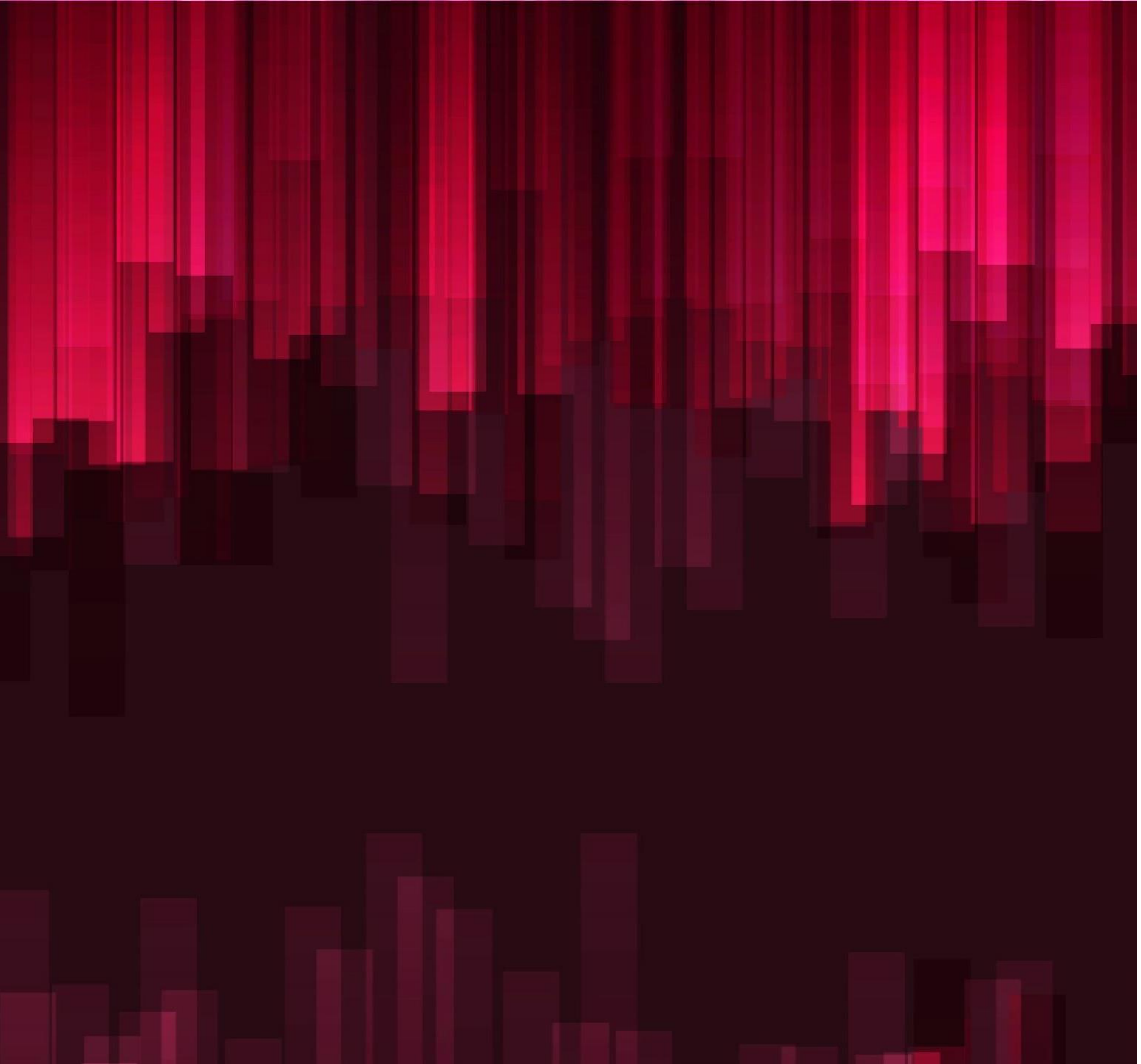


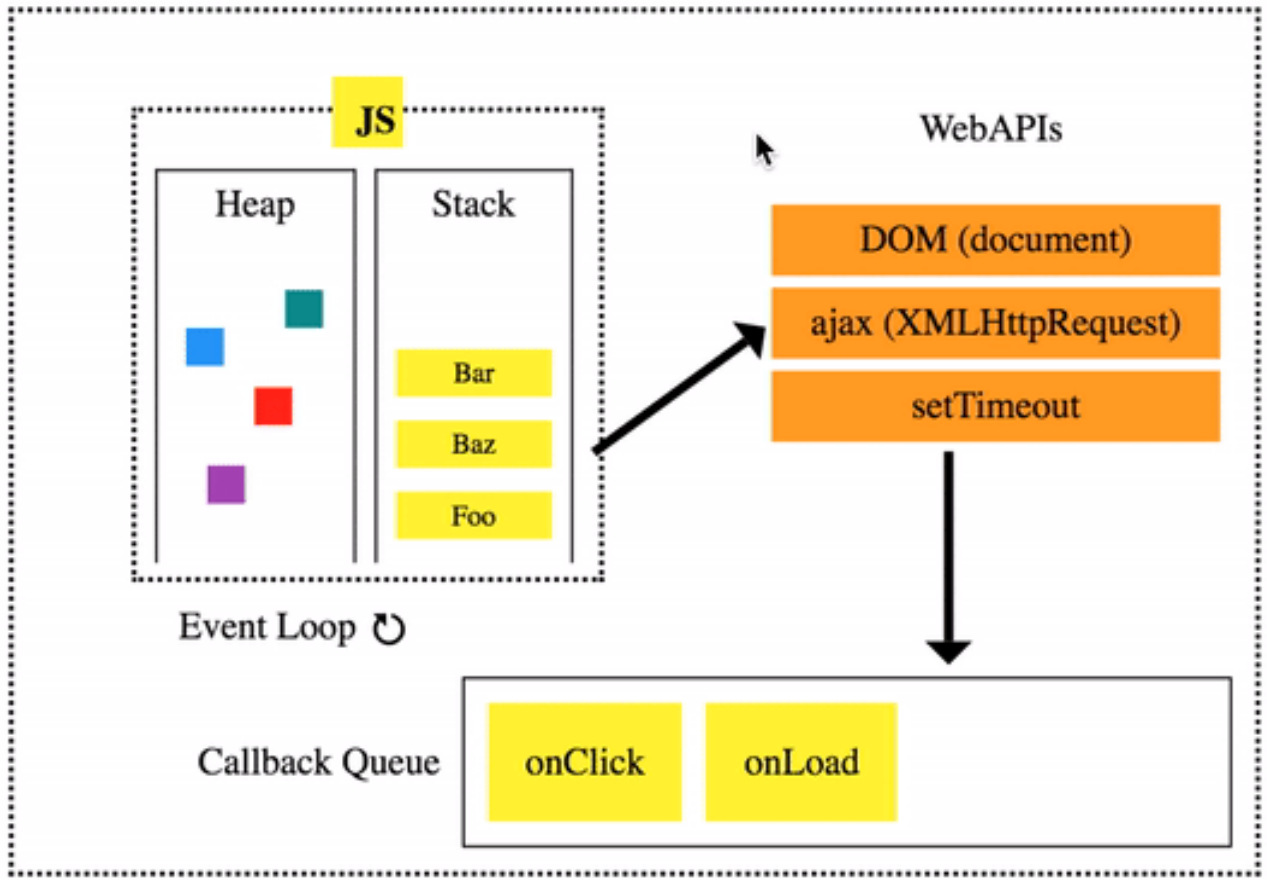
Lab 6

ASYNCHRONOUS PROGRAMMING

SINGLE THREADED JS

- JavaScript is inherently single threaded
- So in order to execute requests that are non blocking such as setTimeout and DOM events, it uses asynchronous programming
- The JS engine has an event queue which keeps track of all the non blocking requests that are completed
- If the callstack is empy, it schedules the tasks from callback queue

Event Loop

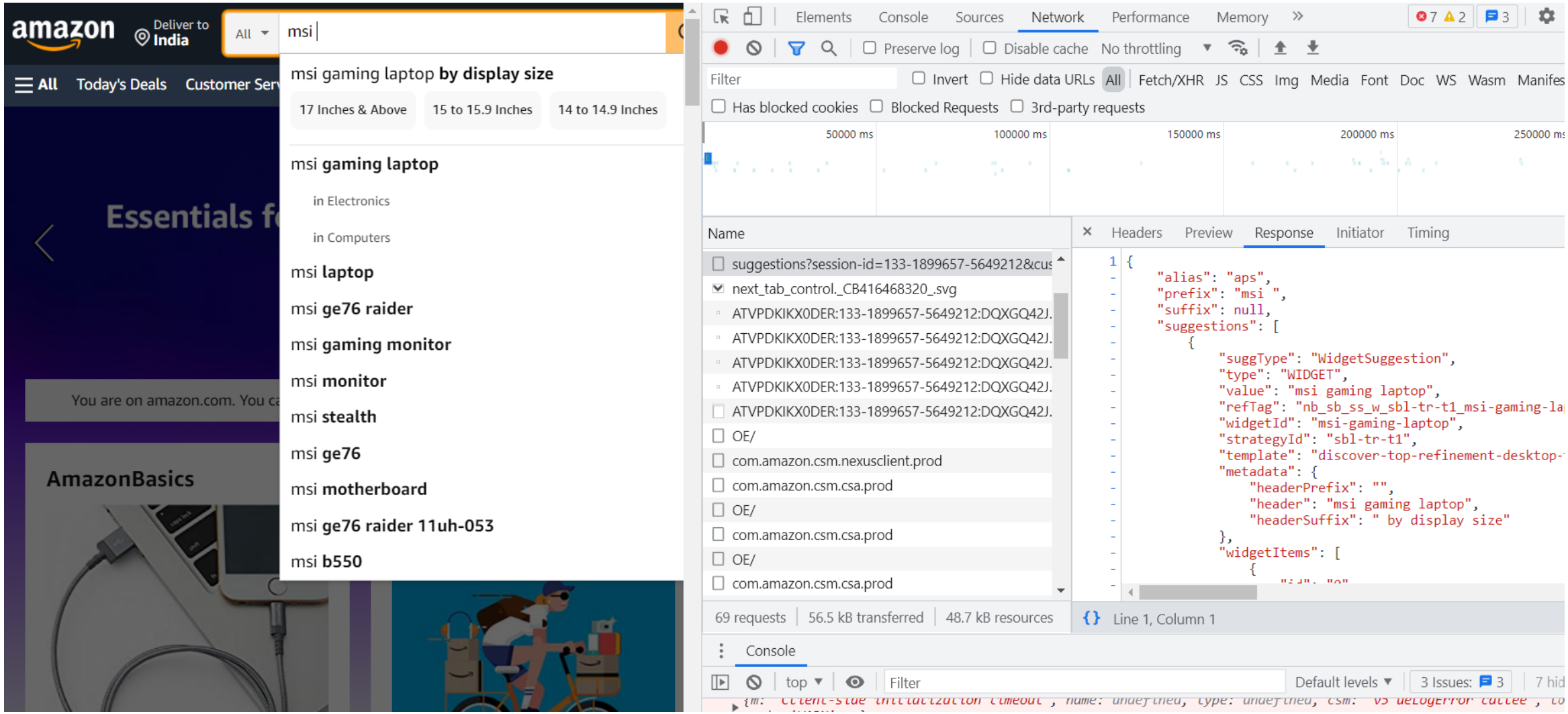

JS
Heap
Stack
Bar
Baz
Foo
Event Loop
WebAPIs
DOM (document)
ajax (XMLHttpRequest)
setTimeout
Callback Queue
onClick
onLoad

EXAMPLE

http://latentflip.com/loupe/

AJAX

- It is used to communicate with the server and update the page without having to refresh it
- Uses the XMLHttpRequest object to communicate with the servers
- Send and receive data to the servers in the background asynchronously


amazon
Deliver to India
All
msi
msi gaming laptop by display size
17 Inches & Above
15 to 15.9 Inches
14 to 14.9 Inches
msi gaming laptop
in Electronics
in Computers
msi laptop
msi ge76 raider
msi gaming monitor
msi monitor
msi stealth
msi ge76
msi motherboard
msi ge76 raider 11uh-053
msi b550
All
Today's Deals
Customer Serv
Essentials f
You are on amazon.com. You ca
AmazonBasics
Elements
Console
Sources
Network
Performance
Memory
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
All
Fetch/XHR
JS
CSS
Img
Media
Font
Doc
WS
Wasm
Manifes
Has blocked cookies
Blocked Requests
3rd-party requests
50000 ms
100000 ms
150000 ms
200000 ms
250000 ms
Name
suggestions?session-id=133-1899657-5649212&cus
next_tab_control._CB416468320_.svg
ATVPDKIKX0DER:133-1899657-5649212:DQXGQ42J.
ATVPDKIKX0DER:133-1899657-5649212:DQXGQ42J.
ATVPDKIKX0DER:133-1899657-5649212:DQXGQ42J.
ATVPDKIKX0DER:133-1899657-5649212:DQXGQ42J.
ATVPDKIKX0DER:133-1899657-5649212:DQXGQ42J.
OE/
com.amazon.csm.nexusclient.prod
com.amazon.csm.csa.prod
OE/
com.amazon.csm.csa.prod
OE/
com.amazon.csm.csa.prod
69 requests
56.5 kB transferred
48.7 kB resources
Headers
Preview
Response
Initiator
Timing
{
"alias": "aps",
"prefix": "msi ",
"suffix": null,
"suggestions": [
{
"suggType": "WidgetSuggestion",
"type": "WIDGET",
"value": "msi gaming laptop",
"refTag": "nb_sb_ss_w_sbl-tr-t1_msi-gaming-la
"widgetId": "msi-gaming-laptop",
"strategyId": "sbl-tr-t1",
"template": "discover-top-refinement-desktop-
"metadata": {
"headerPrefix": "",
"header": "msi gaming laptop",
"headerSuffix": " by display size"
},
"widgetItems": [
{
Line 1, Column 1
Console
top
Filter
Default levels
3 Issues: 3

PROMISES

- A promise is an object that may produce a single value some time in the future
- A promise may be in one of 3 possible states: fulfilled, rejected, or pending
- Promise users can attach callbacks to handle the fulfilled value or the reason for rejection

the call of .then(handler) always returns a promise:

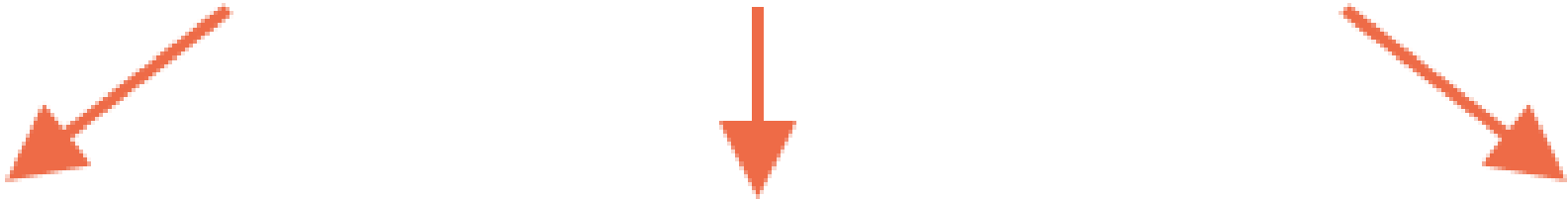
state: "pending"
result: undefined

if handler ends with…

return value

throw error

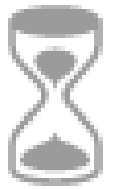
return promise



that promise settles with:

state: "fulfilled"
result: value

state: "rejected"
result: error



…with the result
of the new promise…

MISC PROMISE METHODS

- Promise.all():
  - Takes list of promises as input
  - Returns a promise when all of the promises have been resolved
  - If any one of the promises are rejected, the promise itself is rejected
- Promise.any()
  - Similar to promise.all() but returns a promise when any one of the promises have been resolved
  - It rejects when all of the promises are rejected