

New issue

Jump to bottom

feat(ios): allow for for per-image-request-headers #691

Merged

DylanVann merged 8 commits into DylanVann:master from dreampiggy:bugfix_ios_sd_header_per_image_request on Jul 17, 2020

Conversation 38 Commits 8 Checks 0 Files changed 3



dreampiggy commented on Jun 5, 2020 • edited

Contributor

See detail issue description in #690. See SDWebImage issue: [SDWebImage/SDWebImage#3031](#)
Close #242

Fix the implementation on iOS for per-image-request-header-setup. Should not use `SDWebImageDownloader` which manage the global shared headers.

Should Use the context option of request modifier. Or when concurrent request comes, the headers may mass up.

This MR also bump webp coder because the old version contains OOM issue.

CC @DylanVann @andreialec



dreampiggy added 2 commits 2 years ago

Bump the dependency of SDWebImage into 5.8+ because of the SDWebImage...

8f4aadf

Fix the implementation on iOS for per-image-request-header-setup. Sho...

a45ae95

dreampiggy mentioned this pull request on Jun 5, 2020

Automatically managing HTTP headers (host, accept) SDWebImage/SDWebImage#3031

Closed

3 tasks

andreialec commented on Jun 5, 2020 • edited



Add the support for data:image URL for WebP images

f7b2b98

dreampiggy commented on Jun 5, 2020 • edited

Contributor Author

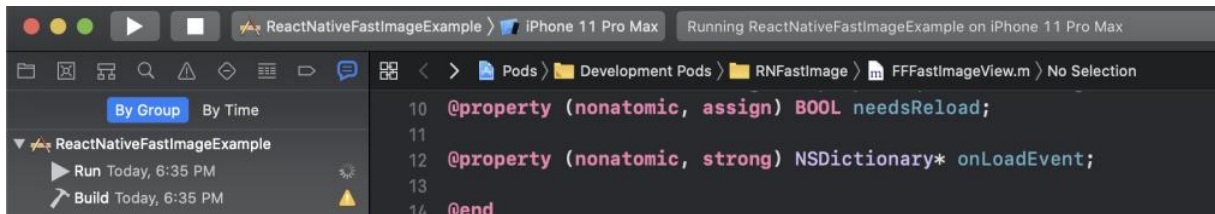
Extra changes:

1. Fix the issue of `data:image` for WebP images. Using SDWebImage's API can keep the external format support, including WebP/APNG/AVIF/HEIF. As well as iOS system default format.
2. Always bypass invalid SSL certificate error, same behavior like React-Native itself

dreampiggy commented on Jun 5, 2020

Contributor Author

The CI environment seems broken (Not related to this MR changes). I can ensure the build success on my local development environment.



```
✓ Creating entry file 2.6 secs
(babel plugin) Error: Cannot find module '@babel/compat-data/corejs3-shipped-proposals'
Require stack:
  - /home/circleci/project/node_modules/@babel/preset-env/lib/polyfills/corejs3/usage-plugin.js
  - /home/circleci/project/node_modules/@babel/preset-env/lib/index.js
  - /home/circleci/project/node_modules/@babel/core/lib/config/files/plugins.js
  - /home/circleci/project/node_modules/@babel/core/lib/config/files/index.js
  - /home/circleci/project/node_modules/@babel/core/lib/index.js
  - /home/circleci/project/node_modules/tsdx/dist/createRollupConfig.js
  - /home/circleci/project/node_modules/tsdx/dist/createBuildConfigs.js
  - /home/circleci/project/node_modules/tsdx/dist/index.js
Error: Cannot find module '@babel/compat-data/corejs3-shipped-proposals'
Require stack:
  - /home/circleci/project/node_modules/@babel/preset-env/lib/polyfills/corejs3/usage-plugin.js
  - /home/circleci/project/node_modules/@babel/preset-env/lib/index.js
  - /home/circleci/project/node_modules/@babel/core/lib/config/files/plugins.js
  - /home/circleci/project/node_modules/@babel/core/lib/config/files/index.js
  - /home/circleci/project/node_modules/@babel/core/lib/index.js
  - /home/circleci/project/node_modules/tsdx/dist/createRollupConfig.js
  - /home/circleci/project/node_modules/tsdx/dist/createBuildConfigs.js
  - /home/circleci/project/node_modules/tsdx/dist/index.js
    at Function.Module._resolveFilename (internal/modules/cjs/loader.js:1029:15)
    at Function.Module._load (internal/modules/cjs/loader.js:898:27)
```

dreampiggy commented on Jun 5, 2020

Contributor Author

@andreialecu If your RN iOS use CocoaPods. You can use the git commit dependency to test the behavior.

```
pod 'react-native-fast-image', :git => 'https://github.com/dreampiggy/react-native-fast-image.git', :branch => 'bugfix_ios_sd_header_per_image_request'
```

andreialecu commented on Jun 5, 2020

I generally like to use [patch-package](#). I think it should work just the same.

I seem to be getting this error on `pod install` though. I can probably update the deployment target, but it may result in breaking changes for users?

```
[!] CocoaPods could not find compatible versions for pod "SDWebImage":
  In snapshot (Podfile.lock):
    SDWebImage (= 5.7.0, ~> 5.0)

  In Podfile:
    RNFastImage (from `../node_modules/react-native-fast-image`) was resolved to 8.1.5, which depends on
      SDWebImage (~> 5.8)

Specs satisfying the `SDWebImage (= 5.7.0, ~> 5.0), SDWebImage (~> 5.8)` dependency were found, but they required a higher minimum deployment target.
```

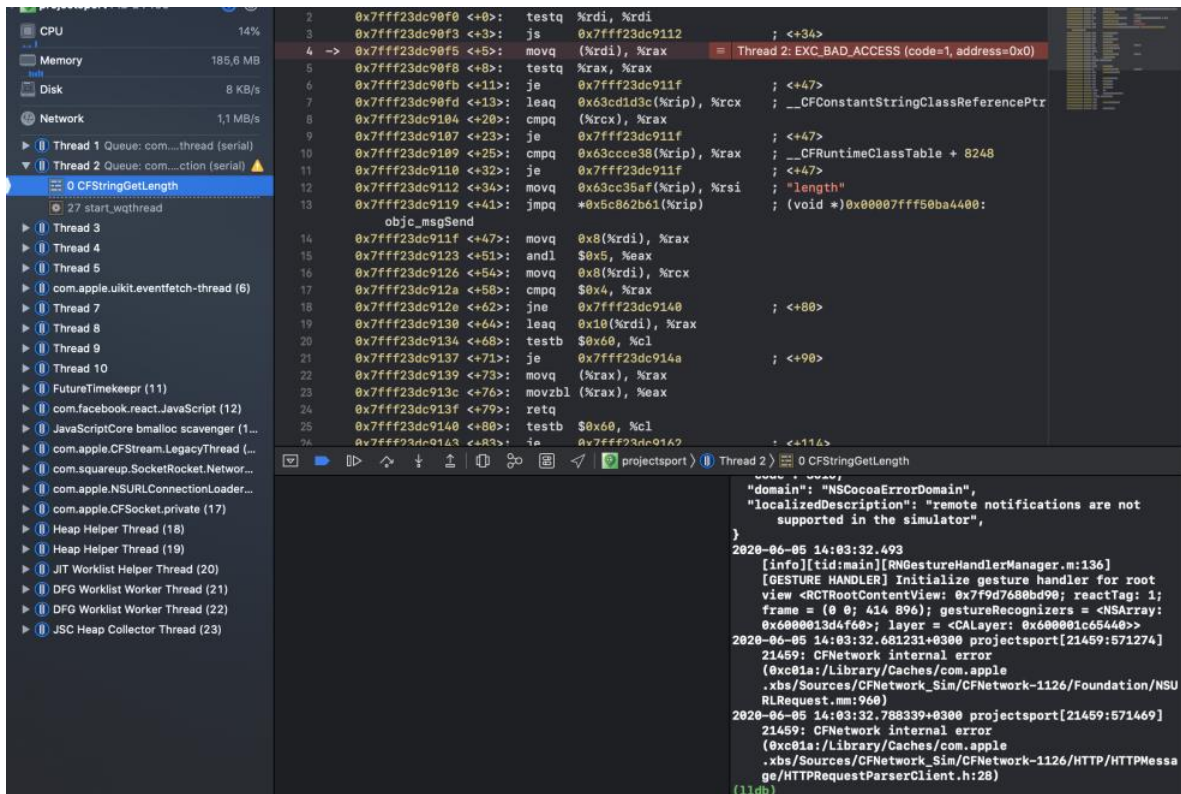
andreialecu commented on Jun 5, 2020

Okay, deleting `Podfile.lock` then running `pod repo update` then `pod install` got it going.

It's probably an artifact of using `patch-package` to update the package.

andreialecu commented on Jun 5, 2020 • edited

I seem to be seeing this crash:



Not sure what is wrong, could be a problem on my end. It did not occur previously though

The error above goes away if I remove the specific `FastImage` that fails to load, so it must be related to it.

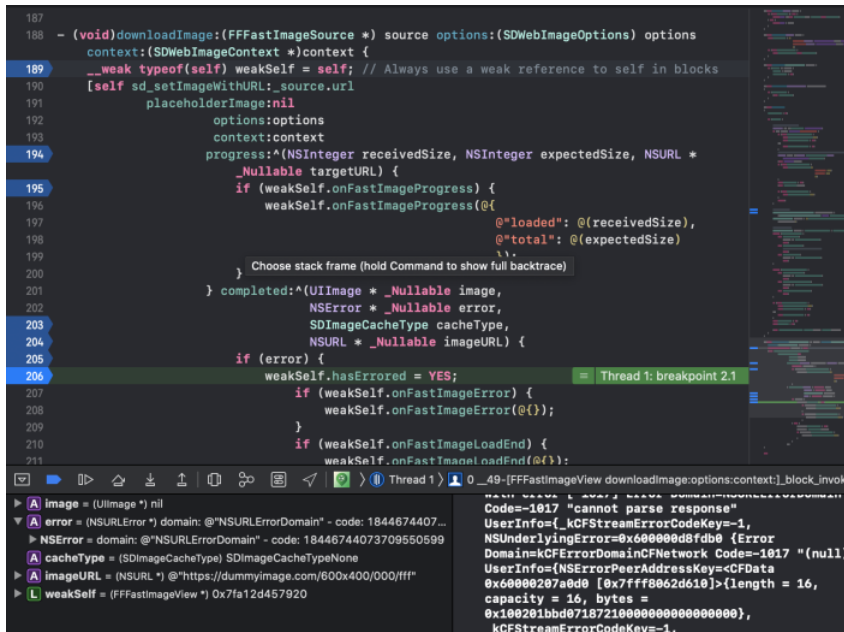
I'm trying to clean the ios build folder and will report back.

andreialecuc commented on Jun 5, 2020

No luck:

```
return (<FastImage
  style={{ height: 100, width: 100 }}
  source={{ url: "https://dummyimage.com/600x400/000/fff" }}
/>);
```

Enters error handler here:



But more interestingly:

```
return (<FastImage
  style={{ height: 100, width: 100 }}
/>);
```

```
source={{ uri: "https://loremflickr.com/320/240"}}
/>};
```

This crashes with `EXC_BAD_ACCESS` shown in a previous message. I suspect it's because the URL is a redirect and doesn't store the image directly.

Both images display properly if I replace `FastImage` with the react-native `Image`



andreialecu commented on Jun 5, 2020

Commenting out this line makes the images load:

<https://github.com/DylanVann/react-native-fast-image/pull/691/files#diff-c18dcaa41187abeeba66a03f2fd7b89aR193>

But that would make custom headers not work.

I guess something is incomplete related to the context.

dreampiggy commented on Jun 5, 2020

Contributor Author

Please paste the demo code or project so I can have a debug check tomorrow. Seems some more strange behavior inside.

andreialecu commented on Jun 6, 2020 • edited

@dreampiggy minimal repro at: <https://github.com/andreialecu/rnfastimage-691>

To get it going, run:

```
yarn
cd ios && pod install
yarn ios
```

Take a look at `App.js`, I left a comment there. Should show both the crash and the empty images.

(the changes in this PR are applied by patch-package automatically on yarn install, via the file in `patches/*`, so they are ready to use)

dreampiggy commented on Jun 6, 2020 • edited

Contributor Author

After using native debug code, I found the following issue:

1. Your example URL `https://dummyimage.com/600x400/000/fff`, use the `multipart/mixed` contents for image, which is not considered in current `SDWebImage` implementation. (Usually image server return `image/jpeg` or something, not a `multipart` contents, just raw blob data).

But React-Native provide the implementation, see: <https://github.com/facebook/react-native/blob/0.63-stable/React/Base/RCTMultipartDataTask.m#L101-L128>

So, which means, until we support the same code, you can not request the stream-style image server provider.

2. Your example URL `https://dummyimage.com/600x400/000/fff` which use a not-trusted SSL certificate. (You have wrong Let's Encrypt config?). Because of this, you will trigger the SSL authorization error for `SDWebImage`. Use `SDWebImageDownloaderAllowInvalidSSLCertificates` can skip this validation.

But however, `react-native-fast-image` does not have anything customizable for this behavior. For temp workaround, I can hardcode this to always skip.

dreampiggy commented on Jun 6, 2020

Contributor Author

The `multipart/mixed` case I can create a Feature Request on `SDWebImage` side. This must need Client Code update to handle.

dreampiggy commented on Jun 7, 2020

Contributor Author

Send TODO feature request: [SDWebImage/SDWebImage#3033](#)

Always bypass invalid SSL certificate error, same behavior like React...

✖ e7e3975

andreialecu commented on Jun 7, 2020 • edited

Strange. I don't see any SSL certificate error in Desktop Chrome or iOS Safari on neither of the images.

Could you clarify which one has the SSL issue?

andreialecu commented on Jun 7, 2020

I've updated the code as per your latest commit and the second image still crashes the app. (loremflickr)

I don't think there's any SSL issue, because the crash occurs when using a google maps photos url as well. The only thing in common is that both do a redirect.

andreialecu commented on Jun 7, 2020 • edited

Additionally, could you clarify where you see `multipart/mixed`? The raw headers for `https://dummyimage.com/600x400/000/fff` are:

```
$ curl https://dummyimage.com/600x400/000/fff -i
HTTP/2 200
server: nginx
```

```
date: Sun, 07 Jun 2020 09:53:34 GMT
content-type: image/png
content-length: 1783
cache-control: public, max-age=7776000
expires: Sat, 05 Sep 2020 09:08:05 +0000
last-modified: Sun, 07 Jun 2020 09:08:05 GMT
x-srcache-fetch-status: HIT
x-srcache-store-status: BYPASS
access-control-allow-origin: *
access-control-allow-methods: GET, POST, OPTIONS
access-control-allow-headers: DNT, User-Agent, X-Requested-With, If-Modified-Since, Cache-Control, Content-Type, Range
access-control-expose-headers: Content-Length, Content-Range
```

This is the second one:

```
$ curl https://loremflickr.com/320/240 -i -L

HTTP/2 302
date: Sun, 07 Jun 2020 10:16:55 GMT
content-type: text/html; charset=UTF-8
set-cookie: __cfduid=d73251b784a012e5458b99e9b1bd8b59d1591525014; expires=Tue, 07-Jul-20 10:16:54 GMT; path=/; domain=.loremflickr.com; HttpOnly; SameSite=Lax; Secure
expires: Thu, 19 Nov 1981 08:52:00 GMT
cache-control: no-store, no-cache, must-revalidate
pragma: no-cache
access-control-allow-origin: *
set-cookie: PHPSESSID=8845ac386084cd41085da14df128f2d8; path=/
location: /cache/resized/65535_49600375582_8b90c8f3ca_n_320_240_nofilter.jpg
vary: User-Agent, Accept-Encoding
cf-cache-status: DYNAMIC
cf-request-id: 032fe195100000290c6bb1f200000001
expect-ct: max-age=604800, report-uri="https://report-uri.cloudflare.com/cdn-cgi/beacon/expect-ct"
server: cloudflare
cf-ray: 59f99ece8c54290c-OTP


HTTP/2 200
date: Sun, 07 Jun 2020 10:16:55 GMT
content-type: image/jpeg
content-length: 16788
set-cookie: __cfduid=d545204559e173f7cf0ac3aa276f227991591525015; expires=Tue, 07-Jul-20 10:16:55 GMT; path=/; domain=.loremflickr.com; HttpOnly; SameSite=Lax; Secure
last-modified: Thu, 04 Jun 2020 11:15:25 GMT
etag: "4194-5a74042e6aa73"
cache-control: public, max-age=604800
expires: Sat, 04 Jul 2020 12:28:58 GMT
vary: User-Agent
access-control-allow-origin: *
access-control-allow-headers: origin, x-requested-with, content-type
access-control-allow-methods: GET, POST, OPTIONS, DELETE, PUT
cf-cache-status: HIT
accept-ranges: bytes
cf-request-id: 032fe196920000290c6bb30200000001
expect-ct: max-age=604800, report-uri="https://report-uri.cloudflare.com/cdn-cgi/beacon/expect-ct"
server: cloudflare
cf-ray: 59f99ed0ed81290c-OTP
```

Seems like all the data would be there. I see `content-type: image/png` and `image/jpeg`. Also both images load with the `master` version of `react-native-fast-image`, but they're broken with this PR.

andreialecu commented on Jun 7, 2020

```
[self sd_setImageWithURL:[source.url
placeholderImage:nil
options:options
context:context
progress:^(NSInteger receivedSize, NSInteger expectedSize, NSURL * _Nullable targetURL) {
```

Again, I just verified that commenting out `context` here makes both images load perfectly. But the headers are (obviously) going to be missing.

 dreampiggy mentioned this pull request on Jun 7, 2020

Base64 WebP on iOS #567

[Open](#)

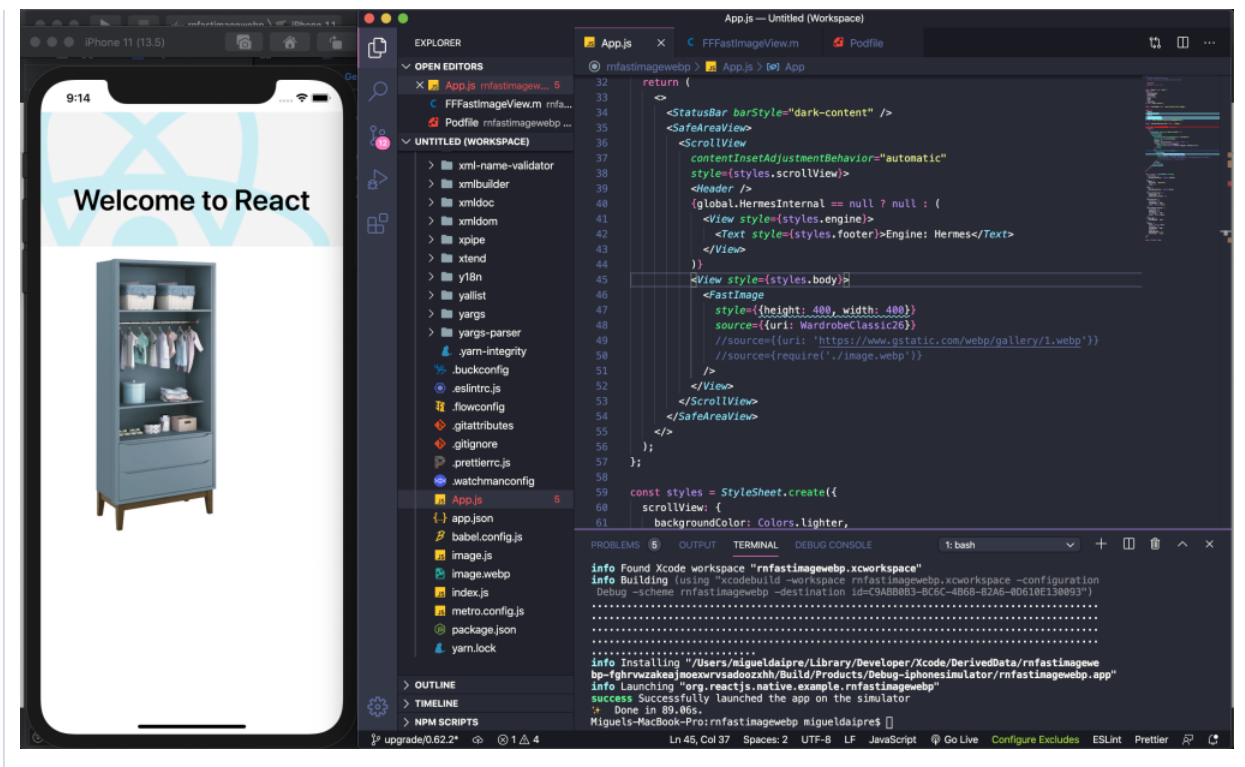
migueldaipre commented on Jun 8, 2020

I did the tests as described by you, and it worked perfectly with local Base64 WebP images.
I could only notice how the @andreialecu said, without commenting on the context the external images do not work.

I got the same error: EXC_BAD_ACCESS (code = 1, address = 0x0)
CFNetwork internal error

I wanted to understand more to help. If you can guide me.

I had problems with Flipper, I changed in PodFile to 0.37.0.



andreialecu commented on Jun 10, 2020

Hey @dreampiggy, was just wondering if you saw the latest discussion here.

dreampiggy commented on Jun 10, 2020

Contributor Author

Busy working in daily job...

For the strange crash, I doubt there are something race condition for iOS native code to mutable copy && copy the NSURLRequest .

If I can reproduce the crash, maybe it's better to do not mutable copy and re-create a new NSURLRequest instead. I'll have a try and verify, if that can fix, I'll update the MR.

andreialecu commented on Jun 10, 2020 • edited

Sure, I understand.

But there's not just the crash, neither of the images load with these changes unless context is removed. I think there's something else wrong with it.

Edit: the multipart/mixed and SSL error may be related to the great firewall perhaps? Don't see those issues myself.

dreampiggy commented on Jun 10, 2020

Contributor Author

Seems the fucky fact that mutableCopy is dangerous.

I write the exact same code in logic, to re-construct a new NSURLRequest , instead of mutable copy the original one and modify, the crash disappears. :)

```
146 // Set headers.
147 NSDictionary *headers = _source.headers;
148 SDWebImageDownloaderRequestModifier *requestModifier = [SDWebImageDownloaderRequestModifier requestModifierWithBlock:^(NSURLRequest *
149     _Nullable(NSURLRequest * _Nonnull request) {
150         if (headers.count > 0) {
151             // Re-create new NSURLRequest, do not use 'mutableCopy' because there maybe something race condition cause crash
152             NSMutableURLRequest *mutableRequest = [NSMutableURLRequest requestWithURL:request.URL cachePolicy:request.cachePolicy
153                 timeoutInterval:request.timeoutInterval];
154             mutableRequest.HTTPShouldHandleCookies = request.HTTPShouldHandleCookies;
155             mutableRequest.HTTPShouldUsePipelining = request.HTTPShouldUsePipelining;
156             mutableRequest.allHTTPHeaderFields = headers;
157             return [mutableRequest copy];
158         } else {
159             return request;
160         }
161     }];
162 // SDWebImageDownloaderRequestModifier *requestModifier = [[SDWebImageDownloaderRequestModifier alloc] initWithHeaders:headers];
163 SDWebImageContext *context = @{SDWebImageContextDownloadRequestModifier : requestModifier};
```

2

Fix the implementation of request modifier on SDWebImage, which have ...

c69a95e

dreampiggy commented on Jun 11, 2020 • edited

Contributor Author

@andreialecu

Confirmed the bug because of SDWebImage's usage.

The `NSURLRequest.HTTPMethod` should not use `nil`, should use `GET` for default value.

I'll provide a hotfix on `SDWebImage` for this.

This PR already apply the patch, and can be merged now. You can have a try again with the latest branch commit.



andreialecu reviewed on Jun 11, 2020

[View changes](#)

ios/FastImage/FFFastImageView.m Outdated

[Show resolved](#)

andreialecu commented on Jun 11, 2020 • edited

@dreampiggy awesome, I just tested it and it loads all images that were previously not loading, and headers are not leaking any more. ~~There's one minor concern about skipping invalid SSL certs, which I added a comment about.~~ (edit: this has been reverted, PR should be good to go)

[Revert the changes to bypass SSL error](#)

[cbac214](#)

[dreampiggy](#) mentioned this pull request on Jun 11, 2020

Fix the issue that the `NSURLRequest` method should not be `nil`, which may cause Crash `SDWebImage/SDWebImage#3037`

[Merged](#)

[8 tasks](#)

[andreialecu](#) mentioned this pull request on Jun 12, 2020

[iOS] Headers for one request are sent for all subsequent requests, even if no headers specified #690

[Closed](#)

gwenoleR commented on Jul 15, 2020

Hi guys, any news about this fix ?

[dreampiggy](#) commented on Jul 15, 2020

[Contributor](#) [Author](#)

I want to merge this, but seems not get any reply from the maintainer yet.

[Merge branch 'master' into bugfix_ios_sd_header_per_image_request](#)

[1ad69f1](#)

[dreampiggy](#) commented on Jul 17, 2020 • edited

[Contributor](#) [Author](#)

@DylanVann I'm glad to see you back here for continue maintains.

This PR not only just used for logic fix, but also fix the security issue talked about by [#690](#). This means you leaks the HTTP headers like Cookie to other domain.

[Remove the unused Podfile.lock changes](#)

[0826cac](#)

[dreampiggy](#) force-pushed the `bugfix_ios_sd_header_per_image_request` branch from [da534b4](#) to [0826cac](#) 2 years ago

[Compare](#)



[dreampiggy](#) commented on Jul 17, 2020

[View changes](#)

.gitignore

[Show resolved](#)

DylanVann commented on Jul 17, 2020

[Owner](#)

@dreampiggy Thanks so much for implementing this. This was an aspect of `SDWebImage` that always frustrated me, great that it's fixed.

[1](#) [3](#)

[DylanVann](#) changed the title ~~Fix the implementation on iOS for per-image-request-header setup~~ feat: on iOS allow for for per-image-request-headers on Jul 17, 2020

[DylanVann](#) changed the title ~~feat: on iOS allow for for per-image-request-headers~~ feat(ios): allow for for per-image-request-headers on Jul 17, 2020

[DylanVann](#) merged commit [4a7cd64](#) into [DylanVann:master](#) on Jul 17, 2020

[github-actions](#) [bot](#) pushed a commit that referenced this pull request on Jul 17, 2020

[release\(version\): Release 8.3.0 \[skip ci\]](#) ...

[866274f](#)

[github-actions](#) [bot](#) commented on Jul 17, 2020

🚀 This PR is included in version 8.3.0 🚀

The release is available on:

- [npm package \(@latest dist-tag\)](#)
- [GitHub release](#)

Your [semantic-release](#) bot 📦 🚀

🔗 [github-actions](#) (bot) added the **released** label on Jul 17, 2020

[dreampiggy](#) commented on Jul 17, 2020 • edited

[Contributor](#) [Author](#)

[@dreampiggy](#) Thanks so much for implementing this. This was an aspect of SDWebImage that always frustrated me, great that it's fixed.

Note this MR also add the support for any external image format support for `base64://` on iOS. So maybe it's better to update the release changelog as well.

Previously, only the HTTP url can load WebP/AVIF/SVG/PDF, etc.

The iOS user just need to register the coder plugin following [SDWebImage's Coder Plugin](#), then all things done.

❤️ 1

🔗 [github-actions](#) (bot) pushed a commit to perrystreetsoftware/react-native-fast-image that referenced this pull request on Aug 3, 2021

🚀 `release(version): Release 6.0.0 [skip ci]` ...

ed0f304

Reviewers

[andreialecu](#)



[DylanVann](#)



Assignees

No one assigned

Labels

released

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

👉 Some images are not showing.

5 participants

