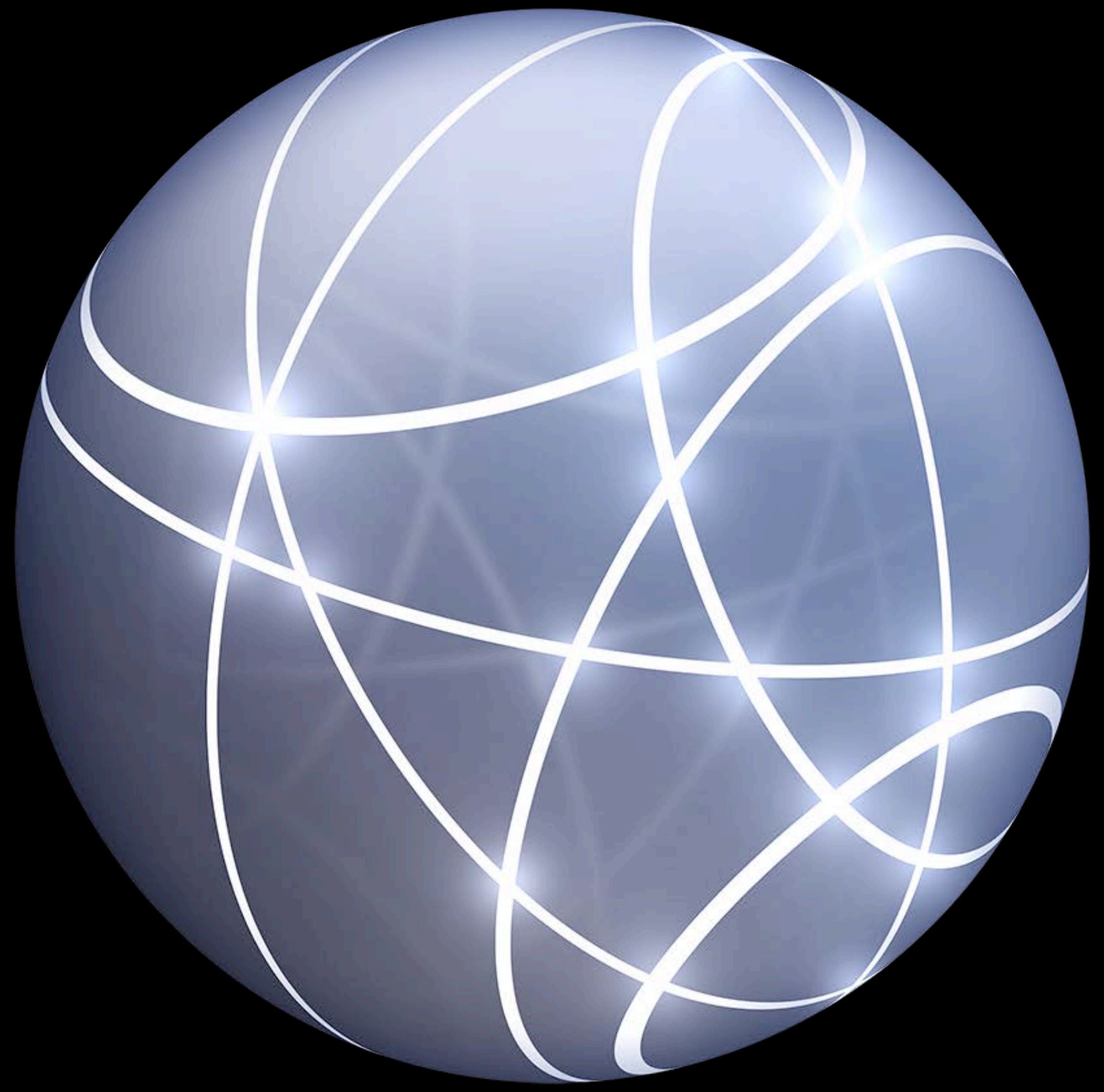
App Frameworks

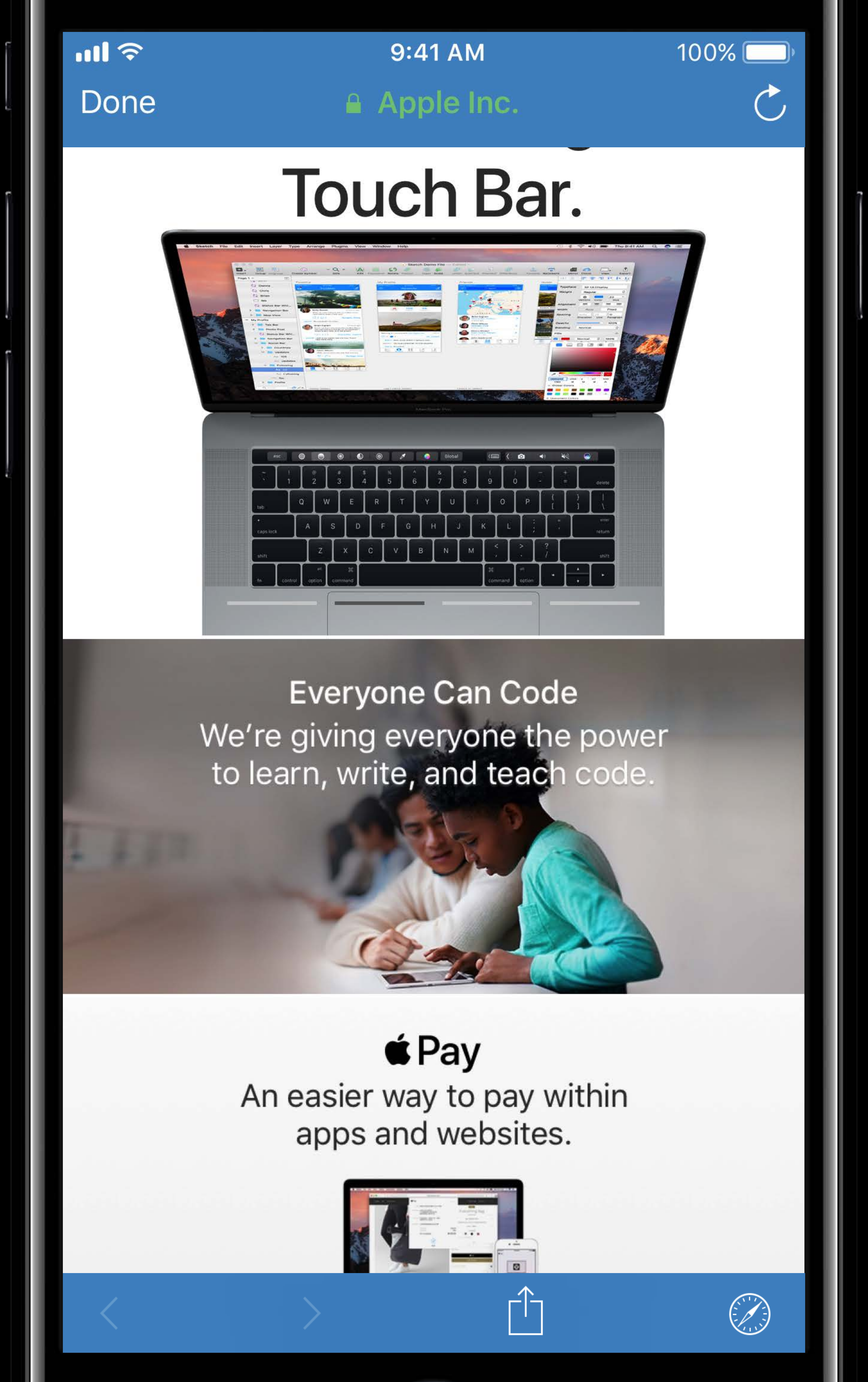
#WWDC17

Customized Loading in WKWebView

Session 220

Brady Eidson, Safari and WebKit Engineer
Alex Christensen, Safari and WebKit Engineer





Touch Bar.

Everyone Can Code
We're giving everyone the power to learn, write, and teach code.

Pay
An easier way to pay within apps and websites.

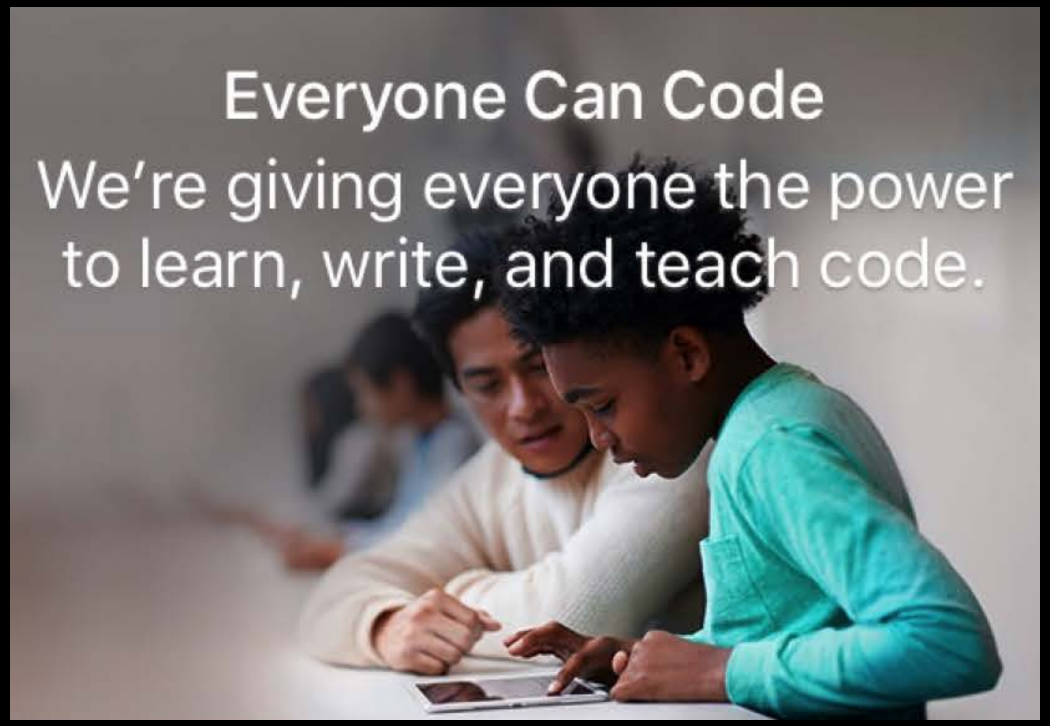
Introducing Touch Bar.

Everyone Can Code

We're giving everyone the power to learn, write, and teach code.

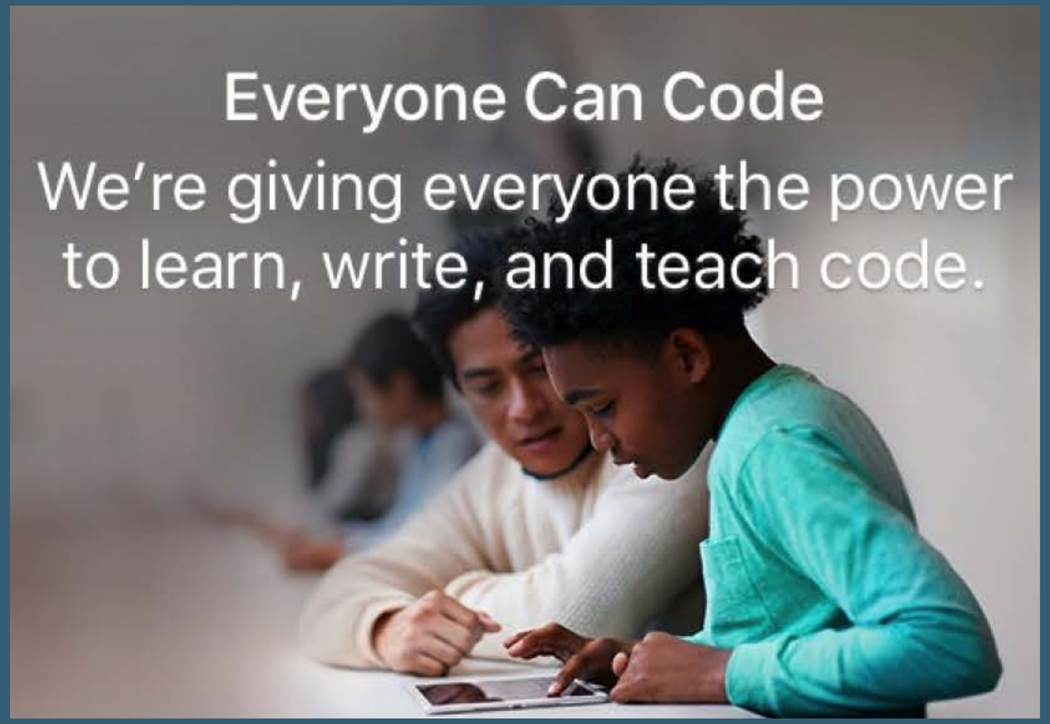
** Pay**

An easier way to pay within apps and websites.

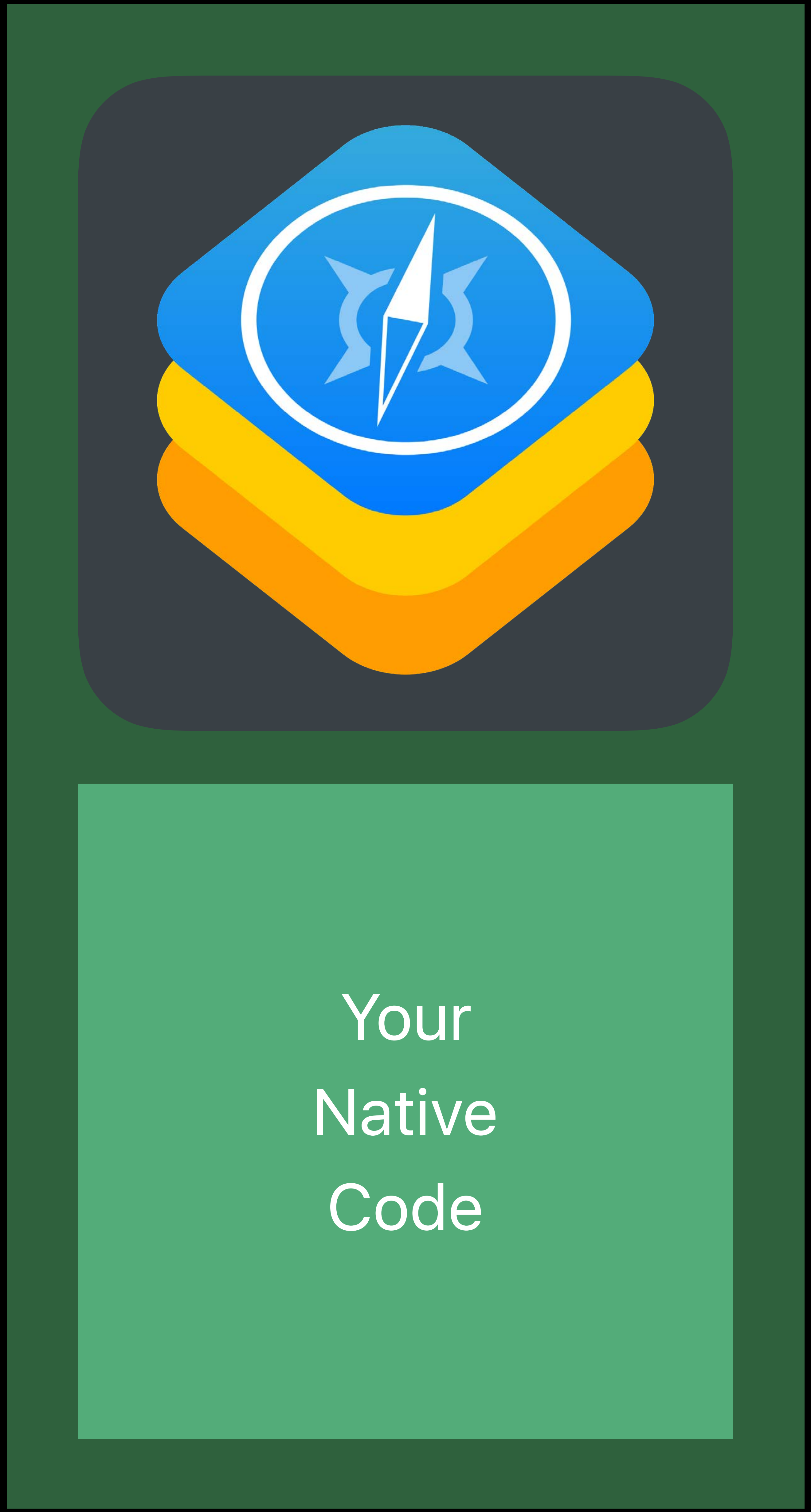


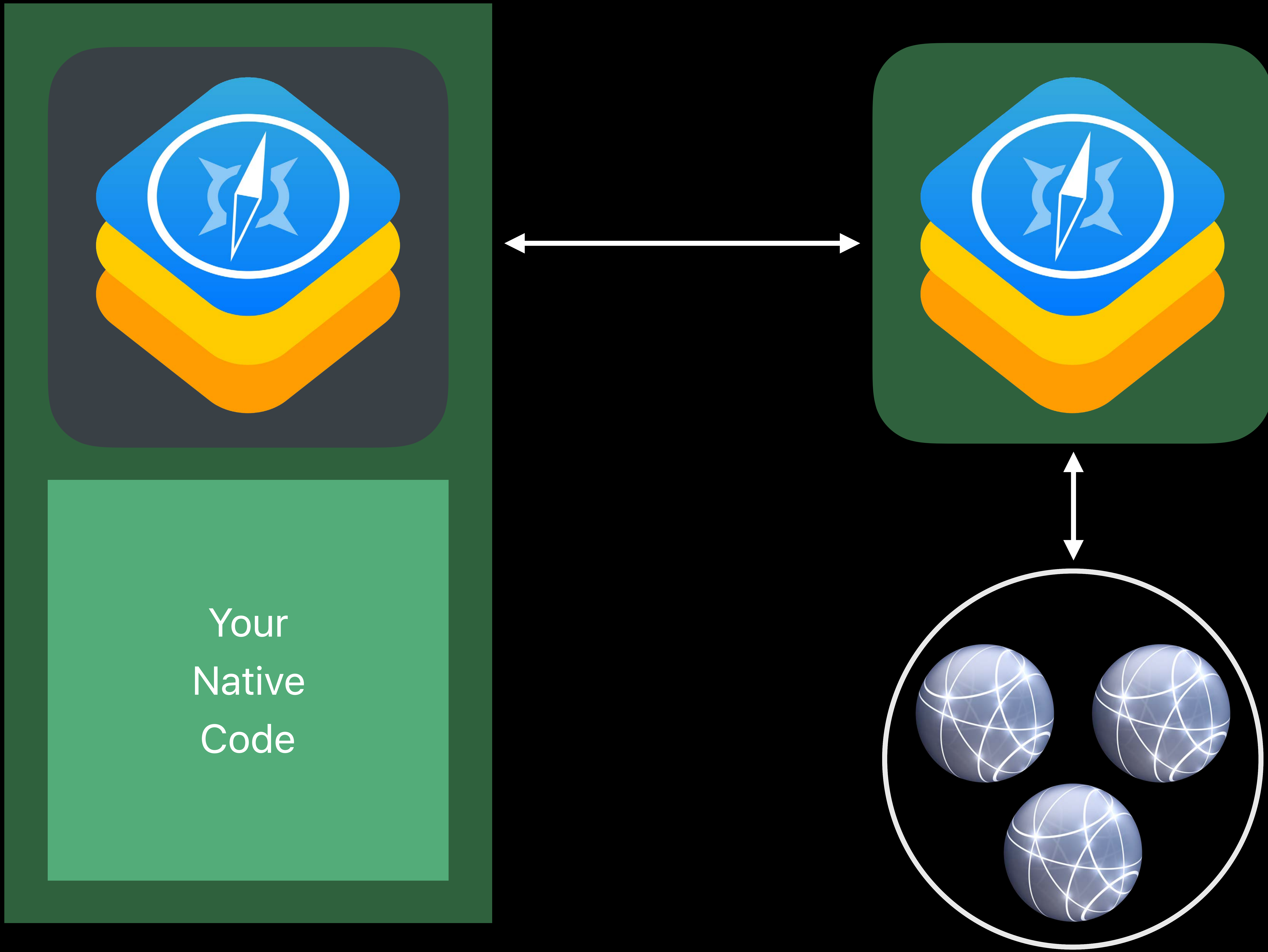
Everyone Can Code

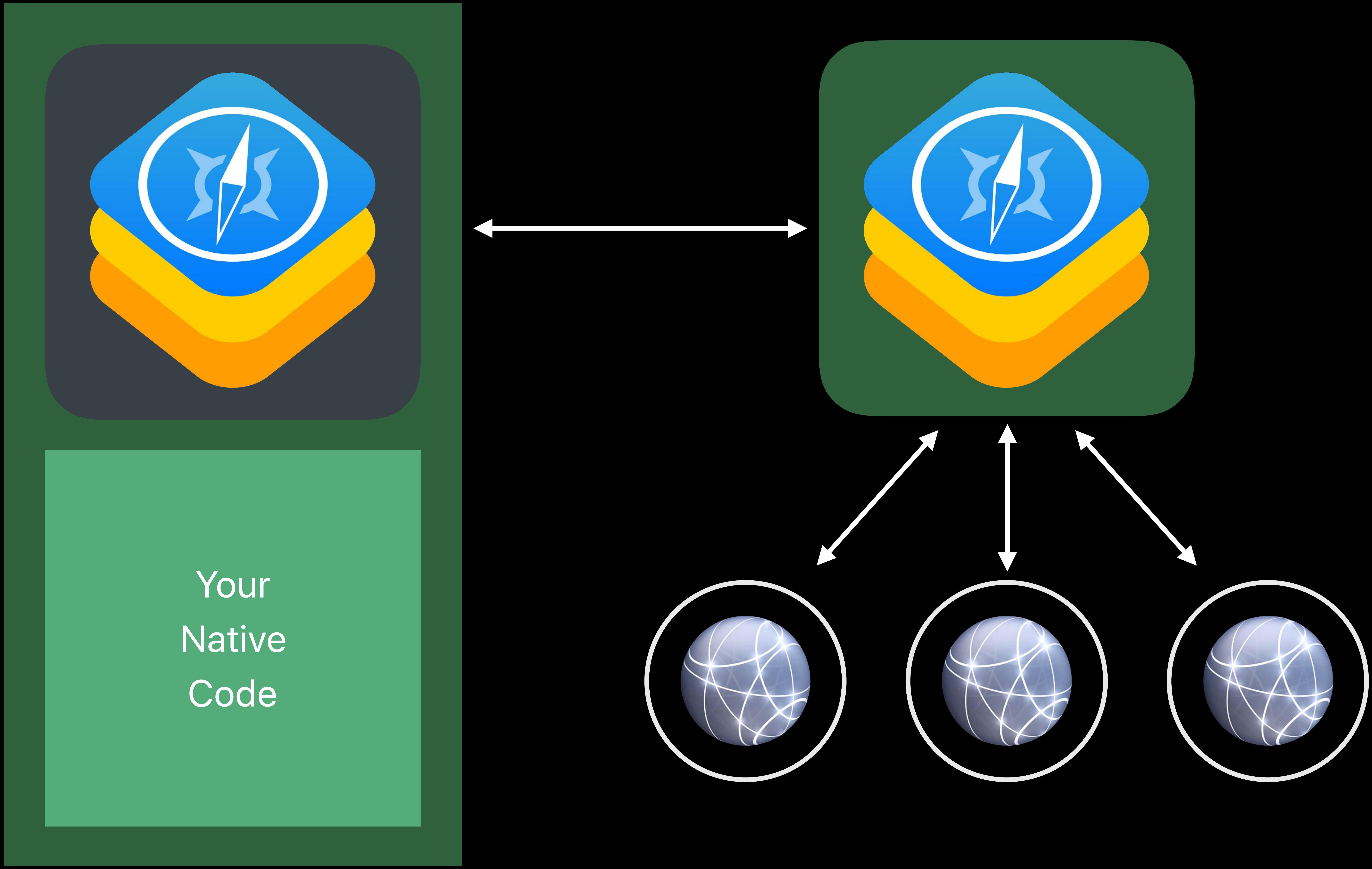
We’re giving everyone the power to learn, write, and teach code.


Everyone Can Code
We're giving everyone the power to learn, write, and teach code.

```
> return "Hello WWDC 2017"
"Hello WWDC 2017"
> return "Whoa, I know
Javascript”
"Whoa, I know Javascript”
>
```


Your
Native
Code


Your
Native
Code


Your
Native
Code

Developer's Requests
Manage cookies
Filter content
Custom resources

Manage cookies

Manage cookies

Filter unwanted content

Manage cookies

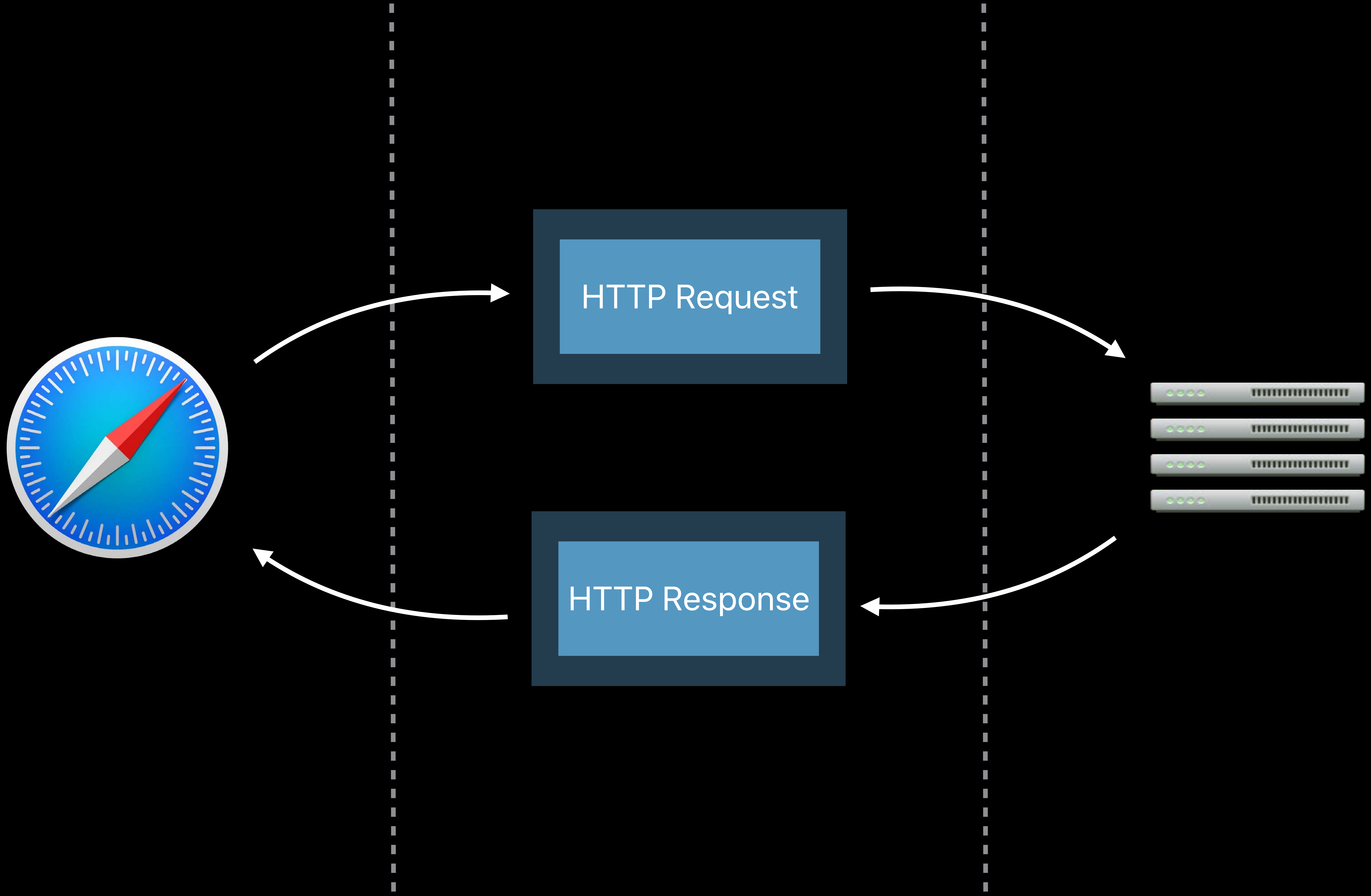
Filter unwanted content

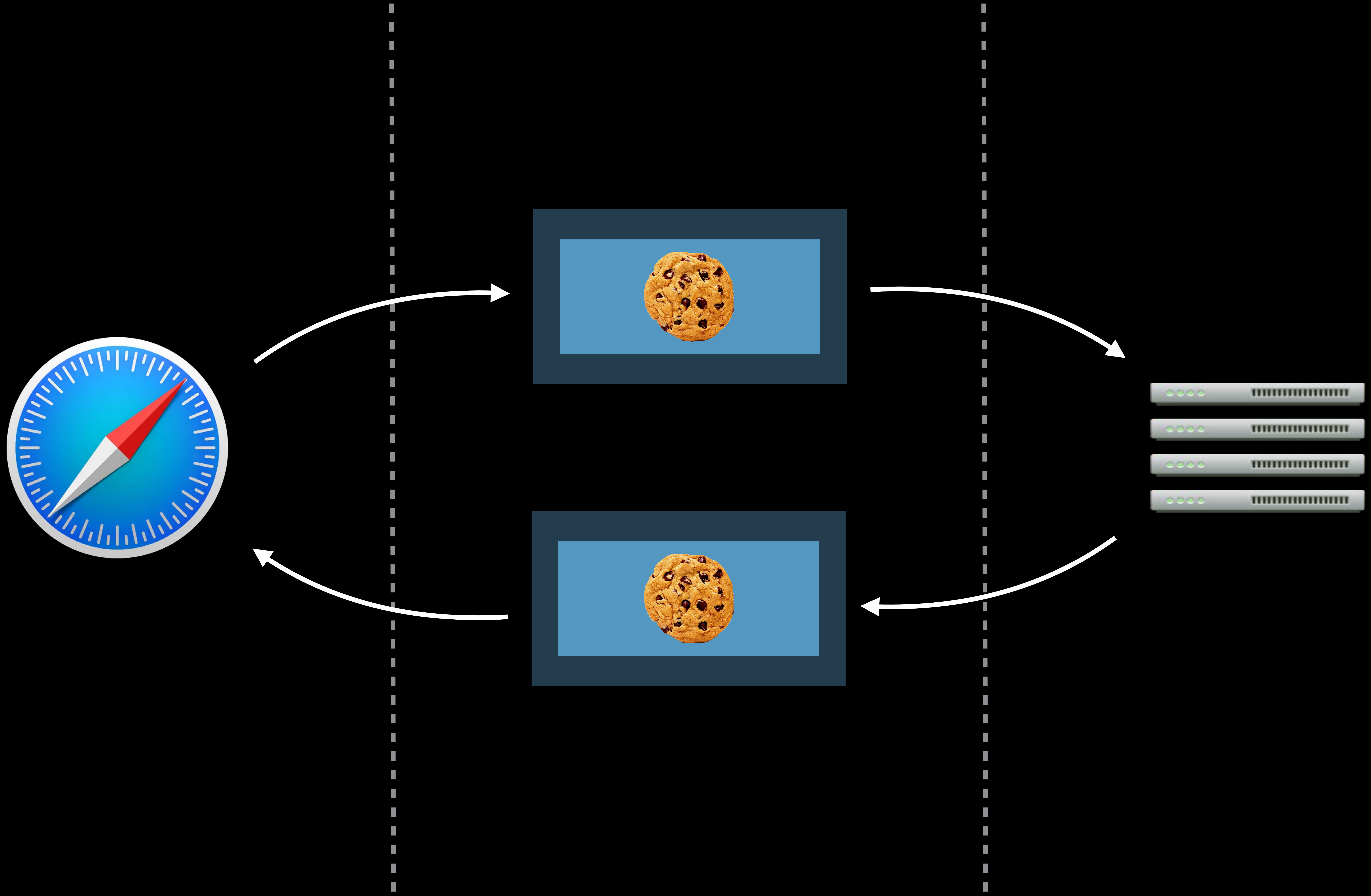
Provide custom resources

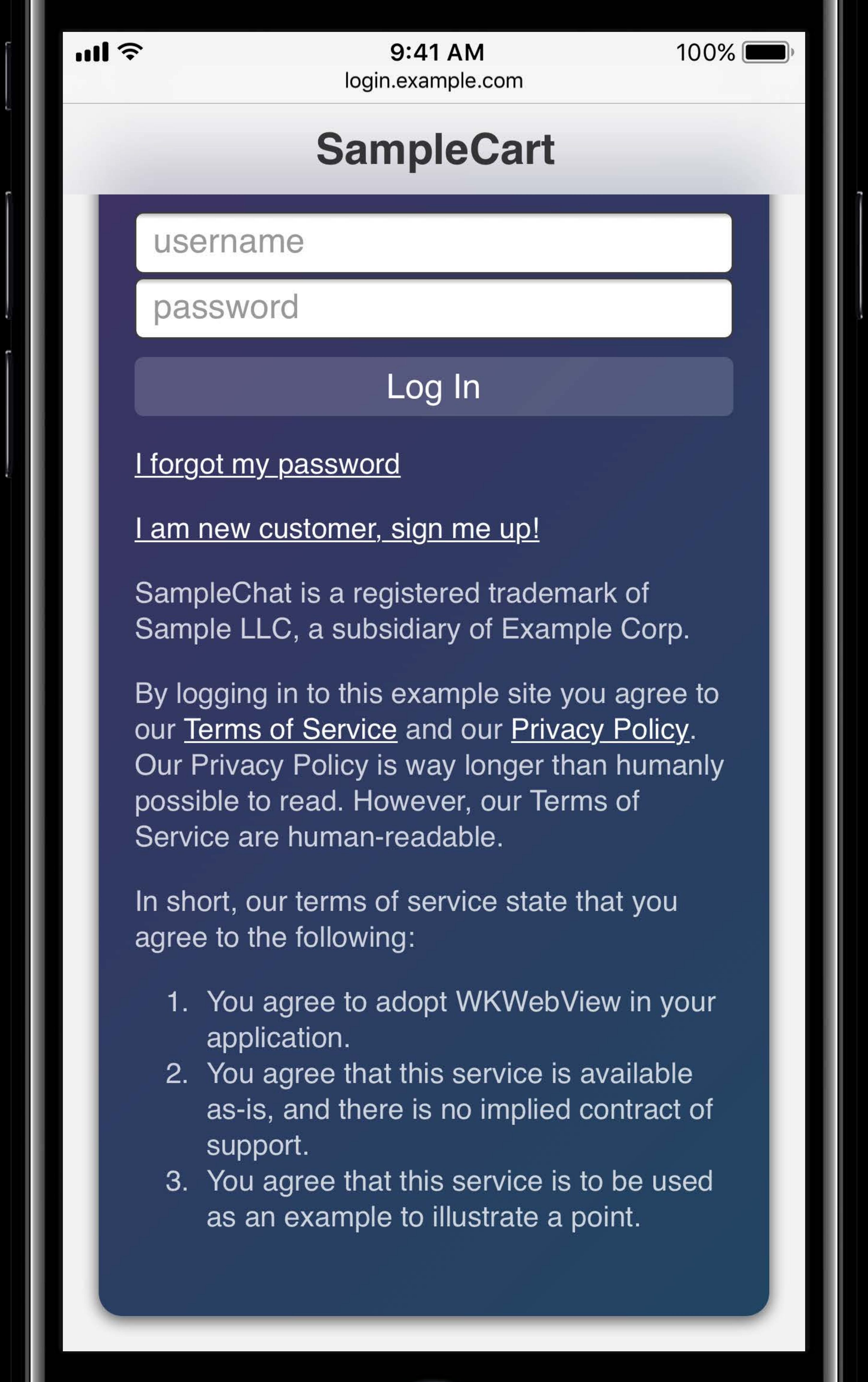
Manage cookies

Filter unwanted content

Provide custom resources


HTTP Request
HTTP Response





SampleCart

username

password

Log In

I forgot my password

I am new customer, sign me up!

SampleChat is a registered trademark of Sample LLC, a subsidiary of Example Corp.

By logging in to this example site you agree to our Terms of Service and our Privacy Policy. Our Privacy Policy is way longer than humanly possible to read. However, our Terms of Service are human-readable.

In short, our terms of service state that you agree to the following:

1. You agree to adopt WKWebView in your application.
2. You agree that this service is available as-is, and there is no implied contract of support.
3. You agree that this service is to be used as an example to illustrate a point.

Do not eat!!!

Manage Cookies

WKHTTPCookieStore

NEW

Manage Cookies
WKHTTPCookieStore

NEW

Add and remove individual cookies

Manage Cookies
WKHTTPCookieStore

NEW

Add and remove individual cookies

Access all cookies visible to a WKWebView

Manage Cookies

WKHTTPCookieStore

NEW

Add and remove individual cookies

Access all cookies visible to a WKWebView

Including HTTP-only cookies

Manage Cookies
WKHTTPCookieStore

NEW

Add and remove individual cookies

Access all cookies visible to a WKWebView

Including HTTP-only cookies

Observe the cookie store for changes

Manage Cookies
WKHTTPCookieStore

NEW

Get to a WKWebViews cookie store through its websiteDataStore

```
open class WKWebsiteDataStore : NSObject, NSCoding {
  open var httpCookieStore: WKHTTPCookieStore { get }
}

let cookieStore = webView.configuration.websiteDataStore.httpCookieStore;
```

Manage Cookies

WKHTTPCookieStore

NEW

Add a cookie

```
let cookie = HTTPCookie(properties: [
  HTTPCookiePropertyKey.domain: "canineschool.org",
  HTTPCookiePropertyKey.path: "/",
  HTTPCookiePropertyKey.secure: true,
  HTTPCookiePropertyKey.name: "LoginSessionID",
  HTTPCookiePropertyKey.value: "5bd9d8cabc46041579a311230539b8d1"])

cookieStore.setCookie(cookie!) {
  webView.load(loggedInURLRequest)
}
```

Manage Cookies

WKHTTPCookieStore

NEW

Add a cookie

```
let cookie = HTTPCookie(properties: [
  HTTPCookiePropertyKey.domain: "canineschool.org",
  HTTPCookiePropertyKey.path: "/",
  HTTPCookiePropertyKey.secure: true,
  HTTPCookiePropertyKey.name: "LoginSessionID",
  HTTPCookiePropertyKey.value: "5bd9d8cabc46041579a311230539b8d1"])

cookieStore.setCookie(cookie!) {
  webView.load(loggedInURLRequest)
}
```

Manage Cookies

WKHTTPCookieStore

NEW

Add a cookie

```
let cookie = HTTPCookie(properties: [
  HTTPCookiePropertyKey.domain: "canineschool.org",
  HTTPCookiePropertyKey.path: "/",
  HTTPCookiePropertyKey.secure: true,
  HTTPCookiePropertyKey.name: "LoginSessionID",
  HTTPCookiePropertyKey.value: "5bd9d8cabc46041579a311230539b8d1"])

cookieStore.setCookie(cookie!) {
  webView.load(loggedInURLRequest)
}
```

Manage Cookies

WKHTTPCookieStore

NEW

Retrieve the set of all cookies in a WKWebsiteDataStore

```
cookieStore.getAllCookies() { (cookies) in
  for cookie in cookies {
    // Find the login cookie
  }
}
```

Manage Cookies
WKHTTPCookieStore

NEW

Retrieve the set of all cookies in a WKWebsiteDataStore

```
cookieStore.getAllCookies() { (cookies) in
  for cookie in cookies {
    // Find the login cookie
  }
}
```

Manage Cookies

WKHTTPCookieStore

NEW

Delete a cookie

```
cookieStore.delete(cookie!) {
  webView.load(loggedOutURLRequest)
}
```

Manage Cookies
WKHTTPCookieStore

NEW

Delete a cookie

```
cookieStore.delete(cookie!) {
  webView.load(loggedOutURLRequest)
}
```

Manage cookies

Filter unwanted content

Provide custom resources

Manage cookies

Filter unwanted content

Provide custom resources

iPad

```
angular/apps/adp/partials/a
istory"><hr/></div><div class="c
/js/angular/apps/adp/partials/amp/_a
c/js/angular/apps/adp/partials/amp/_adm
atic/js/angular/apps/adp/partials/details/_
atic/js/angular/apps/adp/partials/details/_d
atic/js/angular/apps/adp/partials/details/_ex
atic/js/angular/apps/adp/partials/details/_fi
atic/js/angular/apps/adp/partials/details/_fi
atic/js/angular/apps/adp/partials/details/_me

"></ng-include><div id="asset-upload-date" cl
atic/js/angular/apps/adp/partials/details/_pa
atic/js/angular/apps/adp/partials/details/_
atic/js/angular/apps/adp/partials/index.htm
tic/js/angular/apps/adp/partials/metadat
ic/js/angular/apps/adp/partials/metadat
c/js/angular/apps/adp/partials/metadat
ive' || adp.asset.status === 'Pend
d' || adp.asset.status === 'Pub
angular/apps/adp/partials/
adp.asset.status
asset.statu
```

```
down" activation-method=
-download-history" target="_bl
ecta admin-button" ng-click="togg
al><section class="amp-admin-panel" ng
ributor-name with-avatar"><div class="adp
DBACK_DROPBOX_SYNCED"></p></div></script>
class="adp-license-header"><div class="adp
assetType==='vector'"><span class="title">
in availableSizes"><div class="tile-select
="file-id metadata-row"><span data-i18n="ID_

i18n="UPLOAD_DATE"/>{{adp.asset.approvalDat
data-i18n="HOW_TO_PAY"></strong></div><hr/>
warning-no-sub-no-credits" ng-switch-when=
ection class="adp-adult-content-overlay"
id="audio-tempo" class="tempo" ng-show=
s-photo metadata-row"><span id="dimens
tatus === 'Active' ||
ensions-photo metadata-row">
```

Filter Unwanted Content

WKContentRuleList

NEW

Filter Unwanted Content

WKContentRuleList

NEW

Same syntax as Content Blocker extensions for Safari*

*https://developer.apple.com/library/content/documentation/Extensions/Conceptual/ContentBlockingRules/Introduction/Introduction.html

Filter Unwanted Content

WKContentRuleList

NEW

Same syntax as Content Blocker extensions for Safari*

- Block loads

*https://developer.apple.com/library/content/documentation/Extensions/Conceptual/ContentBlockingRules/Introduction/Introduction.html

Filter Unwanted Content

WKContentRuleList

NEW

Same syntax as Content Blocker extensions for Safari*

- Block loads
- Make content invisible

*https://developer.apple.com/library/content/documentation/Extensions/Conceptual/ContentBlockingRules/Introduction/Introduction.html

Filter Unwanted Content

WKContentRuleList

NEW

Same syntax as Content Blocker extensions for Safari*

- Block loads
- Make content invisible
- Make insecure loads secure

*https://developer.apple.com/library/content/documentation/Extensions/Conceptual/ContentBlockingRules/Introduction/Introduction.html

Filter Unwanted Content

WKContentRuleList

NEW

Same syntax as Content Blocker extensions for Safari*

- Block loads
- Make content invisible
- Make insecure loads secure

WebKit compiles your rules into efficient bytecode

*https://developer.apple.com/library/content/documentation/Extensions/Conceptual/ContentBlockingRules/Introduction/Introduction.html

Filter Unwanted Content

WKContentRuleList

NEW

You supply rules in JSON

```
[{
    "trigger": {
        "url-filter": ".*"
    },
    "action": {
        "type": "make-https"
    }
}]
```

Filter Unwanted Content

WKContentRuleList

NEW

Compile your JSON using WKContentRuleListStore

```
let jsonString = loadJSONFromBundle()

WKContentRuleListStore.default().compileContentRuleList(
  forIdentifier: "ContentBlockingRules",
  encodedContentRuleList: jsonString) { (contentRuleList, error) in
    if let error = error {
      return
    }

    createWebViewWithContentRuleList(ruleList!)
}
```

Filter Unwanted Content

WKContentRuleList

NEW

Compile your JSON using WKContentRuleListStore

```
let jsonString = loadJSONFromBundle()

WKContentRuleListStore.default().compileContentRuleList(
  forIdentifier: "ContentBlockingRules",
  encodedContentRuleList: jsonString) { (contentRuleList, error) in
    if let error = error {
      return
    }

    createWebViewWithContentRuleList(ruleList!)
}
```

Filter Unwanted Content

WKContentRuleList

NEW

Compile your JSON using WKContentRuleListStore

```swift
let jsonString = loadJSONFromBundle()

WKContentRuleListStore.default().compileContentRuleList(
  forIdentifier: "ContentBlockingRules",
  encodedContentRuleList: jsonString) { (contentRuleList, error) in
    if let error = error {
      return
    }

    createWebViewWithContentRuleList(ruleList!)
}
```

Filter Unwanted Content

WKContentRuleList

NEW

Access previously compiled WKContentRuleList

```
WKContentRuleListStore.default().lookUpContentRuleList(forIdentifier: "ContentBlockingRules")
{ (contentRuleList, error) in
  // Use previously compiled content rule list
}
```

Filter Unwanted Content

WKContentRuleList

NEW

Add compiled WKContentRuleList to your WKWebView’s configuration

```
let configuration = WKWebViewConfiguration()
configuration.userContentController.add(contentRuleList)
```

Demo

Managing cookies and filtering content

Alex Christensen

Managing Cookies and Filtering Content Demo

Managing Cookies and Filtering Content Demo

Set a cookie before making a request

Managing Cookies and Filtering Content Demo

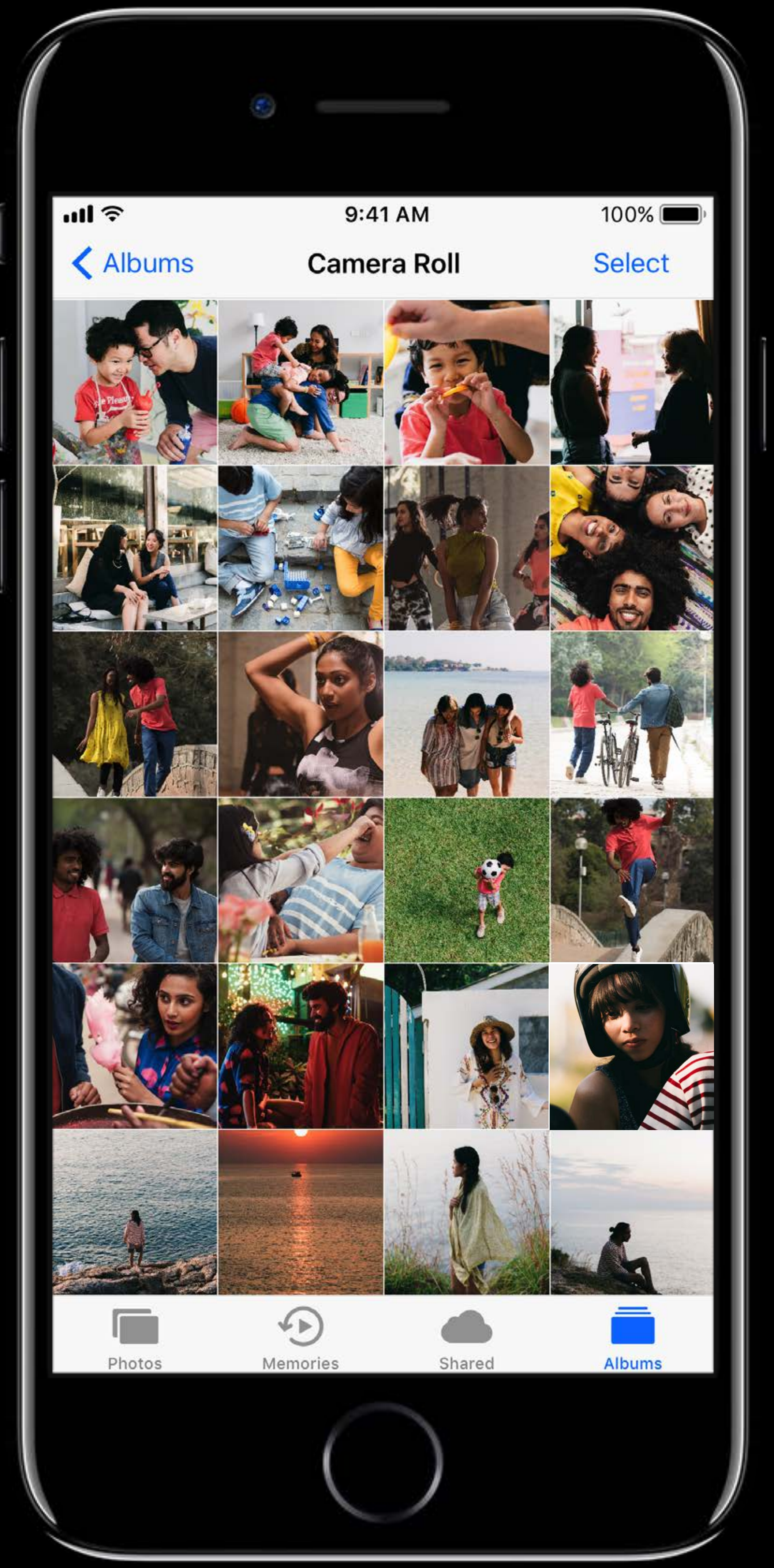
Set a cookie before making a request

Filter unwanted insecure resources

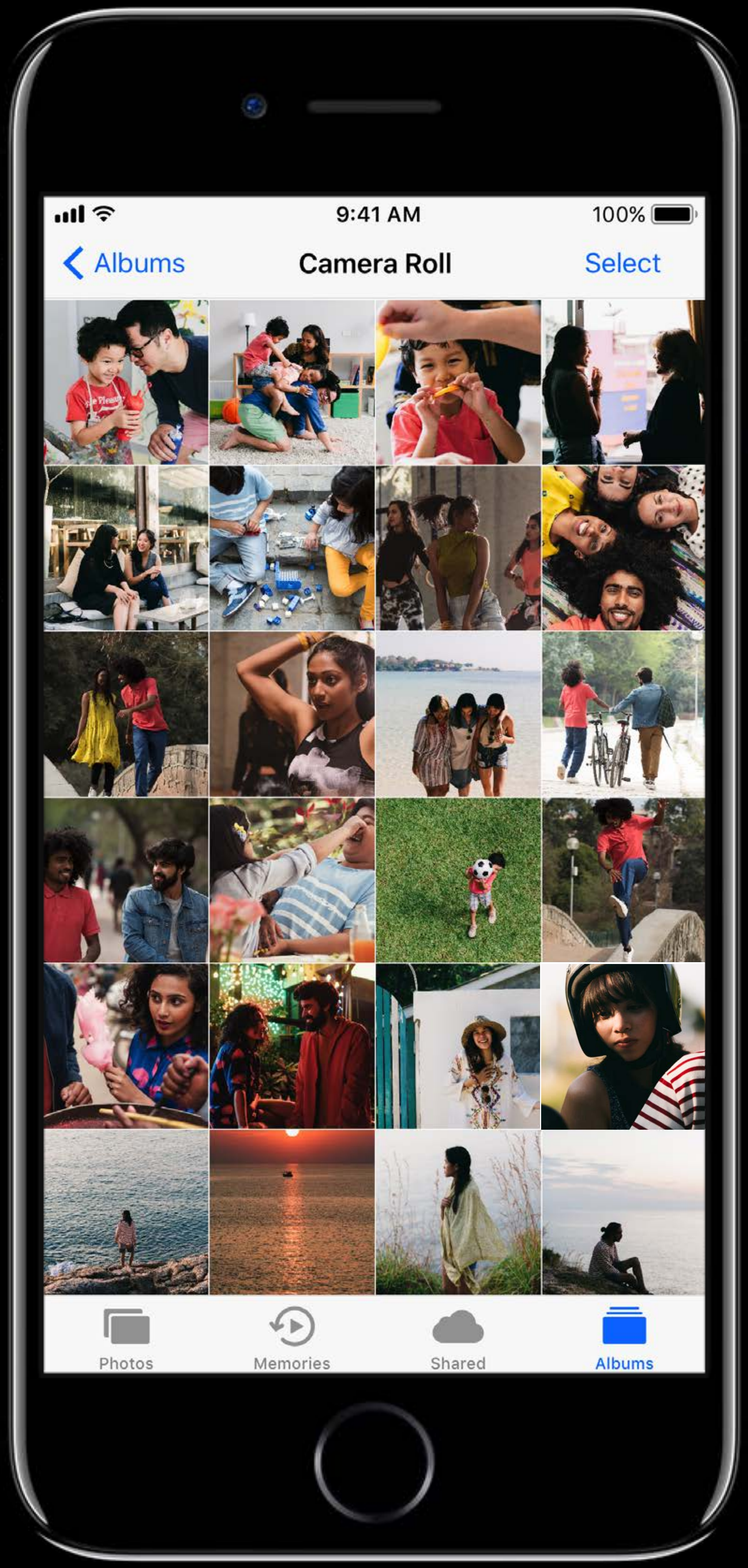
Manage cookies

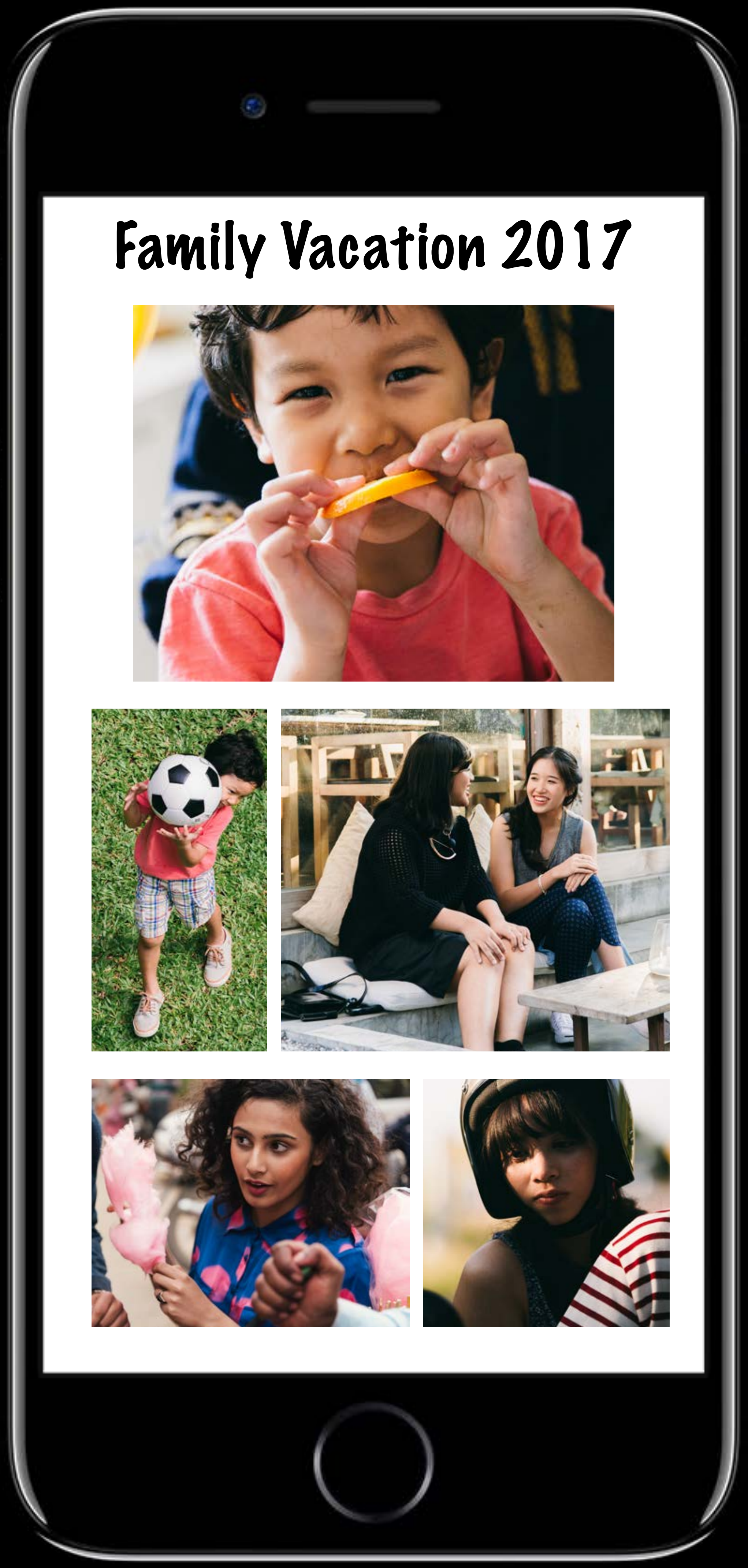
Filter unwanted content

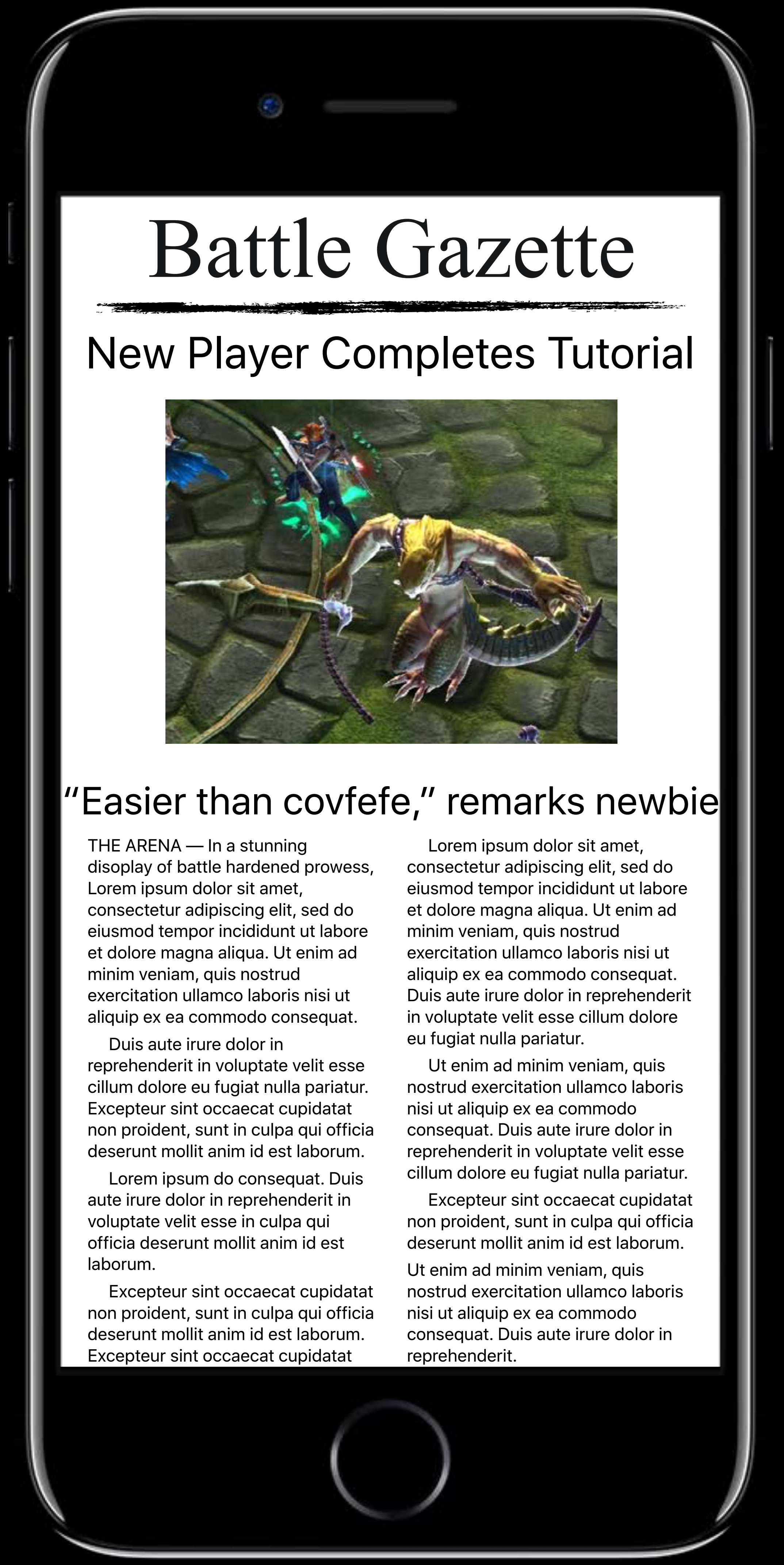
Provide custom resources


9:41 AM
100%
Albums
Camera Roll
Select
Photos
Memories
Shared
Albums

Family Vacation 2017


9:41 AM
100%
Albums
Camera Roll
Select
Photos
Memories
Shared
Albums


Family Vacation 2017



Battle Gazette

New Player Completes Tutorial

“Easier than covfefe,” remarks newbie

THE ARENA — In a stunning disoplay of battle hardened prowess, Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum do consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse in culpa qui officia deserunt mollit anim id est laborum.

Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Excepteur sint occaecat cupidatat

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit.

Provide Custom Resources

WKURLSchemeHandler

NEW

Provide Custom Resources

WKURLSchemeHandler

NEW

Allows your app to handle resource loads for a URL scheme https://www.apple.com

file:///Users/Alex/demo.txt

mailto:j.appleseed@icloud.com

https

file

mailto

Provide Custom Resources

WKURLSchemeHandler

NEW

Allows your app to handle resource loads for a URL scheme

Only custom URL schemes that WebKit doesn’t handle itself

```
let customScheme = "local"
```

Provide Custom Resources

WKURLSchemeHandler

NEW

Allows your app to handle resource loads for a URL scheme

Only custom URL schemes that WebKit doesn’t handle itself

You should future-proof your custom scheme

```
let customScheme = "apple-local"
```

Provide Custom Resources

WKURLSchemeHandler

NEW

Simple protocol you implement

```swift
class MyCustomSchemeHandler : NSObject, WKURLSchemeHandler {
    func webView(_ webView: WKWebView, start urlSchemeTask: WKURLSchemeTask) {
    }

    func webView(_ webView: WKWebView, stop urlSchemeTask: WKURLSchemeTask) {
    }
}
```

Provide Custom Resources

WKURLSchemeHandler

NEW

Set scheme handlers on the WKWebViewConfiguration

```
let configuration = WKWebViewConfiguration()
configuration.setURLSchemeHandler(MyCustomSchemeHandler(), forURLScheme: “apple-local”)
```

Provide Custom Resources

WKURLSchemeHandler

NEW

Load something in your view that uses your scheme

```
let configuration = WKWebViewConfiguration()
configuration.setURLSchemeHandler(MyCustomSchemeHandler(), forURLScheme: "apple-local")

let webView = WKWebView(frame: getFrame(), configuration: configuration)
webView.load(URLRequest(url: URL(string: "http://example.com/demoContent")!))
```

Provide Custom Resources

WKURLSchemeHandler

NEW

Load something in your view that uses your scheme

```
let configuration = WKWebViewConfiguration()
configuration.setURLSchemeHandler(MyCustomSchemeHandler(), forURLScheme: “apple-local”)

let webView = WKWebView(frame: getFrame(), configuration: configuration)
webView.load(URLRequest(url: URL(string: “apple-local:top”)!))
```

Provide Custom Resources

WKURLSchemeHandler

NEW

Load something in your view that uses your scheme

```
let configuration = WKWebViewConfiguration()
configuration.setURLSchemeHandler(MyCustomSchemeHandler(), forURLScheme: “apple-local”)

let webView = WKWebView(frame: getFrame(), configuration: configuration)
webView.load(URLRequest(url: URL(string: “apple-local:top”)!))
```

Provide Custom Resources

WKURLSchemeHandler

NEW

The WKURLSchemeTask sent to your handler represents a resource load

```
protocol WKURLSchemeTask : NSObjectProtocol {
  var request: URLRequest

  func didReceive(_ response: URLResponse)
  func didReceive(_ data: Data)
  func didFinish()
  func didFailWithError(_ error: Error)
}
```

Provide Custom Resources

WKURLSchemeHandler

NEW

The WKURLSchemeTask sent to your handler represents a resource load

```
protocol WKURLSchemeTask : NSObjectProtocol {
  var request: URLRequest

  func didReceive(_ response: URLResponse)
  func didReceive(_ data: Data)
  func didFinish()
  func didFailWithError(_ error: Error)
}
```

Provide Custom Resources

WKURLSchemeHandler

NEW

The WKURLSchemeTask sent to your handler represents a resource load

```
protocol WKURLSchemeTask : NSObjectProtocol {
  var request: URLRequest

  func didReceive(_ response: URLResponse)
  func didReceive(_ data: Data)
  func didFinish()
  func didFailWithError(_ error: Error)
}
```

Provide Custom Resources

WKURLSchemeHandler

NEW

You first prepare a response

```
func webView(_ webView: WKWebView, start urlSchemeTask: WKURLSchemeTask) {
    let resourceData = createHTMLResourceData()

    let response = URLResponse(
        url: urlSchemeTask.request.url!,
        mimeType: "text/html",
        expectedContentLength: resourceData.count,
        textEncodingName: nil)
}
```

Provide Custom Resources

WKURLSchemeHandler

NEW

The response must include the MIME type

```
func webView(_ webView: WKWebView, start urlSchemeTask: WKURLSchemeTask) {
    let resourceData = createHTMLResourceData()

    let response = URLResponse(
        url: urlSchemeTask.request.url!,
        mimeType: “text/html”,
        expectedContentLength: resourceData.count,
        textEncodingName: nil)
}
```

Provide Custom Resources

WKURLSchemeHandler

NEW

Deliver the response

```
func webView(_ webView: WKWebView, start urlSchemeTask: WKURLSchemeTask) {
    let resourceData = createHTMLResourceData()

    let response = ...

    urlSchemeTask.didReceive(response)
    urlSchemeTask.didReceive(resourceData)
    urlSchemeTask.didFinish()
}
```

Provide Custom Resources

WKURLSchemeHandler

NEW

Deliver the resource data

```
func webView(_ webView: WKWebView, start urlSchemeTask: WKURLSchemeTask) {
    let resourceData = createHTMLResourceData()

    let response = ...

    urlSchemeTask.didReceive(response)
    urlSchemeTask.didReceive(resourceData)
    urlSchemeTask.didFinish()
}
```

Provide Custom Resources
WKURLSchemeHandler

NEW

Signal completion

```
func webView(_ webView: WKWebView, start urlSchemeTask: WKURLSchemeTask) {
    let resourceData = createHTMLResourceData()

    let response = ...

    urlSchemeTask.didReceive(response)
    urlSchemeTask.didReceive(resourceData)
    urlSchemeTask.didFinish()
}
```

Demo

Providing custom resources

Alex Christensen

Providing Custom Resources Demo

Providing Custom Resources Demo

Chose a future-proof URL scheme

Providing Custom Resources Demo

Chose a future-proof URL scheme

Delivered image data asynchronously

WKHTTPCookieStore

WKContentRuleList

WKURLSchemeHandler

Developer's Requests

More Information

https://developer.apple.com/wwdc17/220

Related Sessions

What's New in Safari View Controller	Executive Ballroom	Thursday 10:00AM
Introducing Safari View Controller		WWDC 2015
Safari Extensibility: Content Blocking and Shared Links		WWDC 2015
Introducing the Modern WebKit API		WWDC 2014

Labs

Safari, WebKit, and Password AutoFill Lab	Technology Lab B	Wed 5:10PM–6:10PM
Safari, WebKit, and Password AutoFill Lab	Technology Lab K	Thur 11:00AM–1:00PM

WWDC17