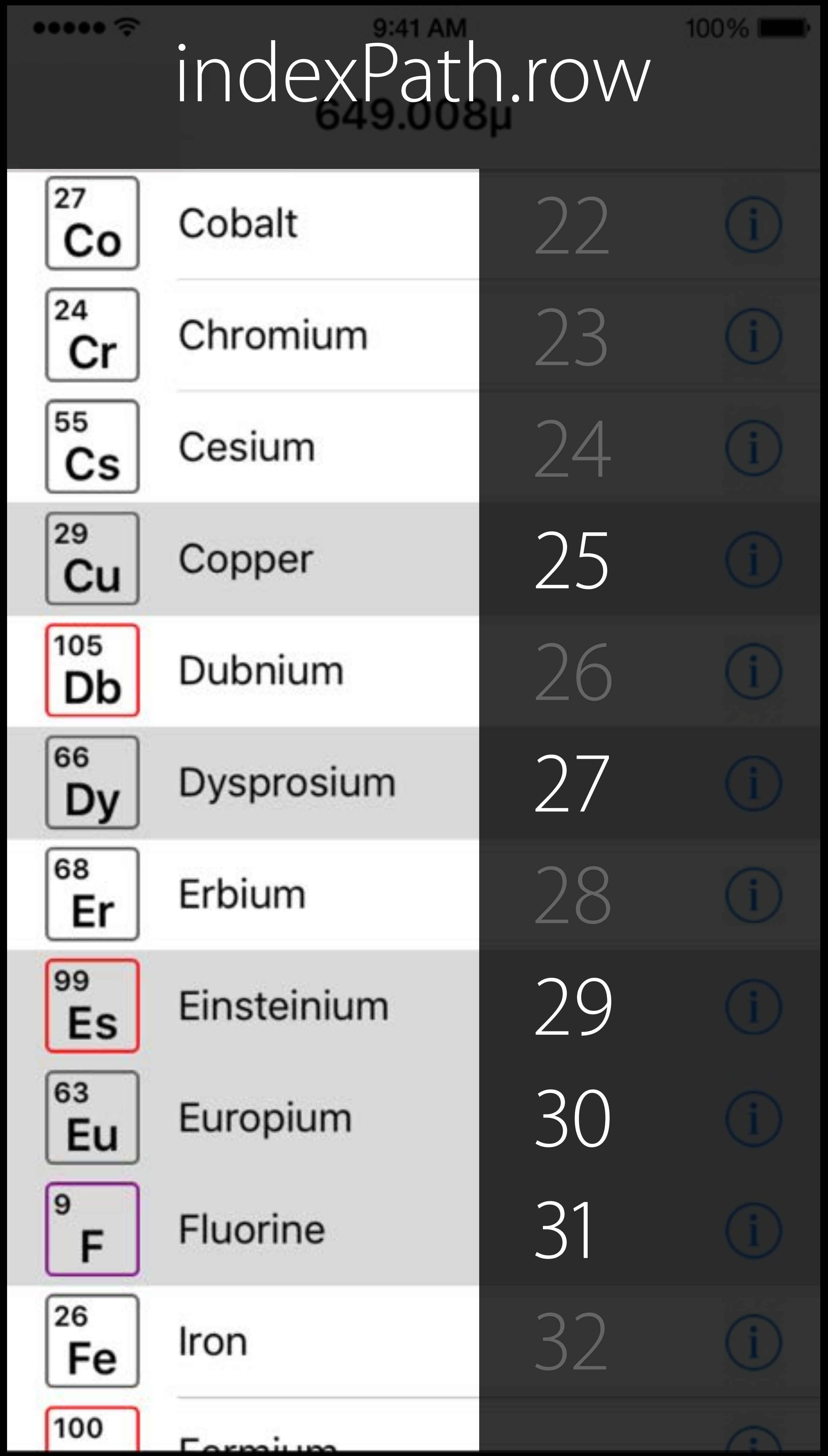
649.008μ

27 Co	Cobalt	ⓘ
24 Cr	Chromium	ⓘ
55 Cs	Cesium	ⓘ
29 Cu	Copper	ⓘ
105 Db	Dubnium	ⓘ
66 Dy	Dysprosium	ⓘ
68 Er	Erbium	ⓘ
99 Es	Einsteinium	ⓘ
63 Eu	Europium	ⓘ
9 F	Fluorine	ⓘ
26 Fe	Iron	ⓘ
100	Fermium	ⓘ



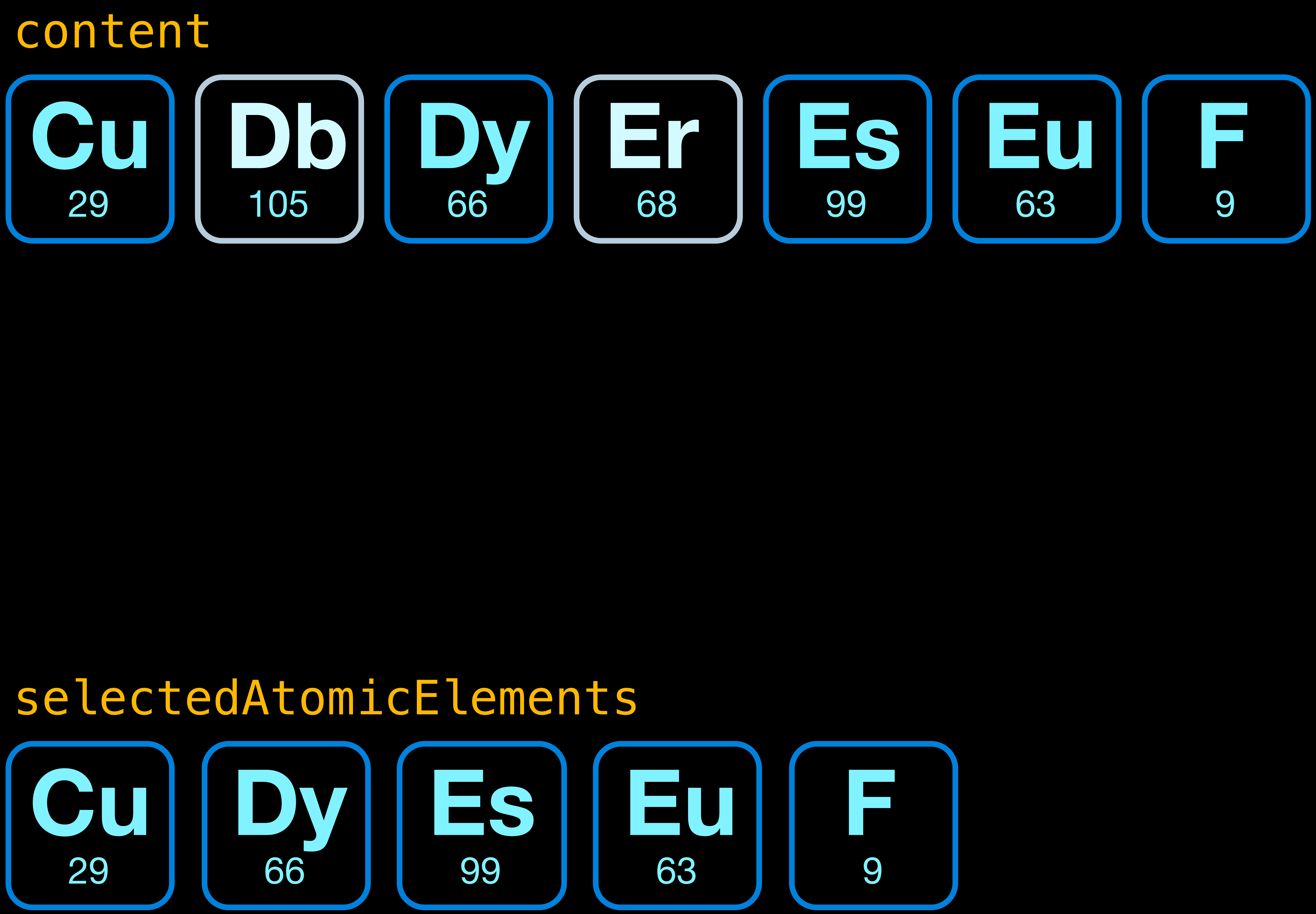
Intermediate Objective

Array of selected atomic elements



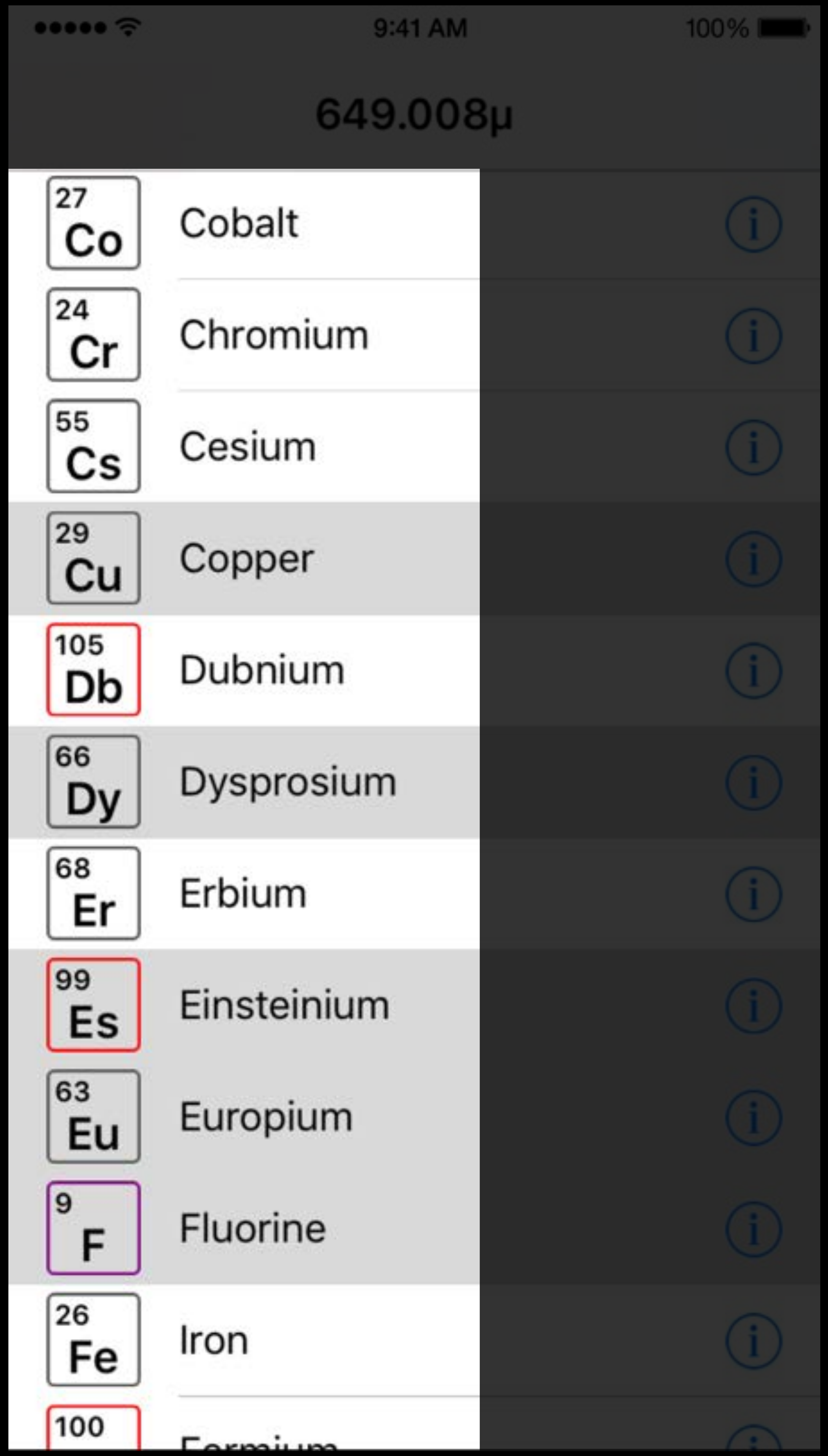
indexPath.row

27 Co	Cobalt	22
24 Cr	Chromium	23
55 Cs	Cesium	24
29 Cu	Copper	25
105 Db	Dubnium	26
66 Dy	Dysprosium	27
68 Er	Erbium	28
99 Es	Einsteinium	29
63 Eu	Europium	30
9 F	Fluorine	31
26 Fe	Iron	32



Intermediate Objective

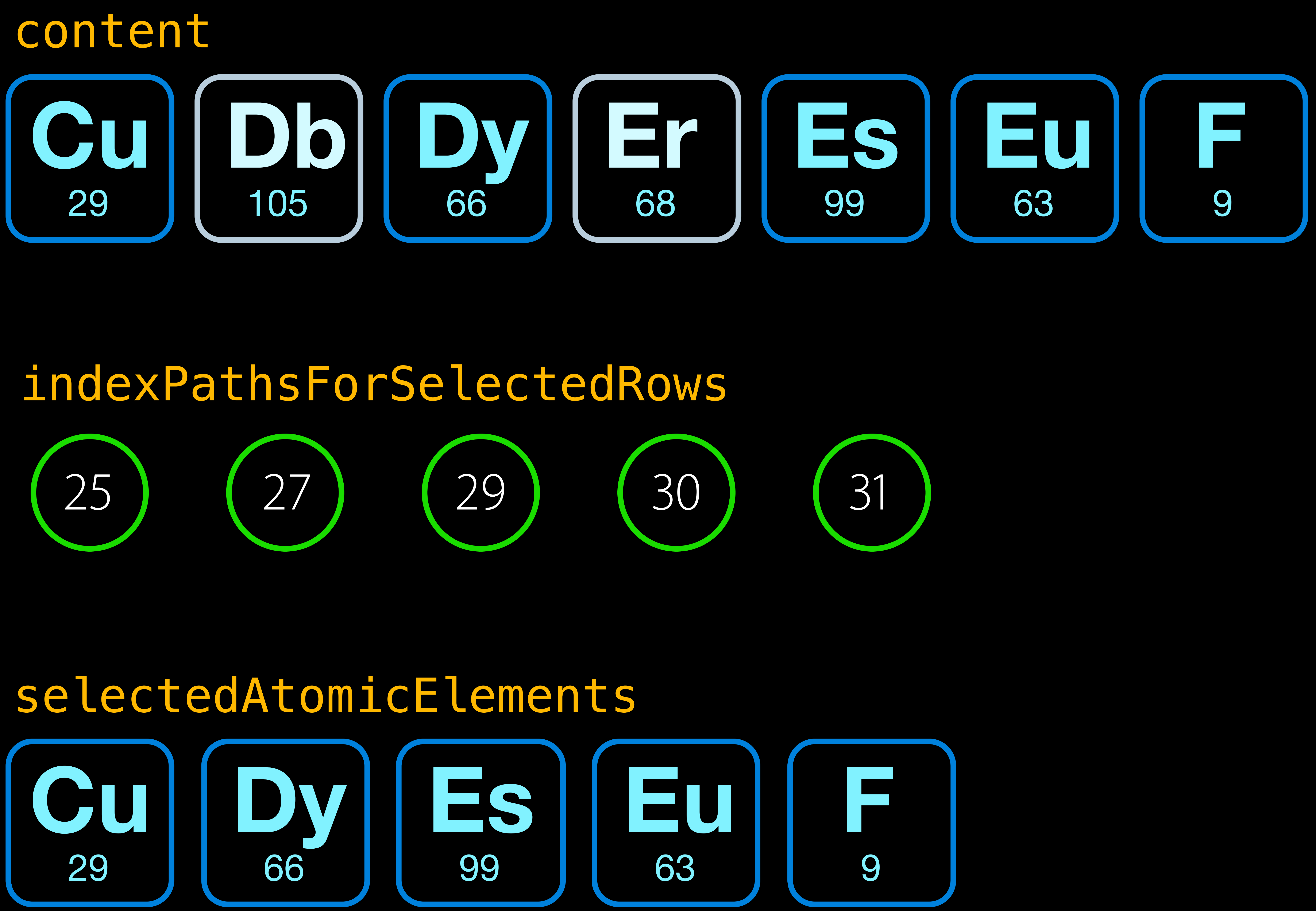
Array of selected atomic elements



9:41 AM 100%

649.008μ

27 Co	Cobalt
24 Cr	Chromium
55 Cs	Cesium
29 Cu	Copper
105 Db	Dubnium
66 Dy	Dysprosium
68 Er	Erbium
99 Es	Einsteinium
63 Eu	Europium
9 F	Fluorine
26 Fe	Iron
100	



Traditional Approach

For in

```
var selectedElements = [AtomicElement]()
if let indexPaths = tv?.indexPathsForSelectedRows {
    for ip in indexPaths {
        let currentElement = content[ip.row]
        selectedElements.append(currentElement)
    }
}
```

Swiftier Way

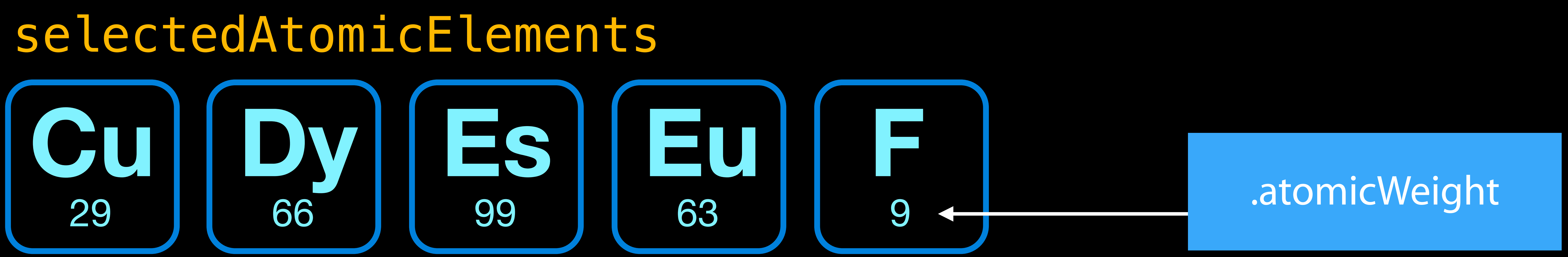
Map

```
var selectedElements = [AtomicElement]()
if let indexPaths = tv?.indexPathsForSelectedRows {
    for ip in indexPaths {
        let currentElement = content[ip.row]
        selectedElements.append(currentElement)
    }
}
```

```
let selectedElements = tv?.indexPathsForSelectedRows?.map{content[$0.row]}
```

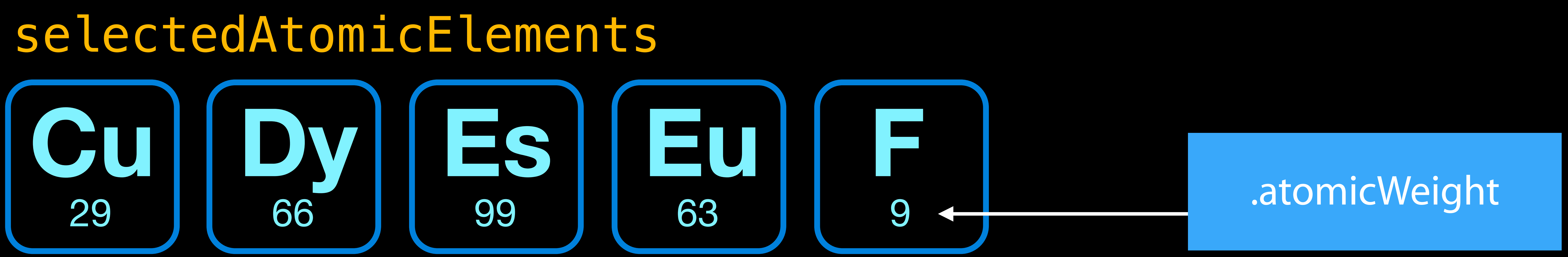
Adding Values of Items

Traditional way



Adding Values of Items

Traditional way



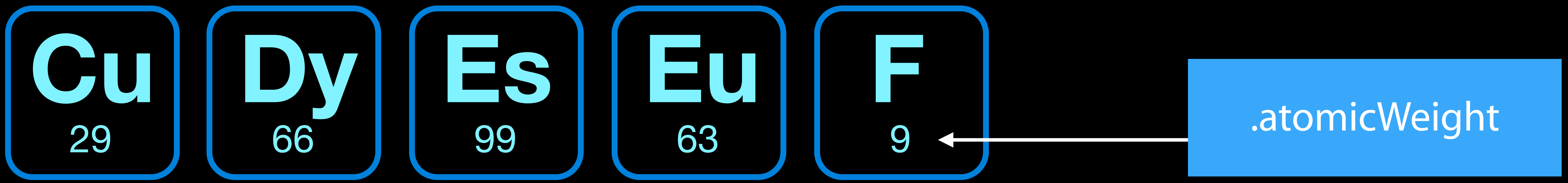
Using a for-in loop

```
var d = 0.0
for element in selectedAtomicElements {
    d = d + element.atomicWeight
}
```

Sum of Atomic Element Weights

Swift's reduce() method

selectedAtomicElements



Using a for-in loop

```
var d = 0.0
for element in selectedAtomicElements {
    d = d + element.atomicWeight
}
```

With Reduce()

```
reduce(0.0, combine: {$0 + $1.atomicWeight})
```

Final Code Snippet

All together, now

```
tableView?.indexPathsForSelectedRows?.map{self.content[$0.row]}.reduce(0.0, combine:
                              {$0 + $1.atomicWeight})
```

Demo

Map & Reduce

Obtaining closure through closures.

Summary

You have existing Objective-C code? Keep it.

You're adding new code? Consider writing it in Swift.

More Information

Stefan Lesser
Swift Evangelist
slesser@apple.com

Swift Language Documentation
http://developer.apple.com/swift

Apple Developer Forums
developer.apple.com/forums

Related Sessions

What's new in Swift	Presidio	Tuesday 11:10AM
Swift and Objective-C Interoperability	Mission	Tuesday 1:30PM
Protocol Oriented Programming in Swift	Marina	Wednesday 2:30PM
Optimizing Performance in Swift	Presidio	Thursday 9:00AM
Swift in Practice	Presidio	Thursday 2:30PM
Building Better UIKit Apps with Swift	Mission	Wednesday 2:00PM

WWDC15