Creation, queueing, and processing

Ceci n'est pas une ThumbnailDownloader

<H>>
Handler
(H, Bitmap) → Unit

fragment lifecycle observer H String

request handler

view lifecycle observer

Looper



PGF on create

constructor

<PhotoHolder>
response handler
on photo downloaded

request handler

fragment lifecycle observer

holder url

request map

Request map is initialized during declaration

Thumbnail Downloader is created inside the on-create method of PhotoGallery Fragment

The PhotoHolder type, the response-handler object, and the on-photo-downloaded function are passed in view lifecycle observer

Looper



PGF on create

<PhotoHolder> response handler on photo downloaded fragment lifecycle observer

request map

holder url

request handler

The on-create method also creates the fragment lifecycle observer

constructor

view lifecycle observer

The fragment lifecycle observer is notified that on-create is running, so the observer's setup method is run

Looper



The setup method starts the thread and prepares the looper

on looper prepared system call

<PhotoHolder> response handler on photo downloaded fragment lifecycle observer

request map

url

view lifecycle observer holder

request handler

The thread's lifecycle method on-looper-prepared sets up the request handler, including its handle-message function

> handle message

Looper



PGF on view created

<PhotoHolder>
response handler
on photo downloaded

request handler

fragment lifecycle observer request map

view lifecycle observer

holder	url

creates the view lifecycle observer

PGF's on-view-created

handle message

Looper



<PhotoHolder>
response handler
on photo downloaded

fragment lifecycle observer request map

_

request handler

view lifecycle observer holder url

All parts of thumbnail downloader are now set up and the it is ready to receive and process messages (tasks)

handle message Looper



<PhotoHolder>
response handler
on photo downloaded

fragment lifecycle observer holder url

PH #21 ...abc.jpg

PH #22 ...xyz.jpg

PH #23 ...def.jpg

request handler

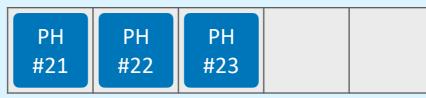
Let's assume there are already a few messages in the queue

And let's look at the process for adding another message to the queue

handle message view lifecycle observer

Looper





enqueuing a message

A view-holder
object and a url
string are passed
to queue thumbnail

PH #24 ...uvw.jpg

PGF Adapter on bind view holder

queue thumbnail

ThumbnailDownloader

<PhotoHolder>
response handler
on photo downloaded

request handler

fragment lifecycle observer

view lifecycle observer request map

holder	url	
PH #21	abc.jpg	
PH #22	xyz.jpg)
PH #23	def.jpg	

...uvw.jpg

The pair is added to the request map

The handler creates a new message

handle message Looper

PH #24



what: MESSAGE DOWNLOAD

ojb: photo holder #24

target: request handler

Message Queue

PH #21

PH #22 PH #23 PH #24 The message is added to the end of the queue

<PhotoHolder>
response handler
on photo downloaded

fragment lifecycle observer request map

view lifecycle

observer

holder url

PH #21 ...abc.jpg

PH #22 ...xyz.jpg

PH #23 ...def.jpg

PH #24 ...uvw.jpg

request handler

Now let's see how a message from the queue gets processed

handle message Looper



Message Queue

PH #21

PH #22 PH #23 PH #24

processing a message

Task 2: execute on photo downloaded, which binds the bitmap image to the photo holder

ThumbnailDownloader

<PhotoHolder>
response handler
on photo downloaded

fragment lifecycle observer

view

lifecycle

observer

PH

#21

holder	url
PH #21	abc.jpg
PH #22	xyz.jpg
PH #23	def.jpg

...uvw.jpg

request map

5

Task 1: remove the (holder, url) pair from the map

request handler

The handler gets the holder from message property obj

It uses holder to get the URL from request map, then uses the URL to fetch the bitmap from the network

Looper

PH #24



The looper gets the next message from the head of the queue and sends it to the request handler's handle message function

Finally, it uses the response handler to post a runnable object, all code inside will executed on the UI

thread. There are two

main tasks...

handle message

PH	
----	--