Conditional Modules & Dynamic Bundling

A NETFLIX ORIGINAL


NETFLIX
THE RANCH
NETFLIX
Master of None
NETFLIX
Dead to Me


#netflixeverywhere
NETFLIX

Conditional Modules & Dynamic Bundling : A NETFLIX Original

Rajat Kumar
rajatk@netflix.com | @rajatkumar

Contents

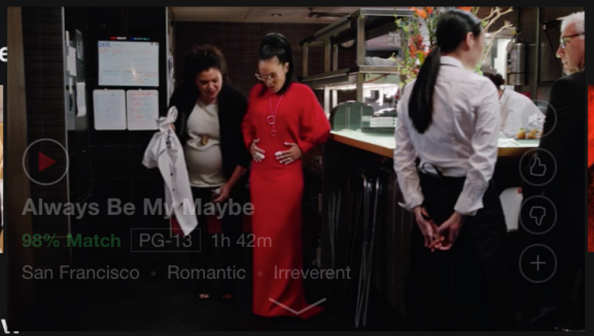
data driven

product development

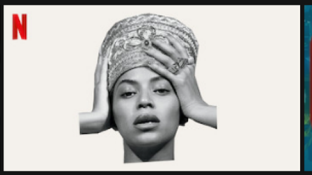
Movies

Genres

New Release



Trending Now

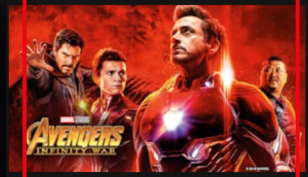


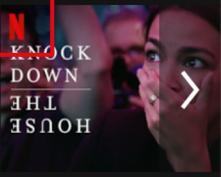
Blockbuster Movies

Movies

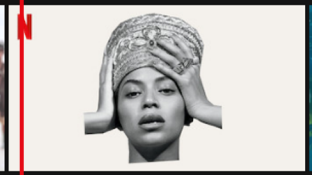
Genres

New Release





Trending Now



Blockbuster Movies

AB tests gives each subscriber a unique and different User Experience

User Experience can also vary based on many other dimensions

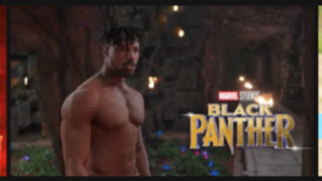
NETFLIX Home TV Shows **Movies** Recently Added My List

Titles, people, genres

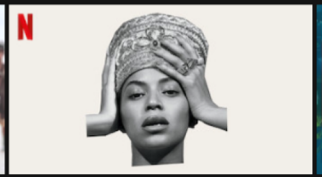
Movies

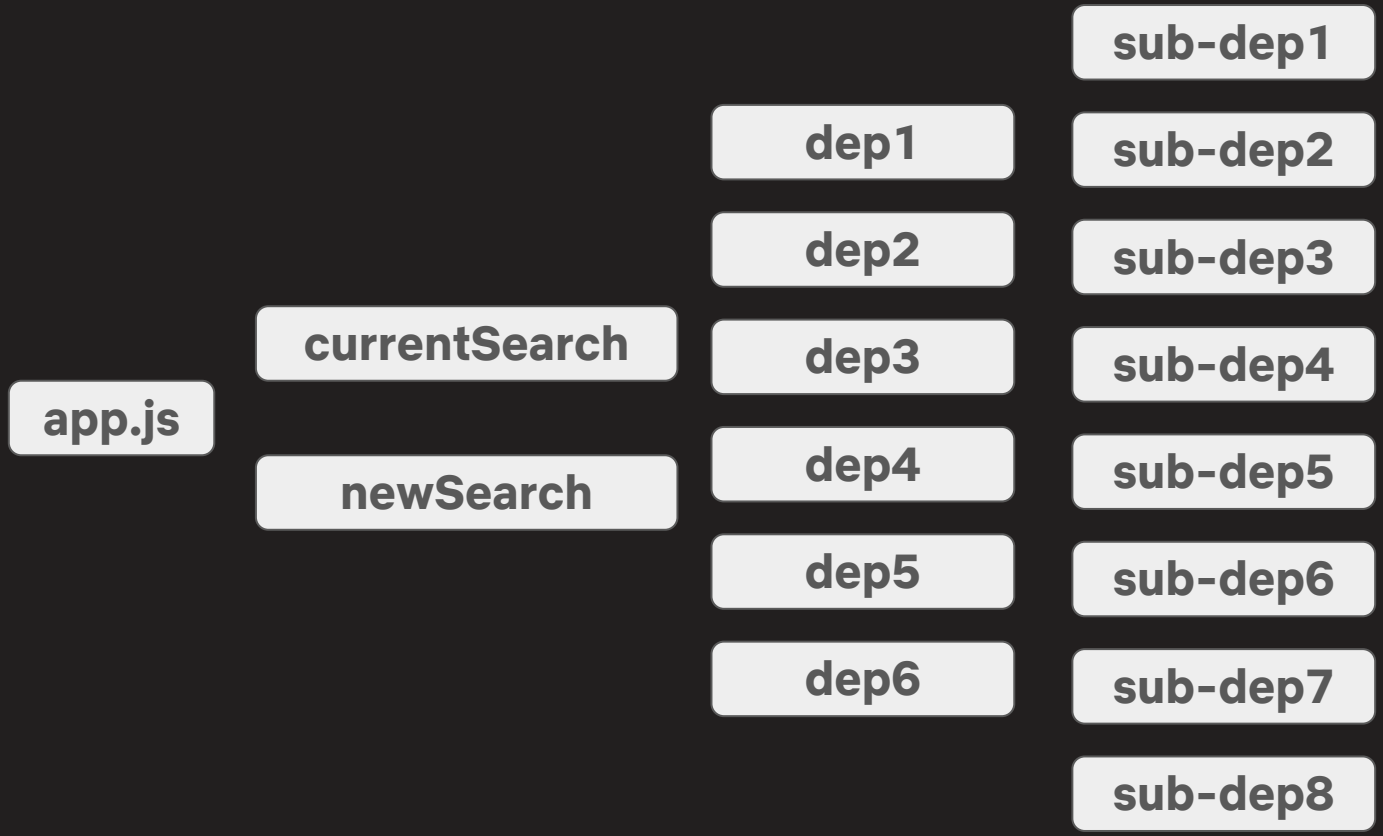
Genres

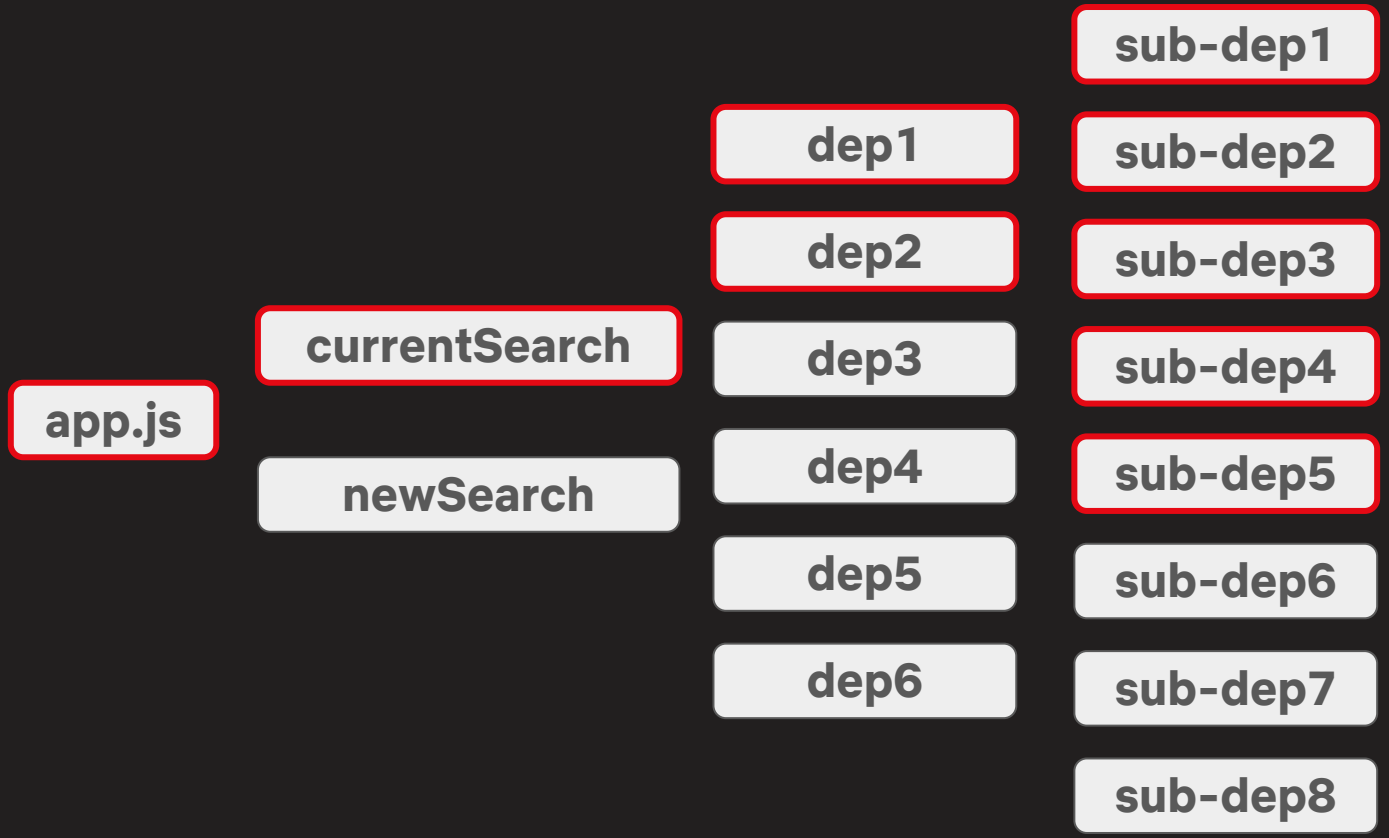
New Releases

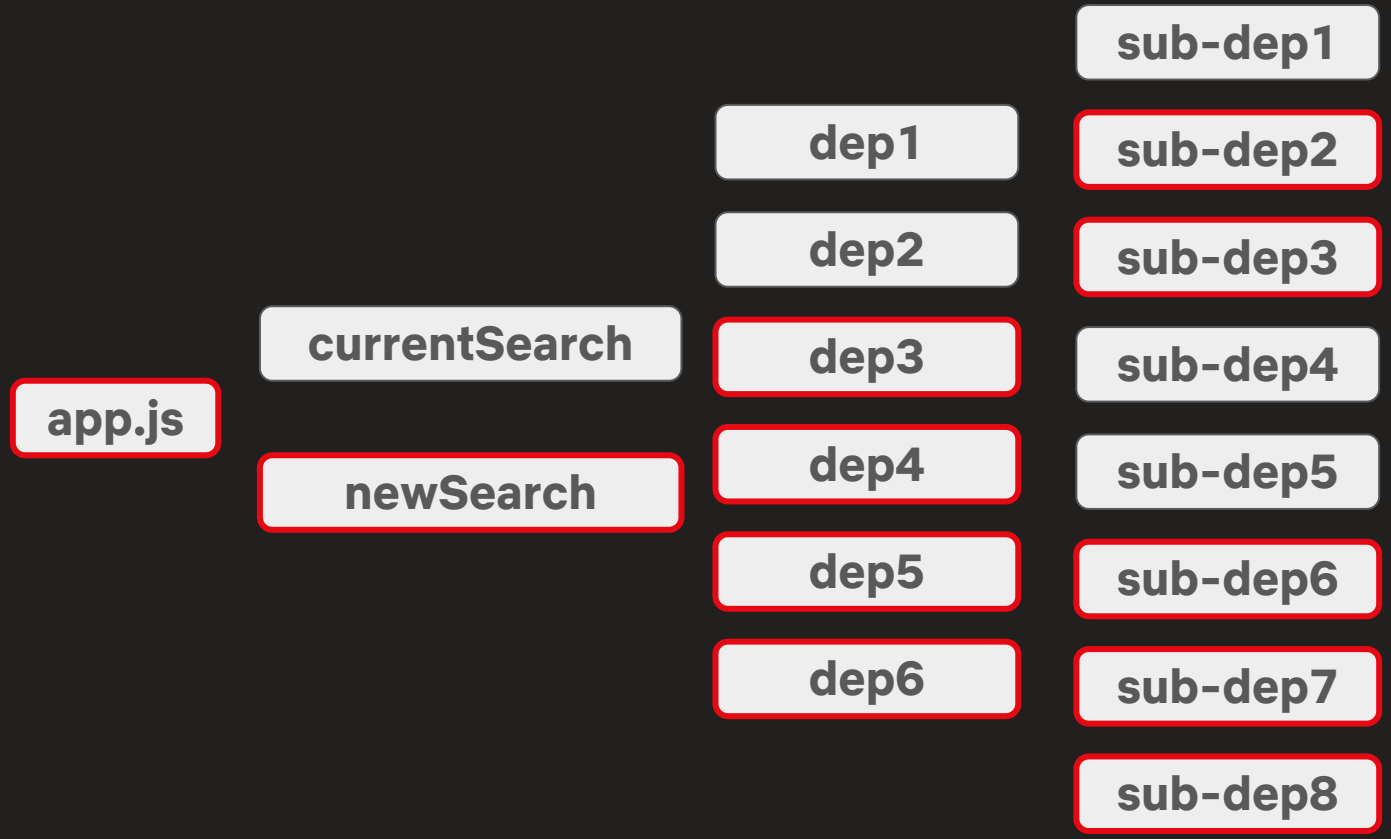


Trending Now




app.js
currentSearch
newSearch
dep1
dep2
dep3
dep4
dep5
dep6
sub-dep1
sub-dep2
sub-dep3
sub-dep4
sub-dep5
sub-dep6
sub-dep7
sub-dep8


app.js
currentSearch
newSearch
dep1
dep2
dep3
dep4
dep5
dep6
sub-dep1
sub-dep2
sub-dep3
sub-dep4
sub-dep5
sub-dep6
sub-dep7
sub-dep8


app.js
currentSearch
newSearch
dep1
dep2
dep3
dep4
dep5
dep6
sub-dep1
sub-dep2
sub-dep3
sub-dep4
sub-dep5
sub-dep6
sub-dep7
sub-dep8

app.js

```
....
import CurrentSearch from 'currentSearch';
import NewSearch from 'newSearch';
....

export ...
```

render

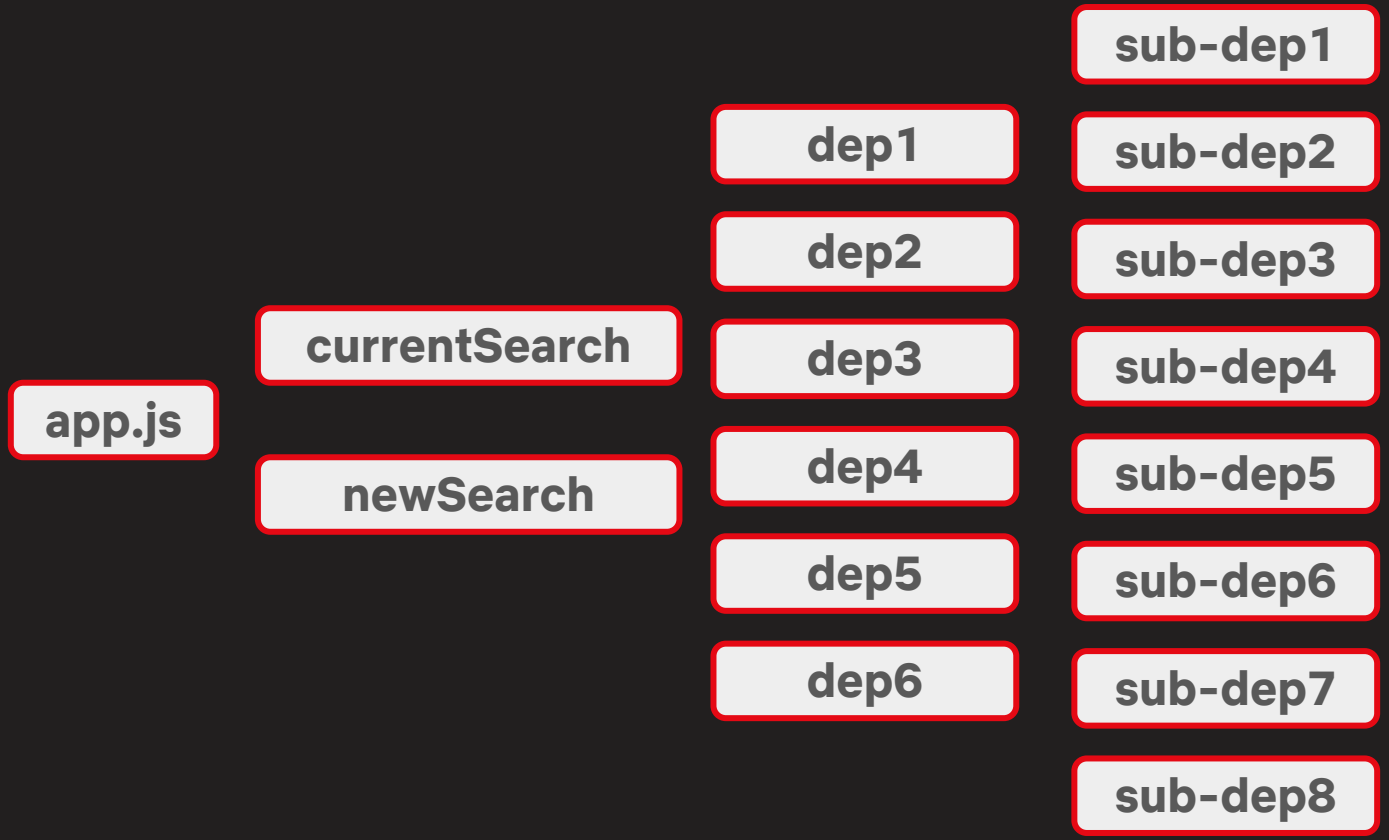
```
const SearchComponent = (someFlag) ? CurrentSearch: NewSearch;

...

<SearchComponent>

...

</SearchComponent>
```


app.js
currentSearch
newSearch
dep1
dep2
dep3
dep4
dep5
dep6
sub-dep1
sub-dep2
sub-dep3
sub-dep4
sub-dep5
sub-dep6
sub-dep7
sub-dep8

The Problem

Our Apps are huge

WebUI ~ 10 MB
TVUI ~ 30 MB

(when we package every experience)

Slower Performing Apps

Longer Time-to-interactive (TTI)

Slower Performing Apps

Longer Time-to-interactive (TTI)

Summary : Poor User Experience

How can we improve the User Experience?

Maybe we can build “smaller” packages that are very specific to the user’s AB tests?

But, can we?

Unfortunately, No!

Netflix runs hundreds of AB tests

Test 1

A

B

Test 1

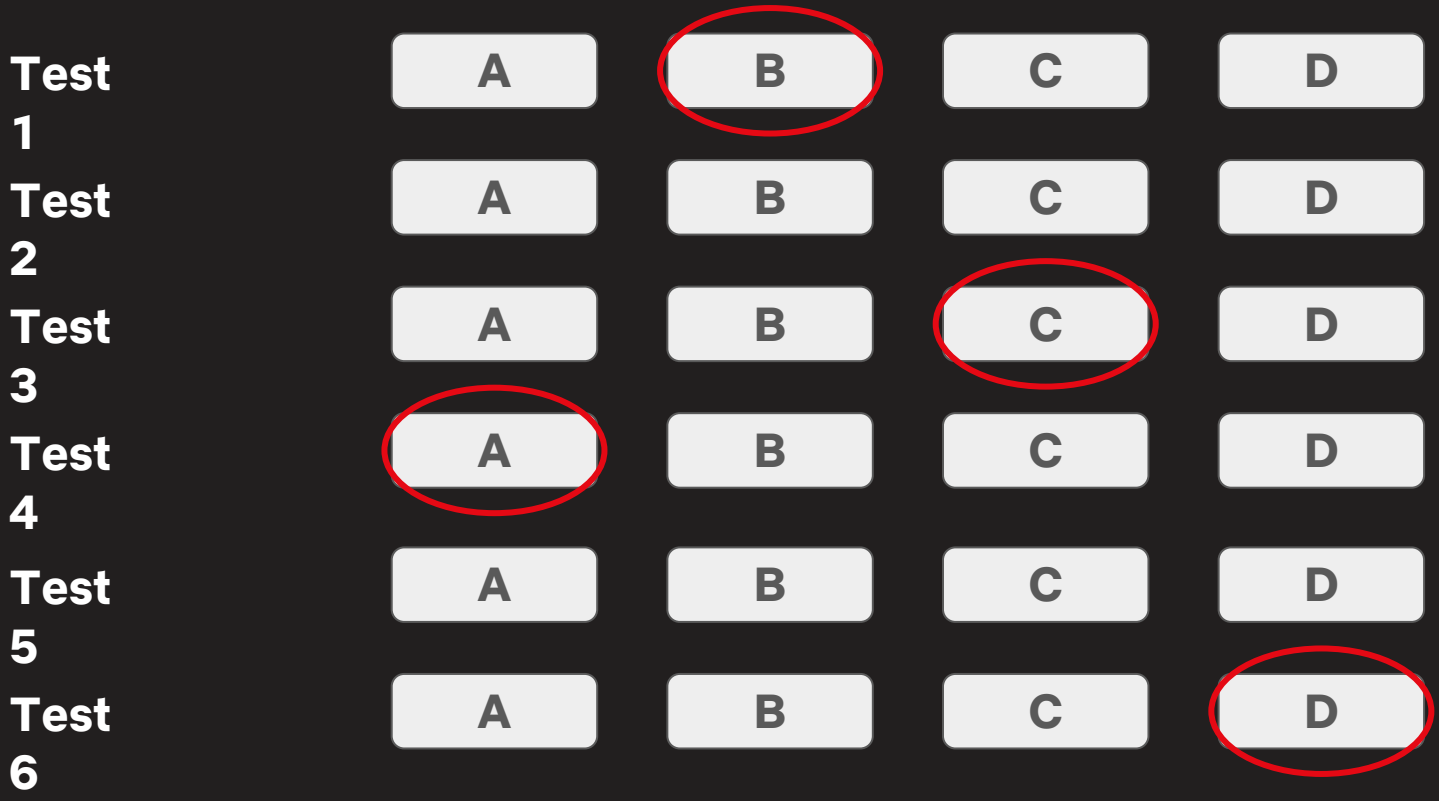
A

B

C

D

Test 1	A	B	C	D
Test 2	A	B	C	D
Test 3	A	B	C	D
Test 4	A	B	C	D
Test 5	A	B	C	D
Test 6	A	B	C	D


Test 1
A
B
C
D
Test 2
A
B
C
D
Test 3
A
B
C
D
Test 4
A
B
C
D
Test 5
A
B
C
D
Test 6
A
B
C
D

Netflix runs hundreds of AB tests

How many packages do we need to build and serve?

Netflix runs hundreds of AB tests

How many packages do we need to build and serve?

$$|S_1| \cdot |S_2| \cdots |S_n| = |S_1 \times S_2 \times \cdots \times S_n|$$

Netflix runs ~~hundreds~~ fifteen AB tests

$$|S_1| \cdot |S_2| \cdots |S_{15}| = |S_1 \times S_2 \times \cdots \times S_{15}|$$

Netflix runs ~~hundreds~~ fifteen AB tests

$$|S_1| \cdot |S_2| \cdots |S_{15}| = |S_1 \times S_2 \times \cdots \times S_{15}|$$

$$4^{15} =$$

Netflix runs ~~hundreds~~ fifteen AB tests

$$|S_1| \cdot |S_2| \cdots |S_{15}| = |S_1 \times S_2 \times \cdots \times S_{15}|$$

$$4^{15} = 1{,}073{,}741{,}824$$

that's over a Billion!

Netflix runs hundreds of AB tests

$$|S_1| \cdot |S_2| \cdots |S_n| = |S_1 \times S_2 \times \cdots \times S_n|$$

Netflix runs hundreds of AB tests

$$|S_1| \cdot |S_2| \cdots |S_n| = |S_1 \times S_2 \times \cdots \times S_n|$$

$$4^{100} = \sim 1.606938e{+}60$$

Netflix runs hundreds of AB tests

How many packages do we need to build and serve?

Billions

We need to build billions of packages

Time taken for 1 build ~ 2mins

We need to build billions of packages

Time taken for 1 build ~ 2mins

Time taken for billions builds ~ 🤯

We need to build billions of packages

Time taken for 1 build ~ 2mins

Time taken for billions builds ~ 🤯

Build & Deploy for Web and TV ~ 🤯 🤯

We need to build billions of packages

Time taken for 1 build ~ 2mins

Time taken for billions builds ~ 🤯

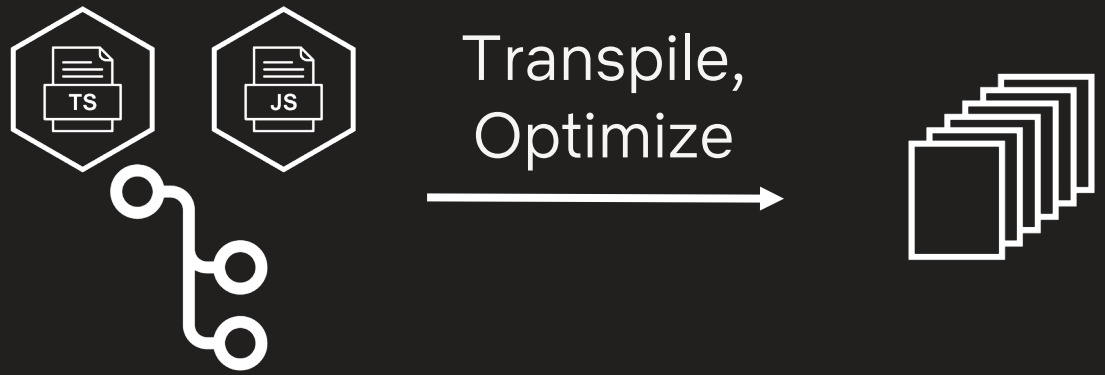
Build & Deploy for Web and TV ~ 🤯 🤯

Build & Deploy 3 times a week ~ 🤯 🤯 🤯

The Problem

Serve billions of smaller packages that are very specific to the User's AB tests

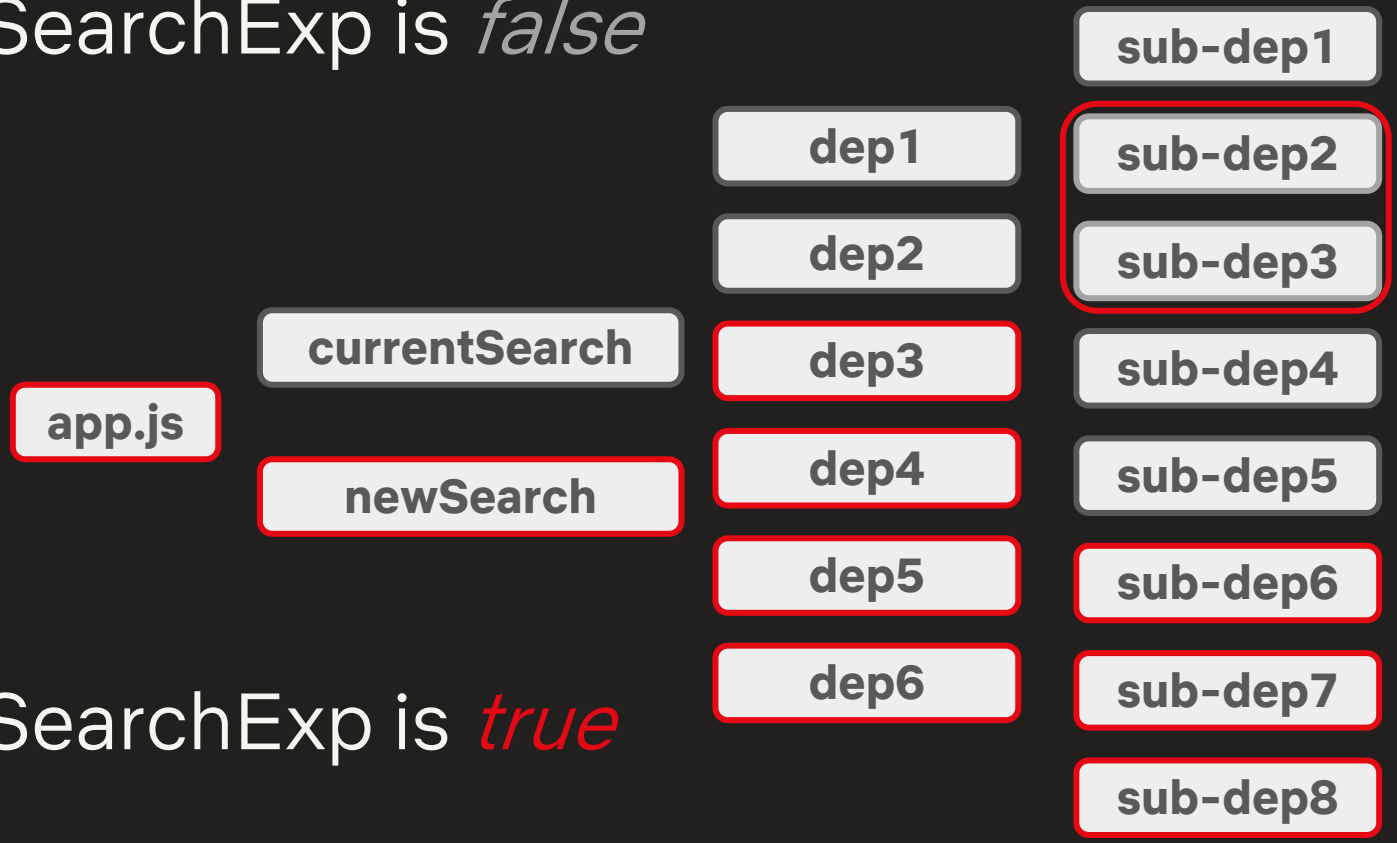
The Idea



- Build multiple packages
- Versioning of multiple packages

- Build multiple packages
  - Static analysis of code
- Versioning of multiple packages

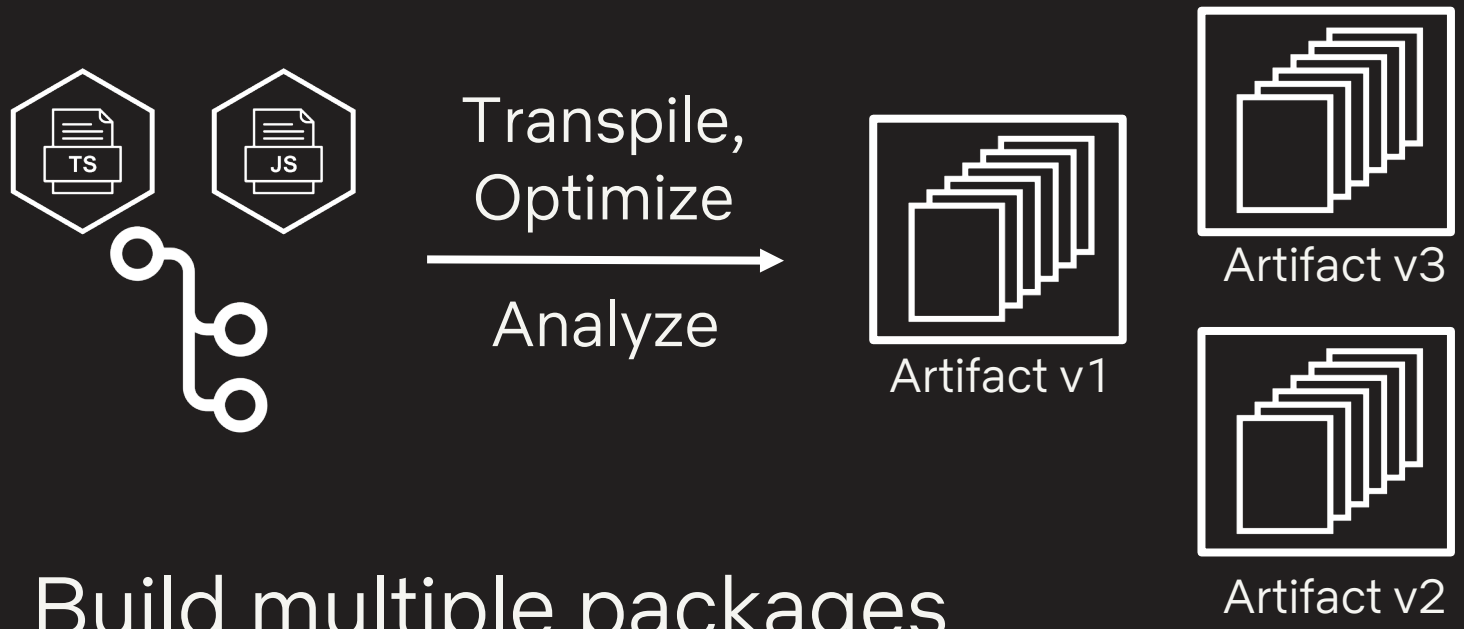
newSearchExp is *false*



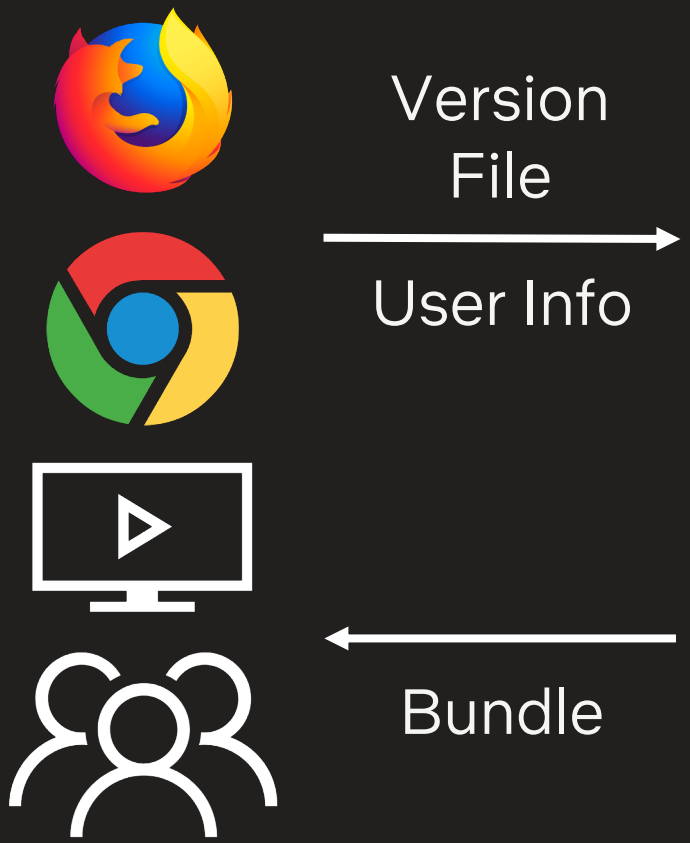
newSearchExp is *true*

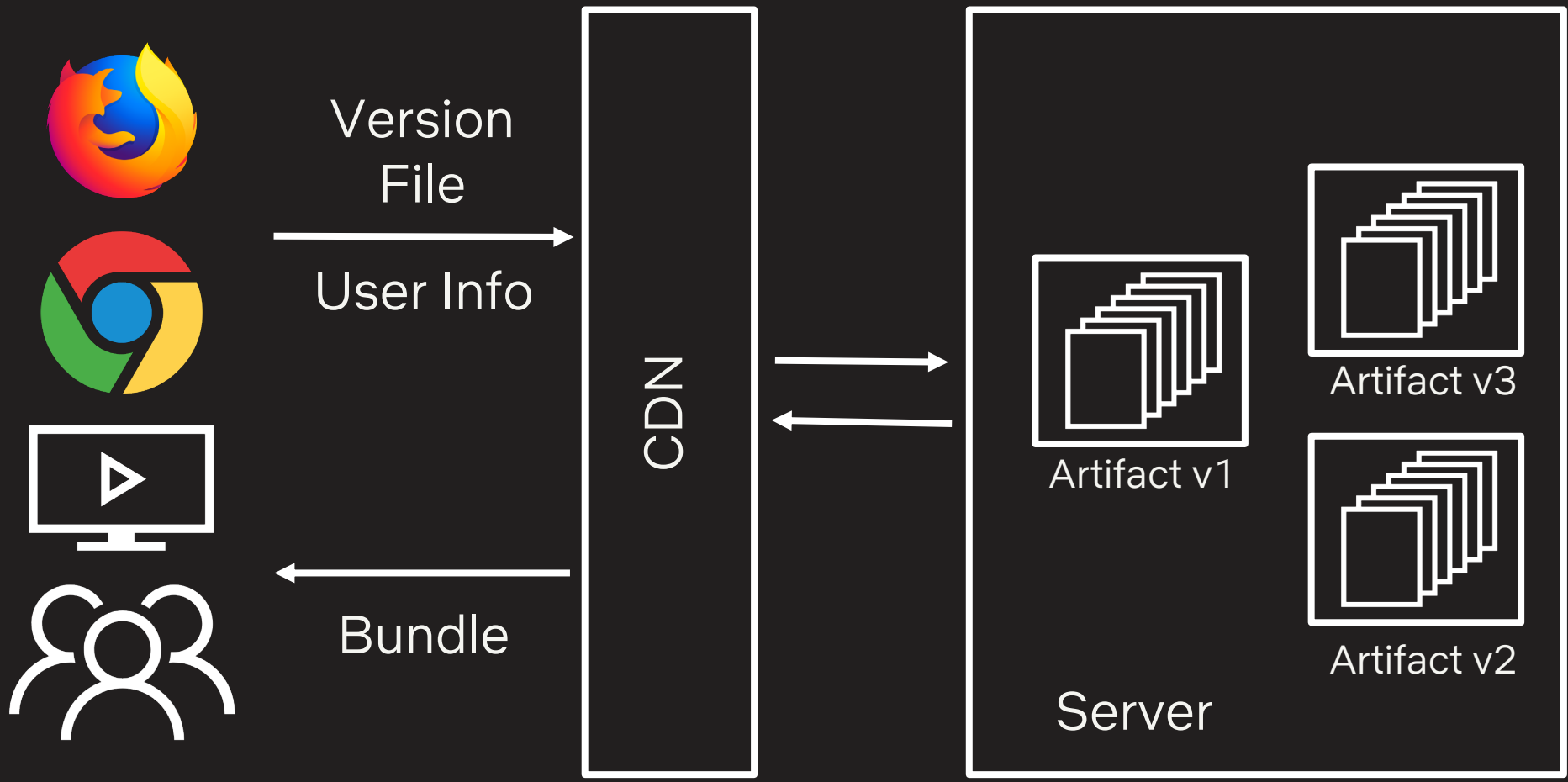
- Build multiple packages
  - Static analysis of code
- Versioning of multiple packages

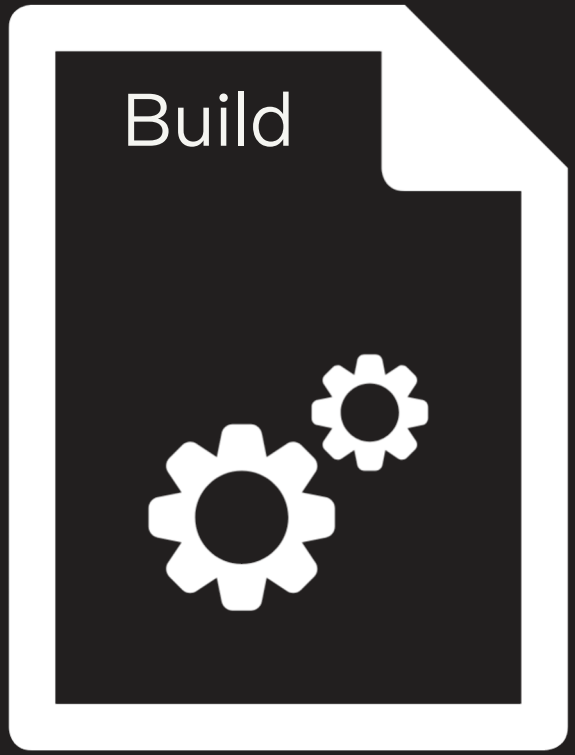
- Build multiple packages
  - Static analysis of code
- Versioning of multiple packages
  - Logical grouping of these packages



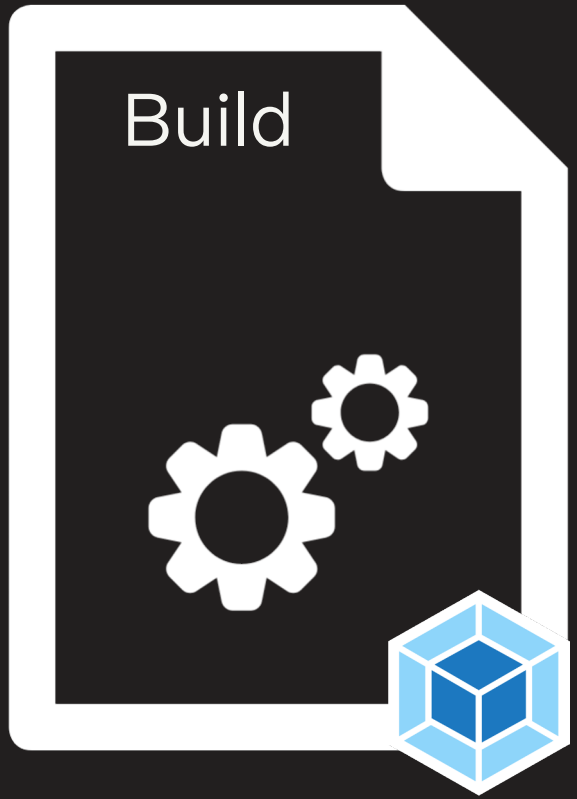
- Build multiple packages
  - Static analysis of code
- Versioning of multiple packages
  - Logical grouping of these packages


Version
File
User Info
Bundle
Artifact v1
Artifact v3
Artifact v2
Server

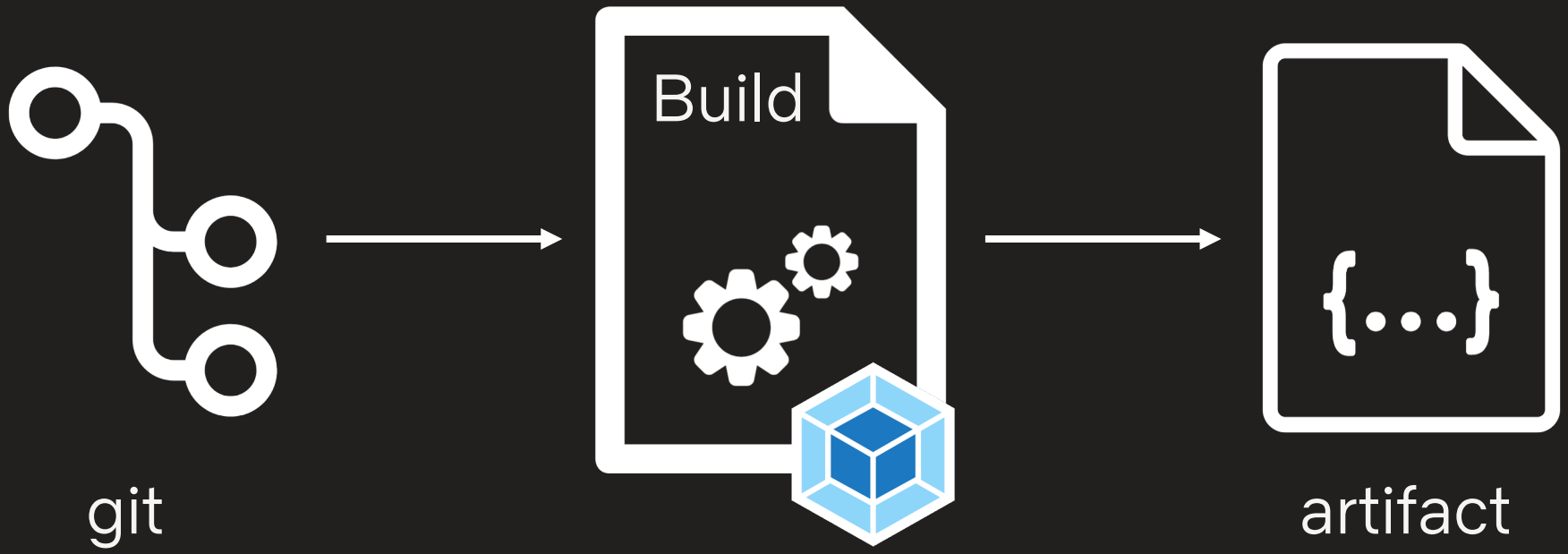

Version
File
User Info
Bundle
CDN
Artifact v1
Artifact v3
Artifact v2
Server

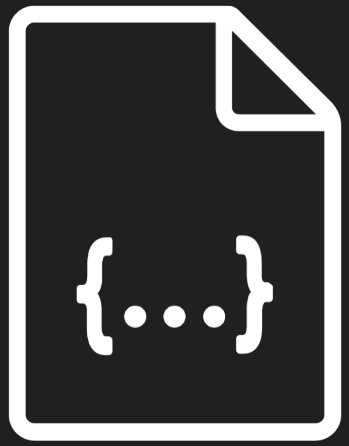

Build


Request


Build

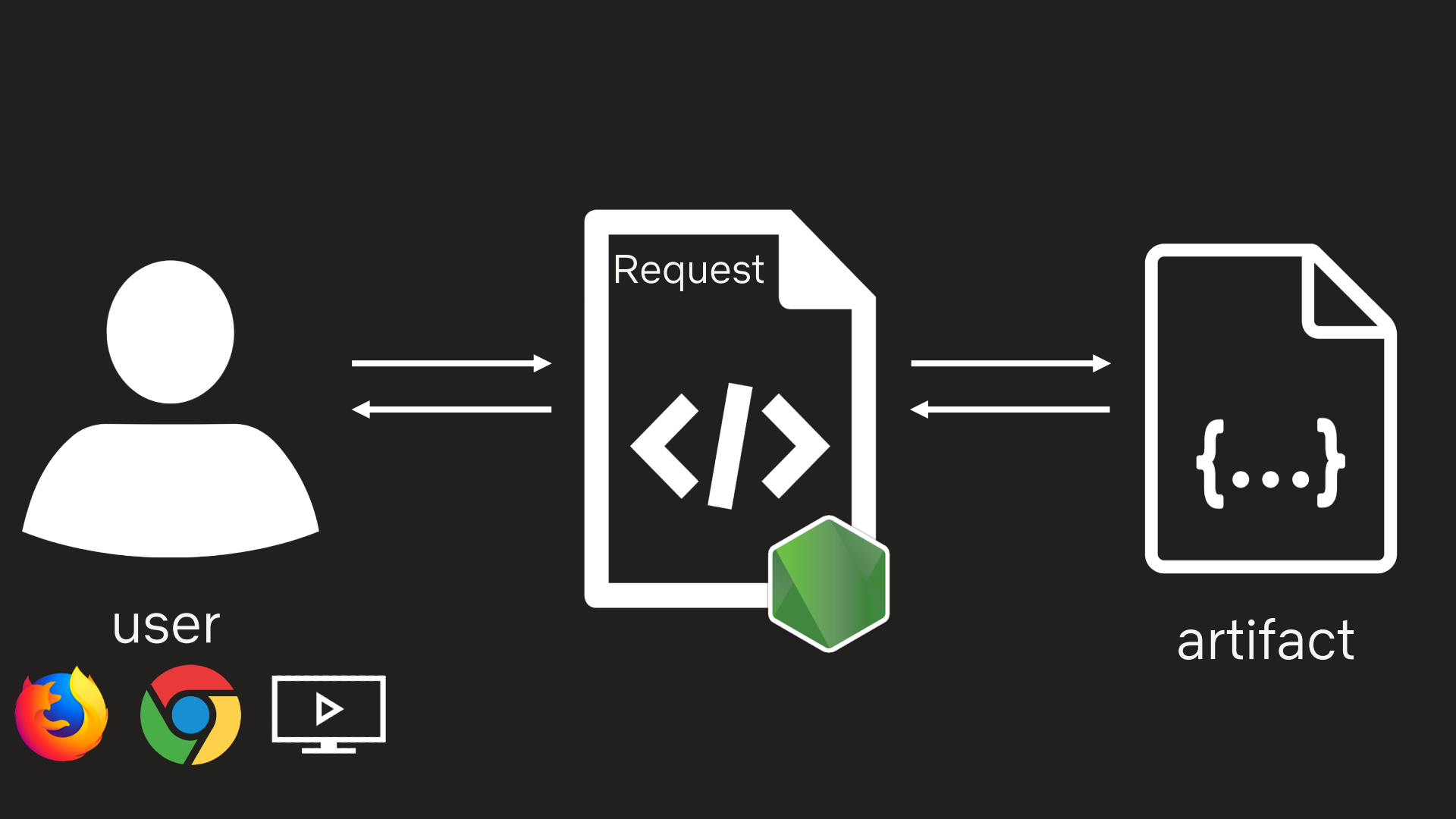

Request


Build
{...}
git
artifact

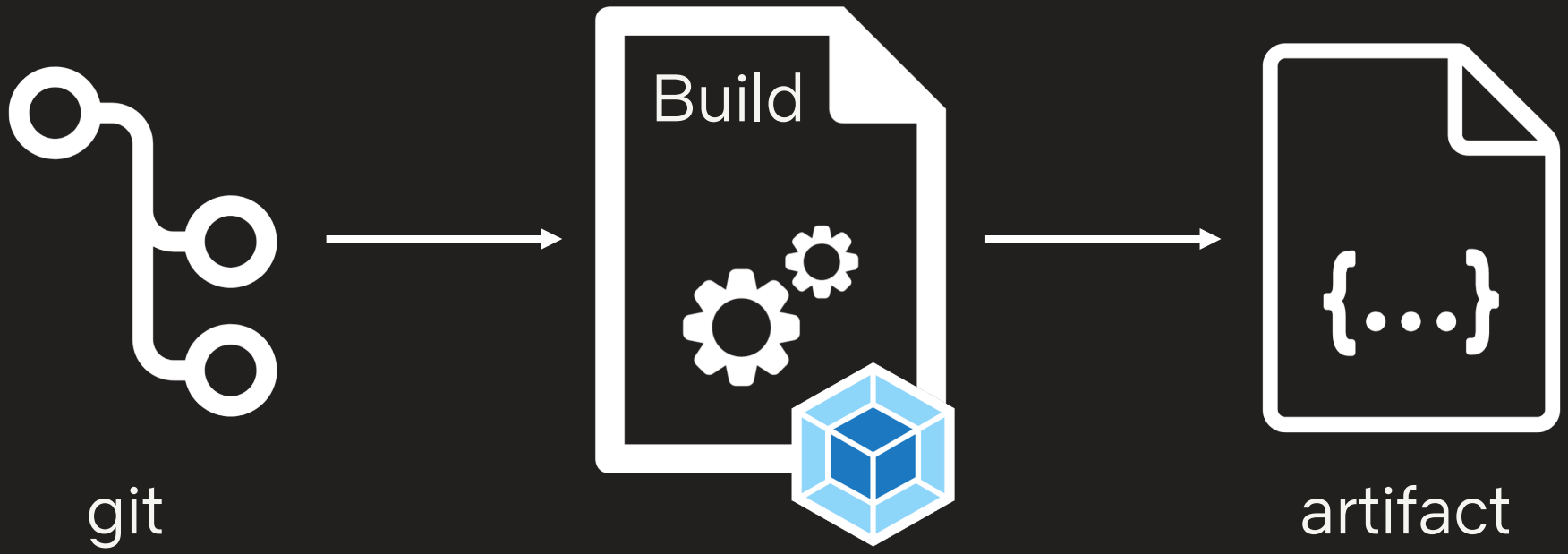


artifact

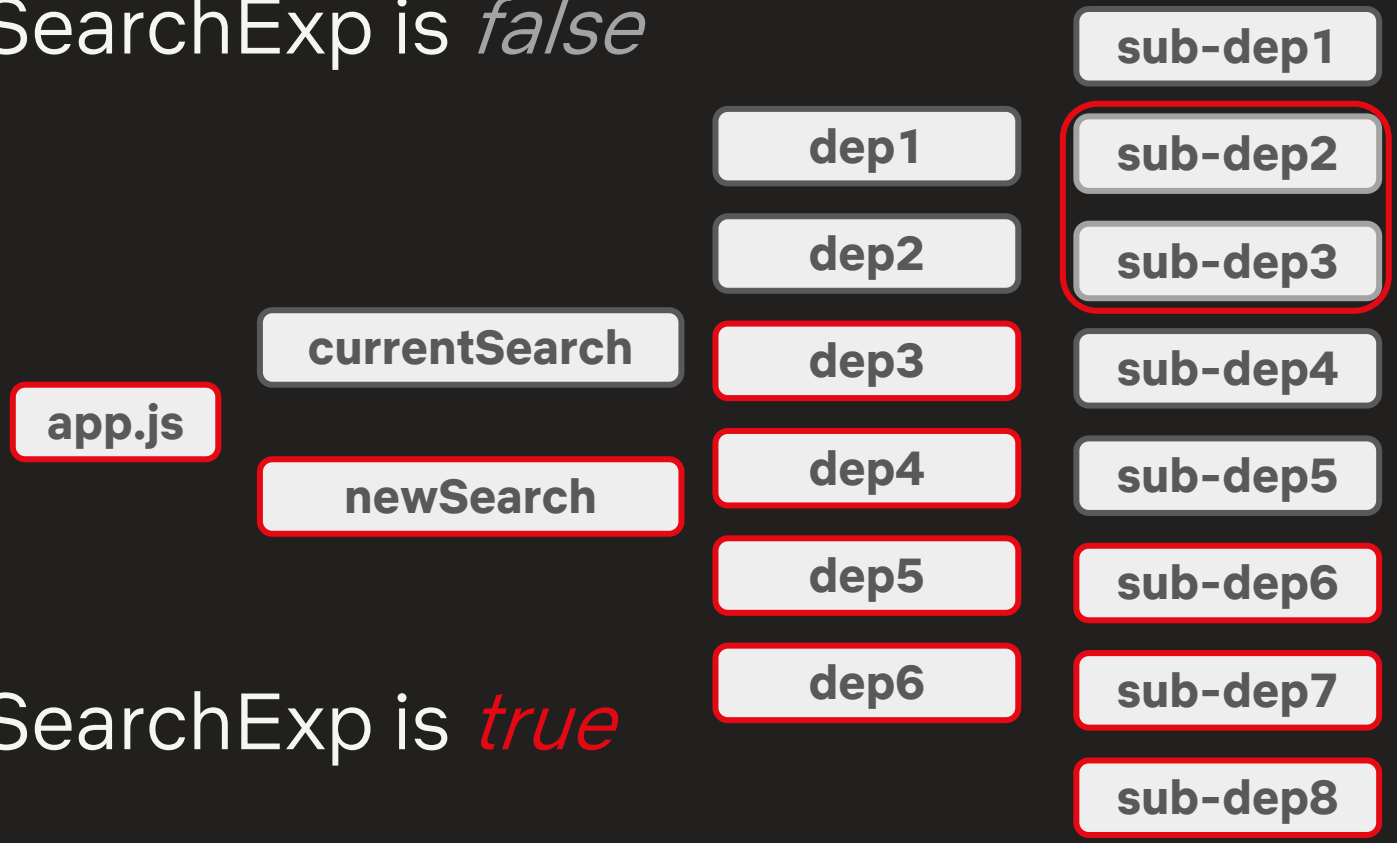
- Dependency Graph
- Metadata to resolve files to serve
- Transpiled and optimized source code for all the files


Request
{...}
user
artifact

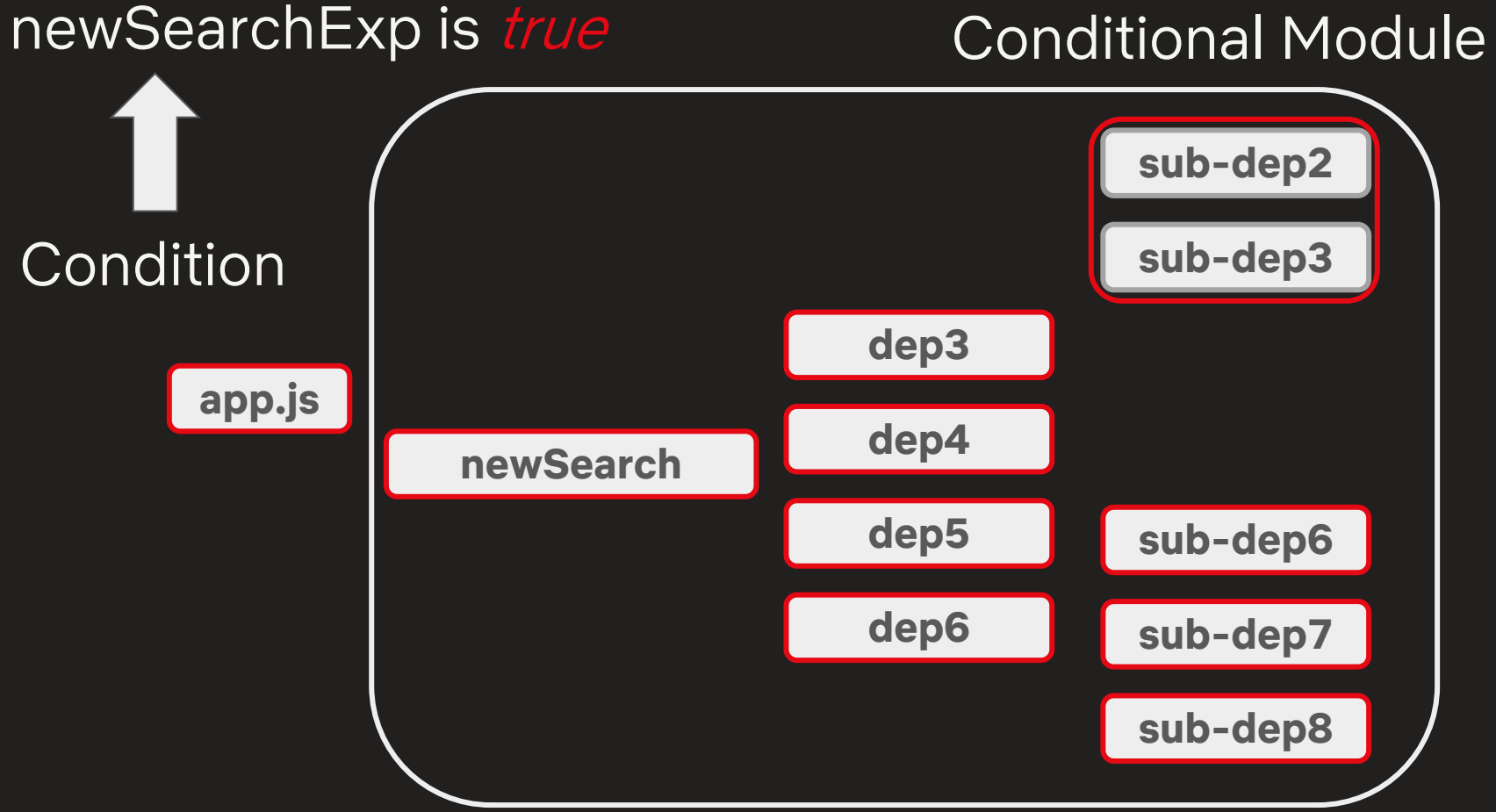
Conditional Modules


Build
{...}
git
artifact

newSearchExp is *false*



newSearchExp is *true*


newSearchExp is *true*
Condition
app.js
Conditional Module
newSearch
dep3
dep4
dep5
dep6
sub-dep2
sub-dep3
sub-dep6
sub-dep7
sub-dep8

How do we mark a module as conditional?

newSearch.js

```
// @condition newSearchExp
import React, { Component } from 'react';
...
// rest of the code
...
export ...
```

Syntax Option 1

app.js

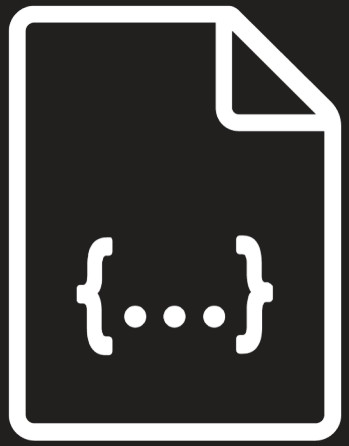
```
...
if ($$conditions$$.newSearchExp === true) {
    search = require('./newSearch');
}
else {
    search = require('./currentSearch');
}
...
export ...
```

Syntax Option 2

With

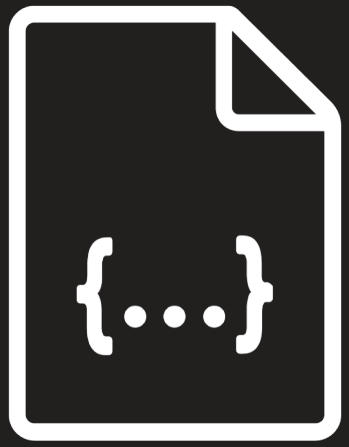
- @condition comment syntax, or,
- $$conditions$$ code syntax

we mark a module as conditional

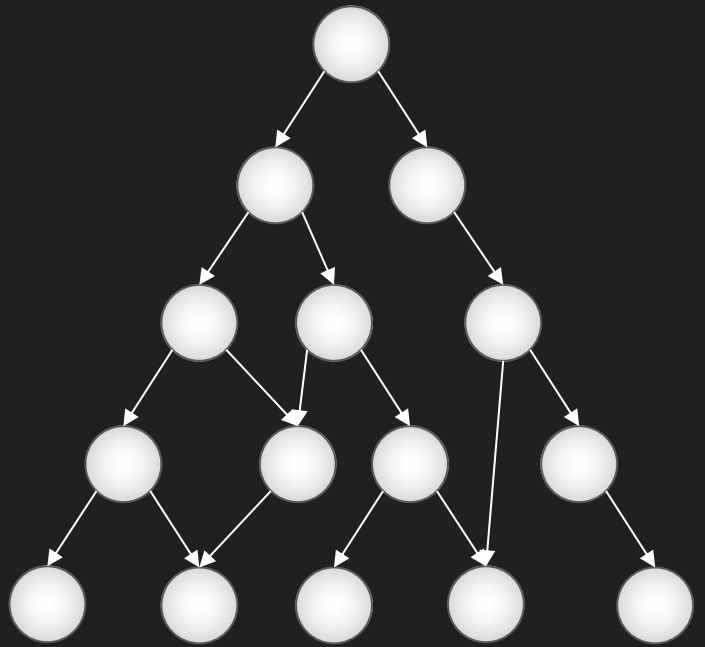


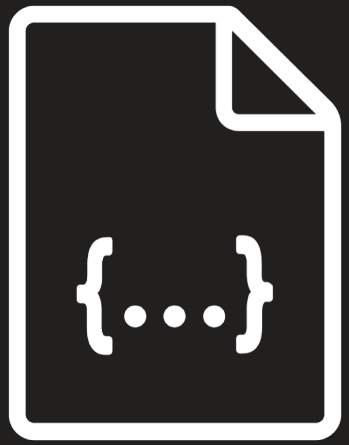
artifact

- Dependency Graph
- **Metadata to resolve files to serve**
- Transpiled and optimized source code for all the files

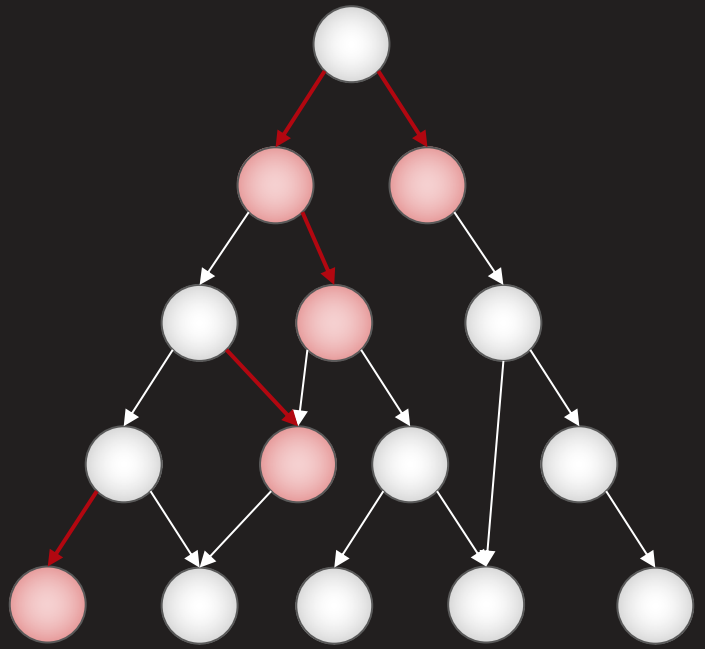


artifact

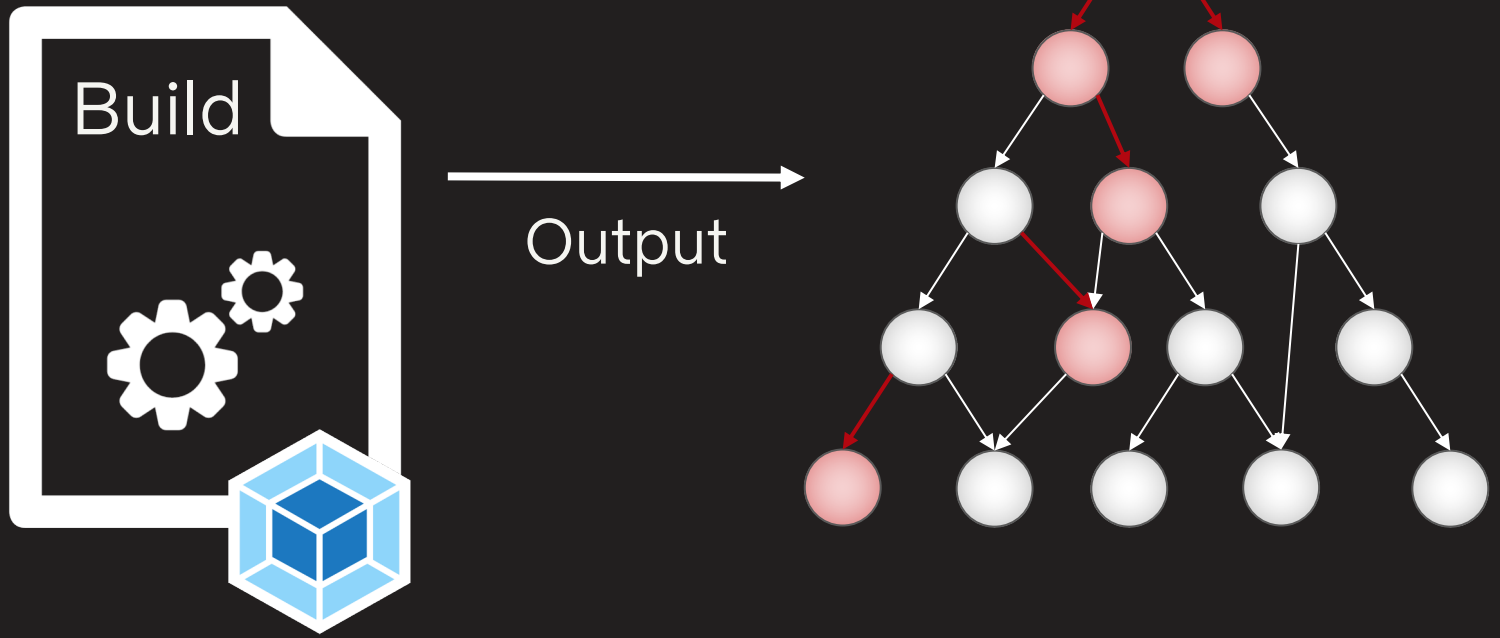




artifact



Webpack



How do we build a
conditional dependency graph
with Webpack?

We built a Webpack plugin and worked with Abstract Syntax Trees (ASTs)

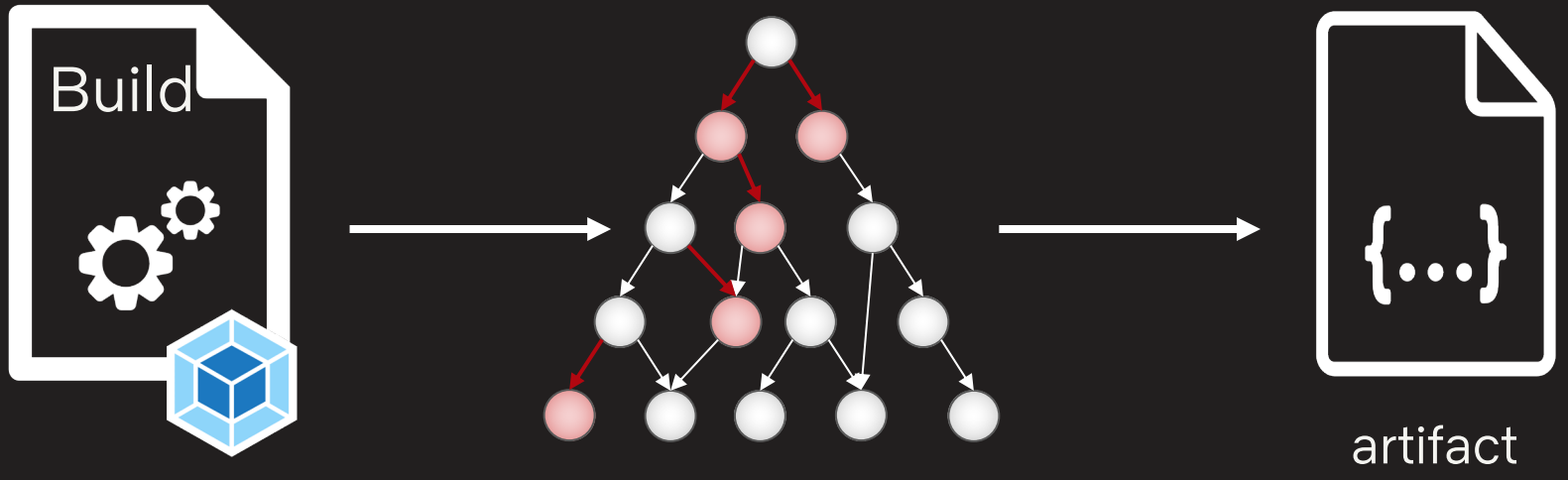
- Hook: parser.hooks.program
- Works for both :
  - @condition syntax
  - $$conditions$$ syntax
- Stores metadata about conditional modules

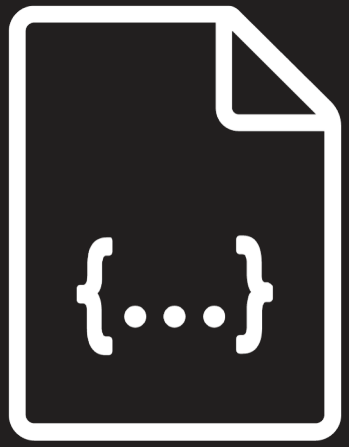
Webpack...

- Already has the full dependency graph
- Already has transpiled and optimized code

Then...

- Hook: compiler.hooks.emit
  - Extract all of data from Webpack
  - Generate conditional dependency graph
  - Serialize it as a JSON file

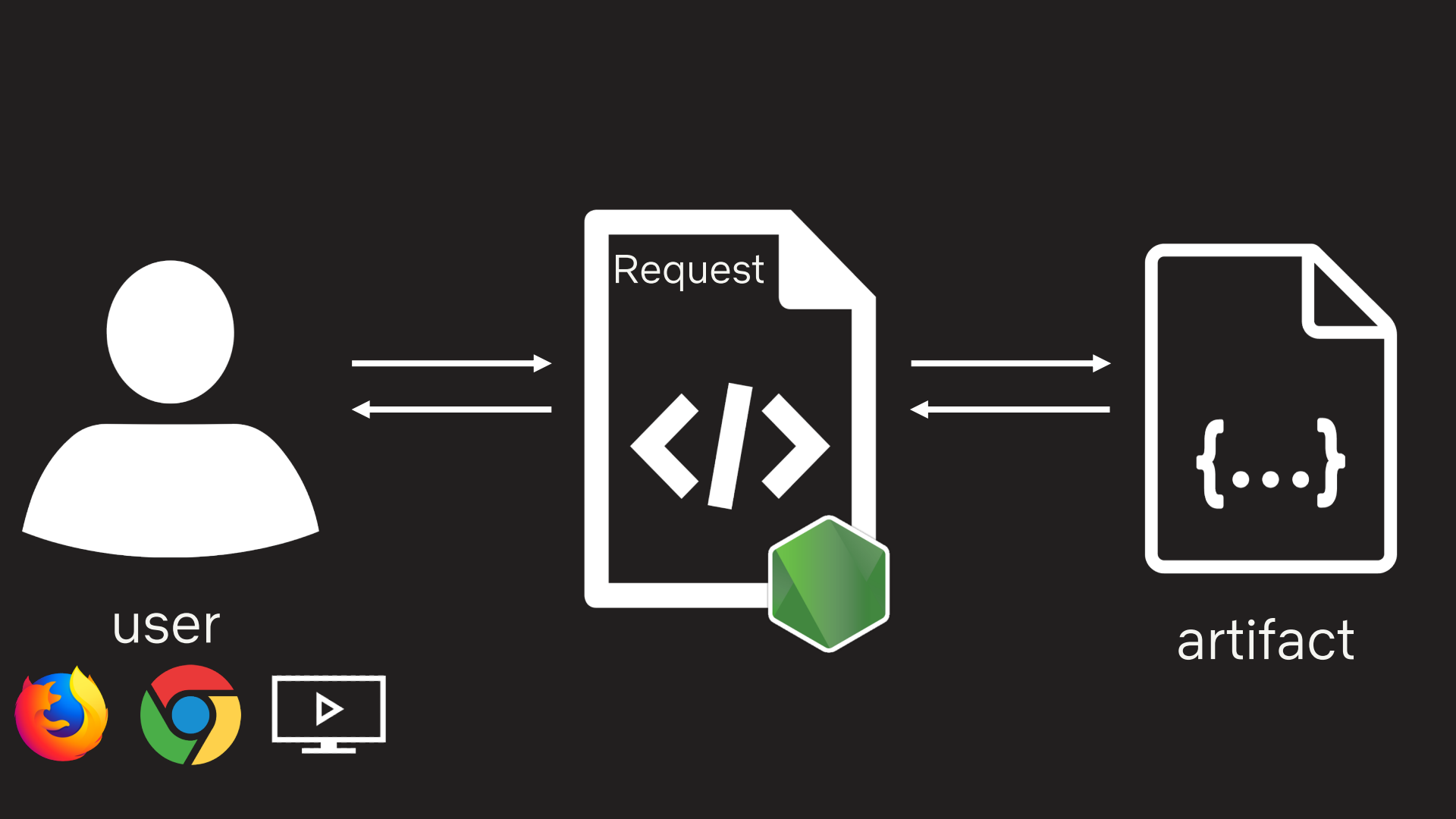

Build
{...}
artifact

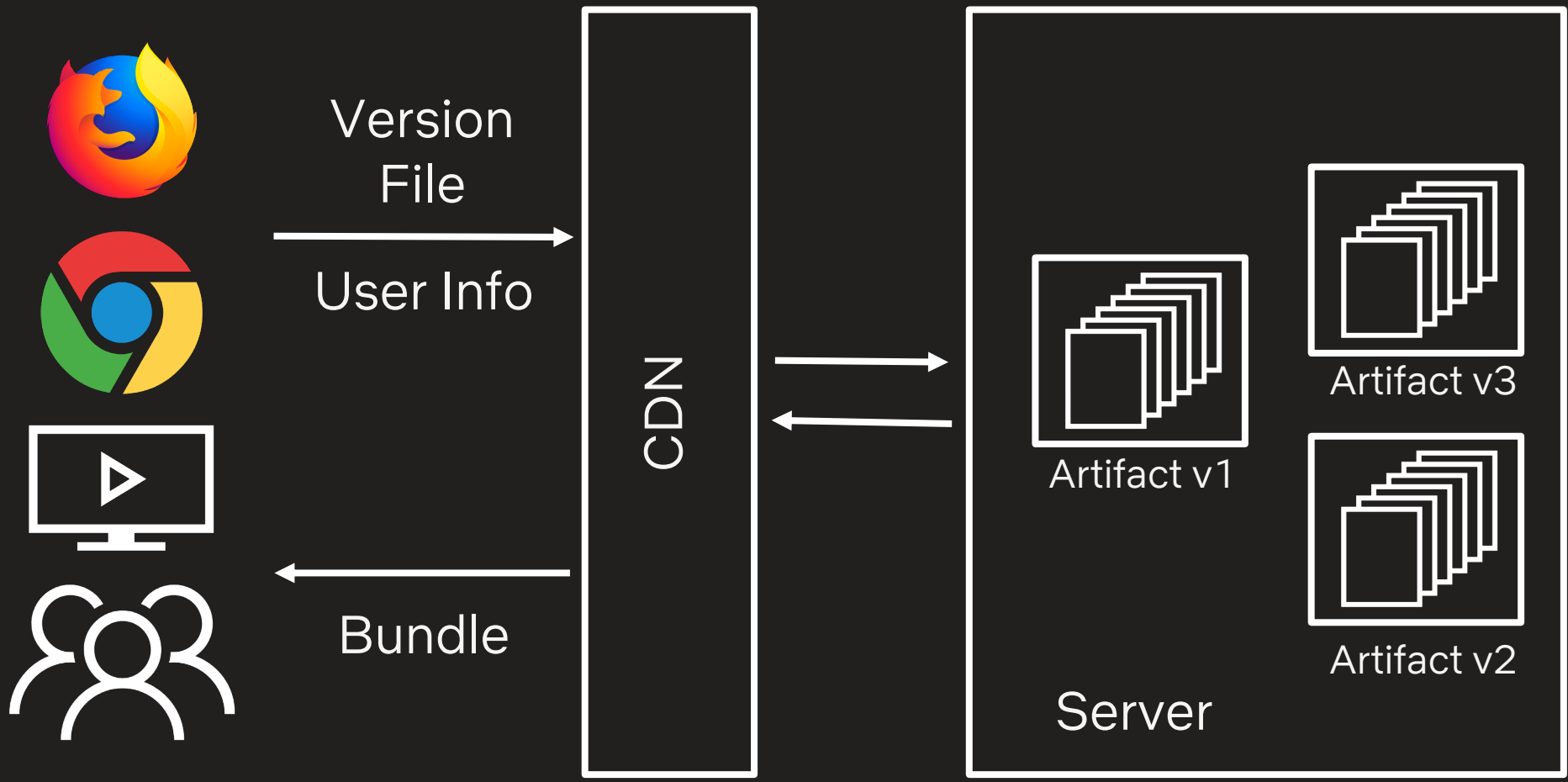


artifact

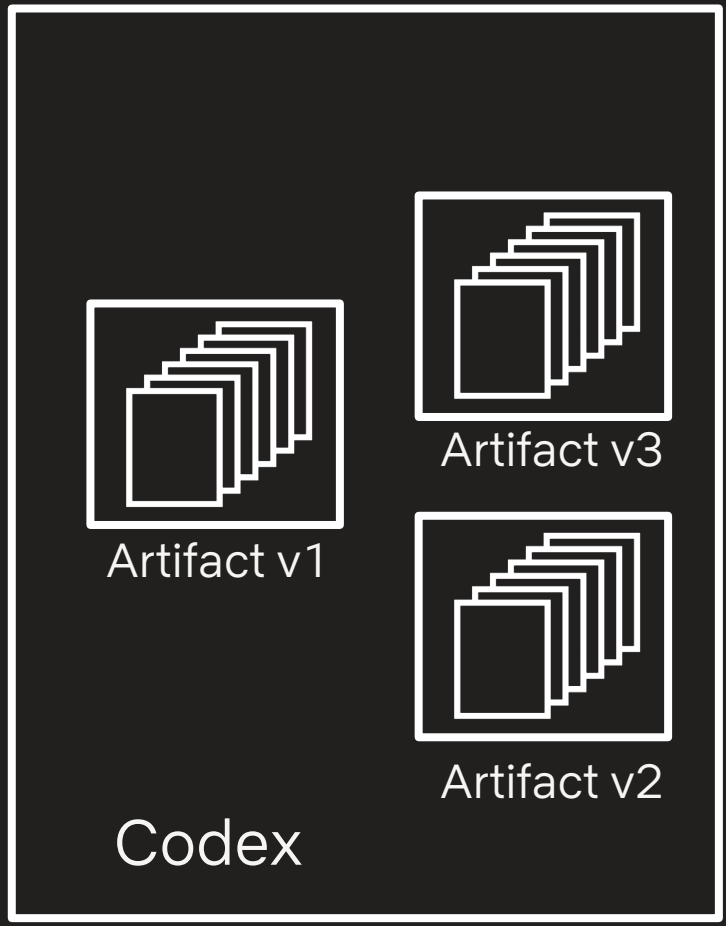
- **Dependency Graph**
- **Metadata to resolve files to serve**
- **Transpiled and optimized source code for all the files**

Dynamic Bundling


Request
{...}
user
artifact


Version
File
User Info
Bundle
CDN
Artifact v1
Artifact v3
Artifact v2
Server

Codex



How do we resolve the
conditional dependencies
at runtime?

To resolve conditional dependencies at runtime, we need

- Request with information, from client
- Versioned artifact, on server

Request Information in URL

https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**

https://codex.nflx.com/**webui**/v3/app.js/newSearchExp,smartPlayback

Request Information in URL

https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**

https://codex.nflx.com/webui/**v3**/app.js/newSearchExp,smartPlayback

Request Information in URL

https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**

https://codex.nflx.com/webui/v3/**app.js**/newSearchExp,smartPlayback

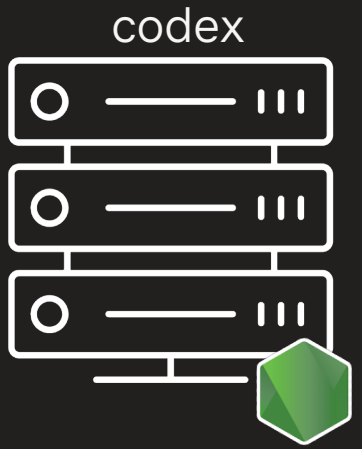
Request Information in URL

https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**

https://codex.nflx.com/webui/v3/app.js/**newSearchExp,smartPlayback**

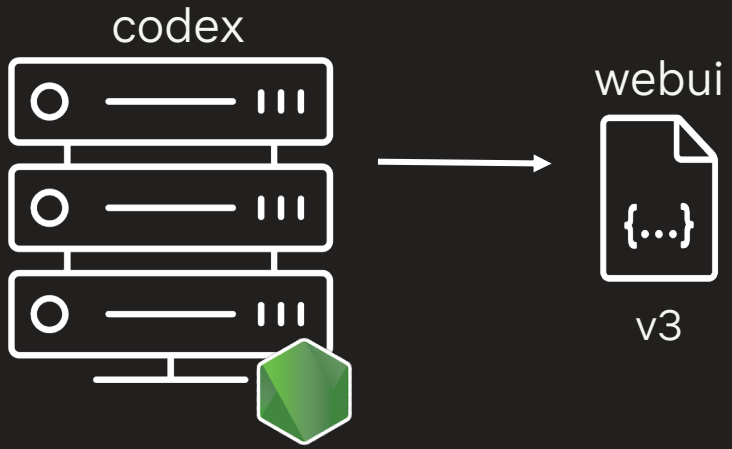
https://codex.nflx.com/webui/v3/app.js/newSearchExp,smartPlayback

https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**



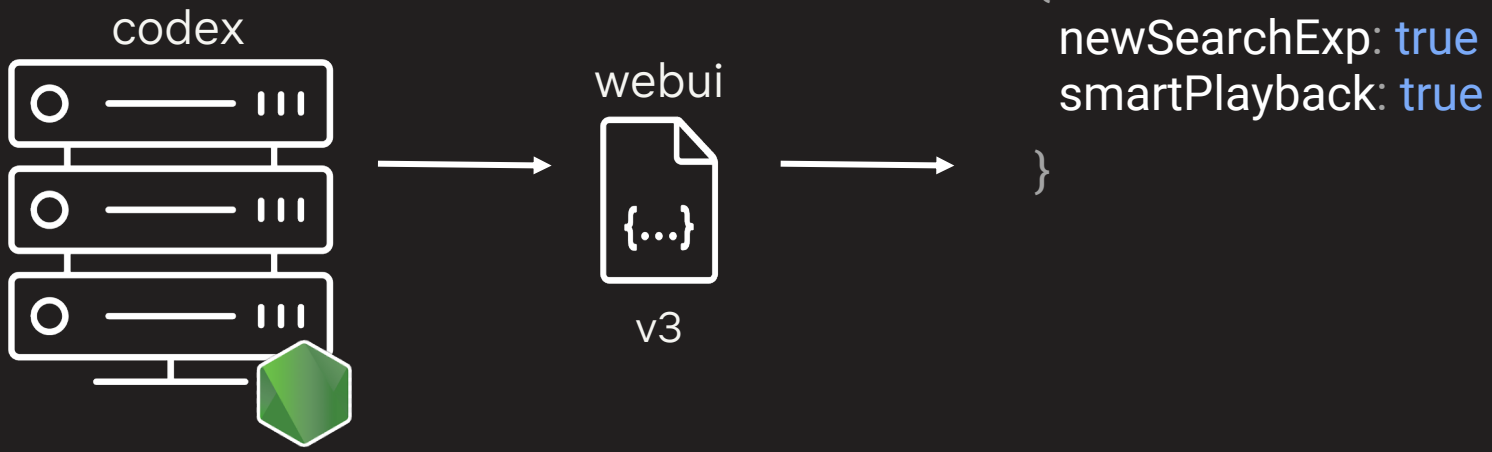
https://codex.nflx.com/**webui**/**v3**/app.js/newSearchExp,smartPlayback

https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**



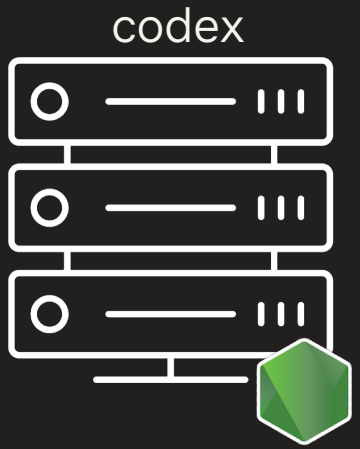
https://codex.nflx.com/webui/v3/app.js/**newSearchExp,smartPlayback**

https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**

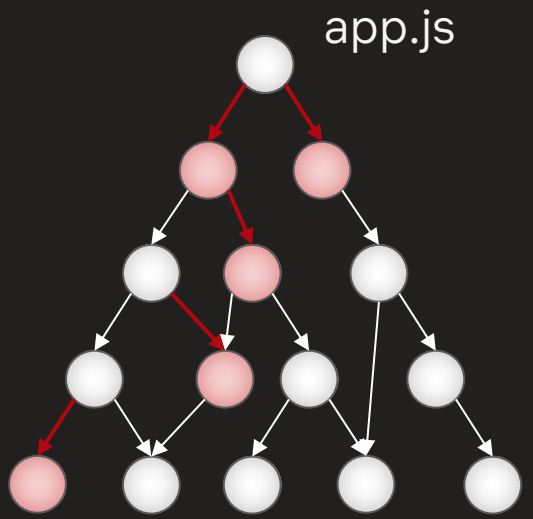


https://codex.nflx.com/webui/v3/**app.js**/newSearchExp,smartPlayback

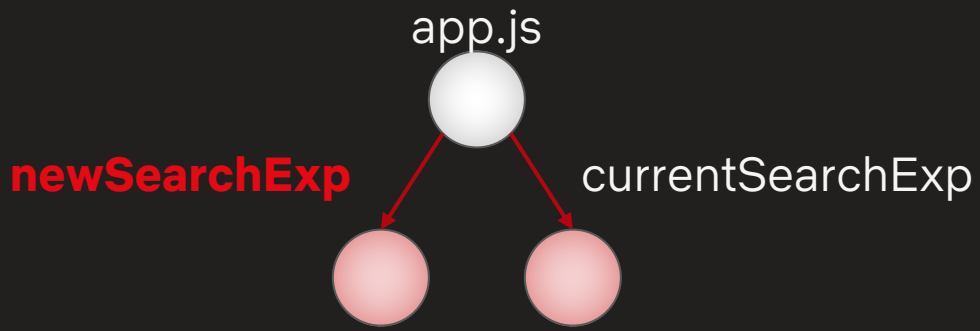
https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**



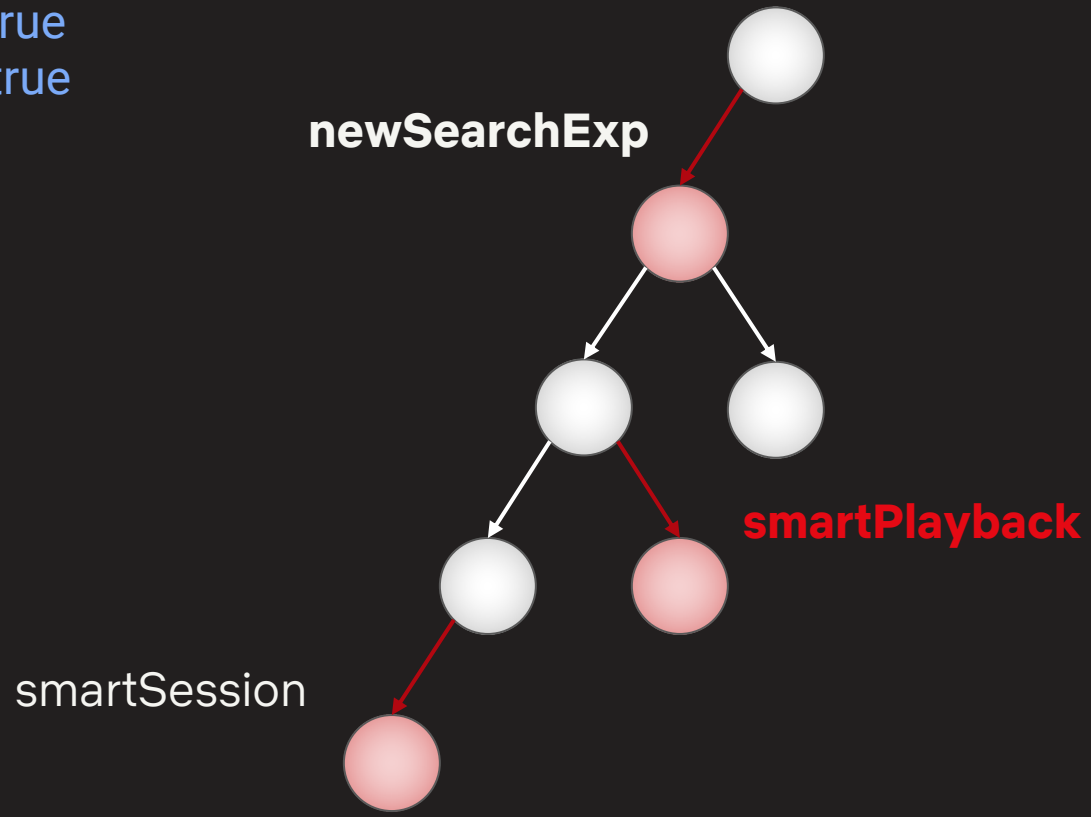

```
{
  newSearchExp: true
  smartPlayback: true
}
```



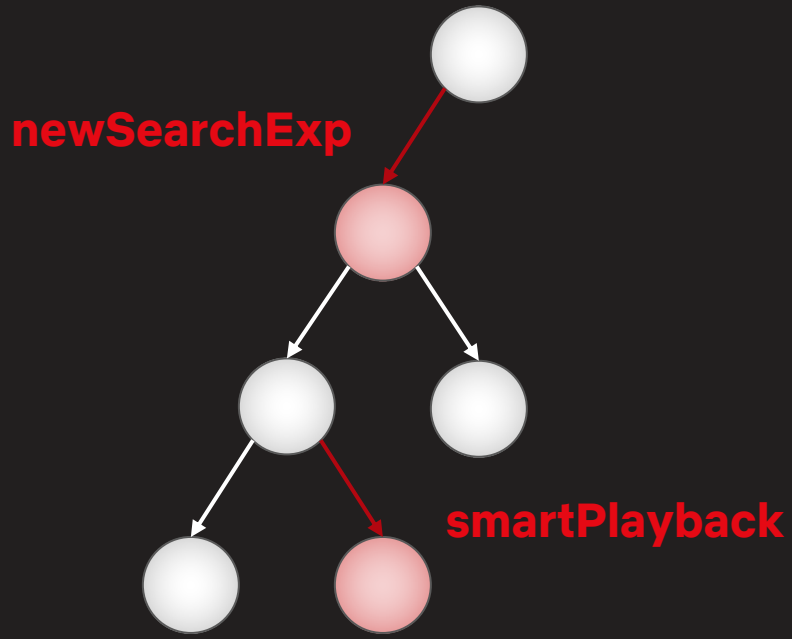
```
{
  newSearchExp: true
  smartPlayback: true
}
```

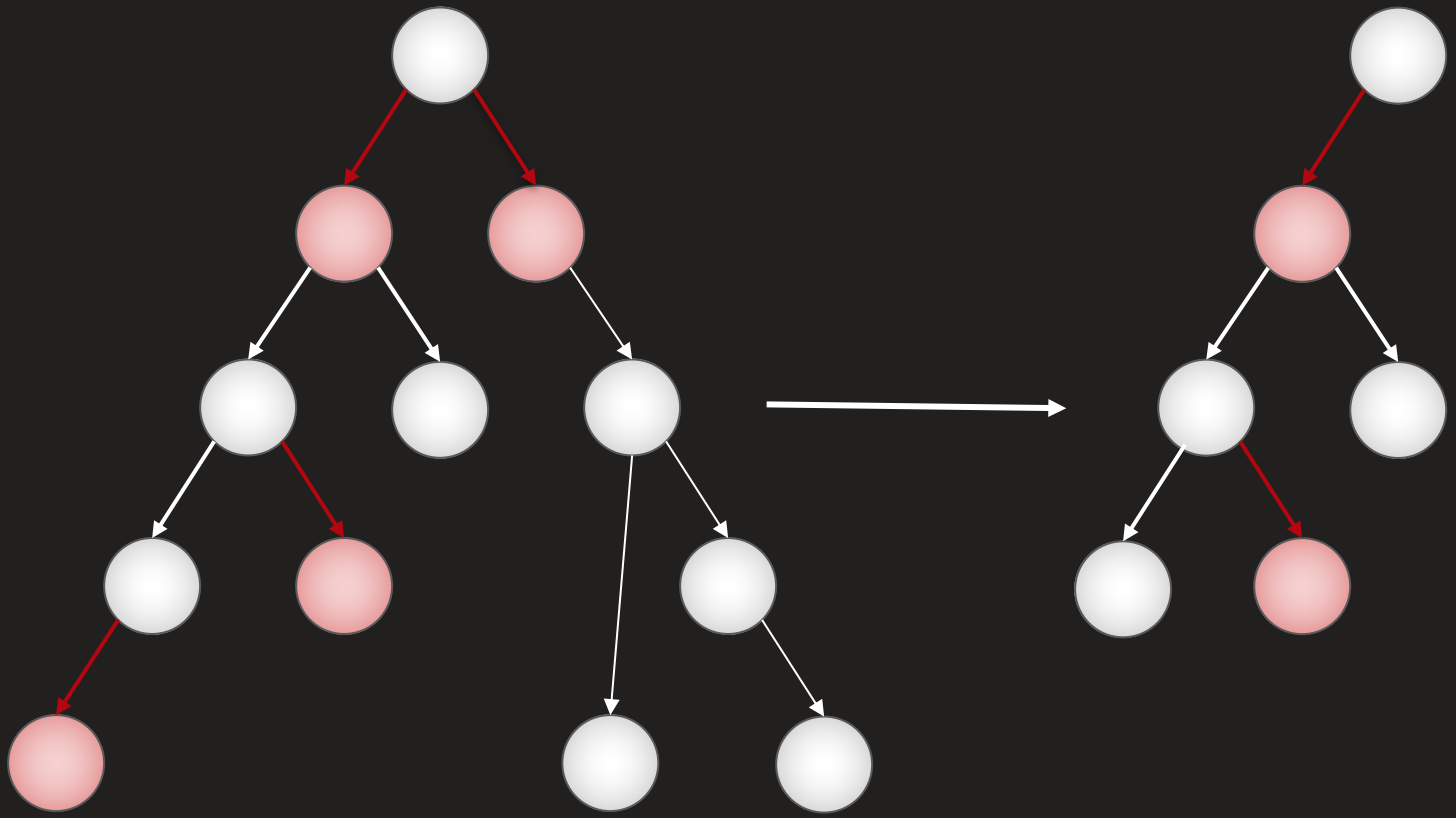


```
{
  newSearchExp: true
  smartPlayback: true
}
```

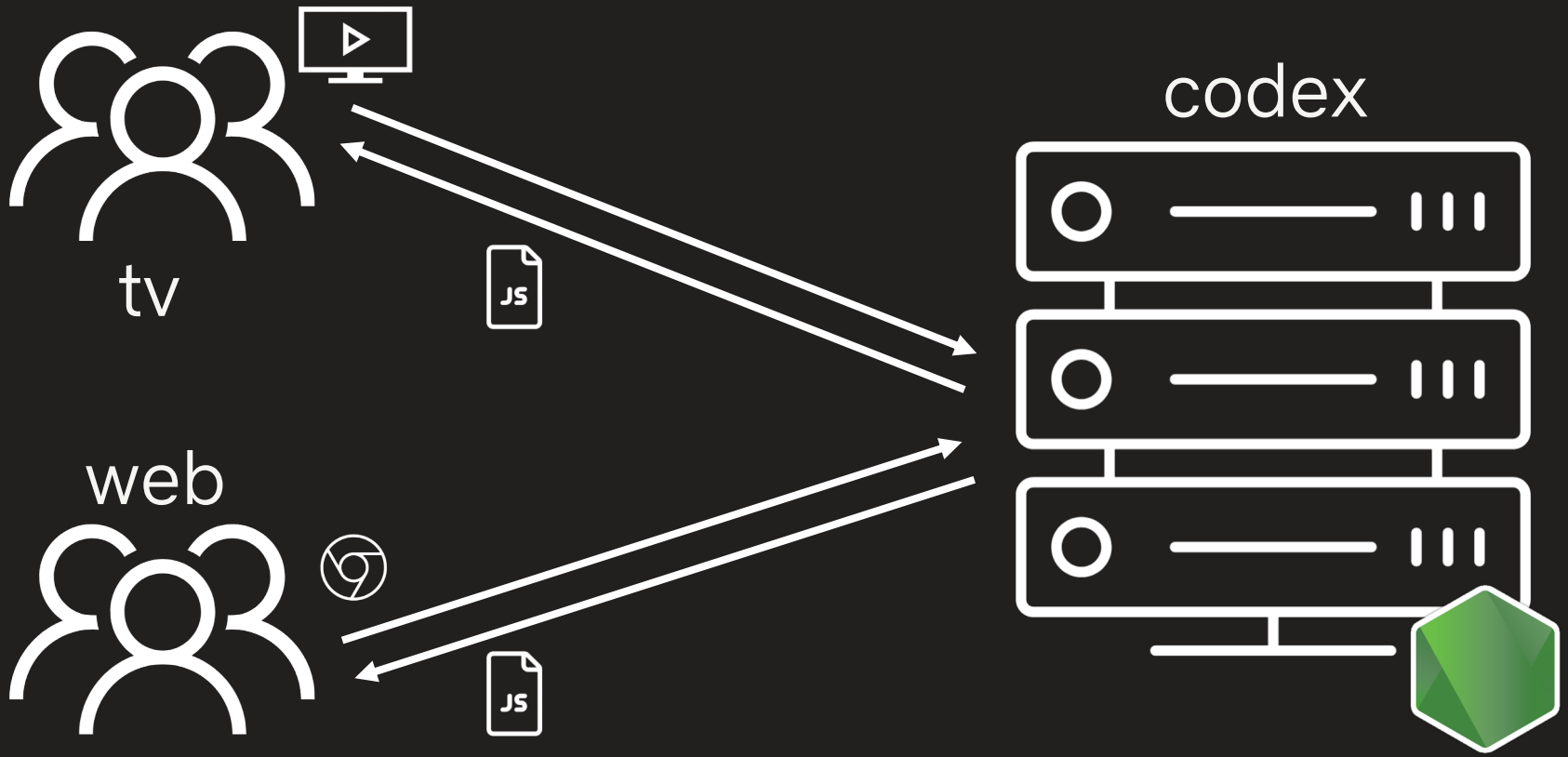


```
{
  newSearchExp: true
  smartPlayback: true
}
```





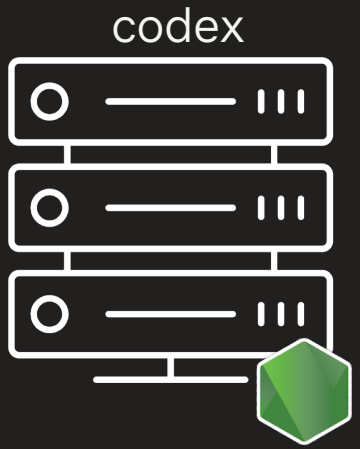
https://codex.nflx.com/**tvui**/v3/app.js/**newSearchExp**,smartSession

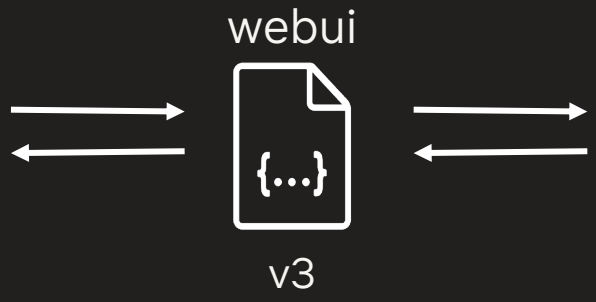


https://codex.nflx.com/**webui**/v3/app.js/**newSearchExp**,smartPlayback

https://codex.nflx.com/webui/v3/app.js/newSearchExp,smartPlayback

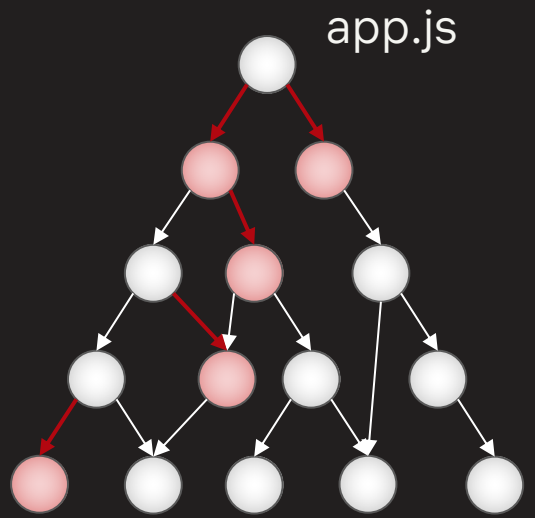
https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**





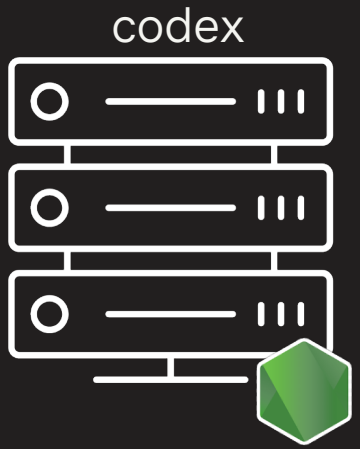

```
{
  newSearchExp: true
  smartPlayback: true
}
```

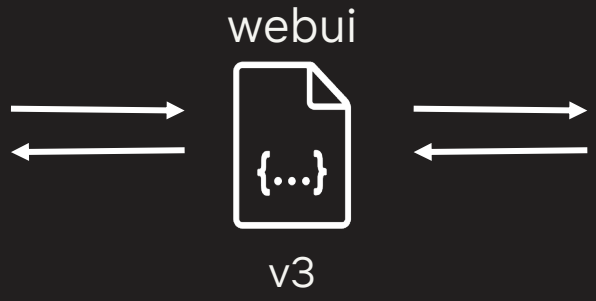
Slow?



https://codex.nflx.com/webui/v3/app.js/newSearchExp,smartPlayback

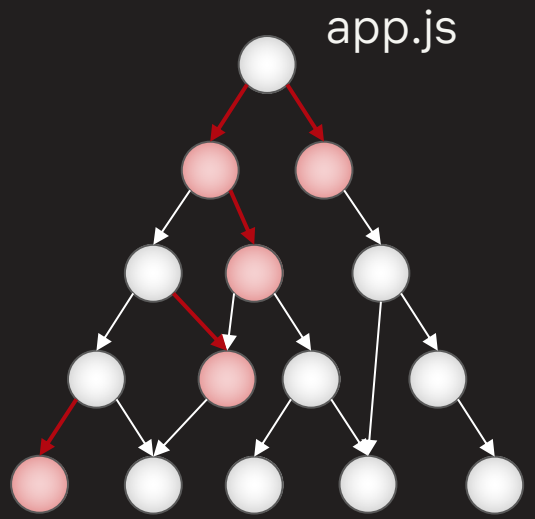
https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**





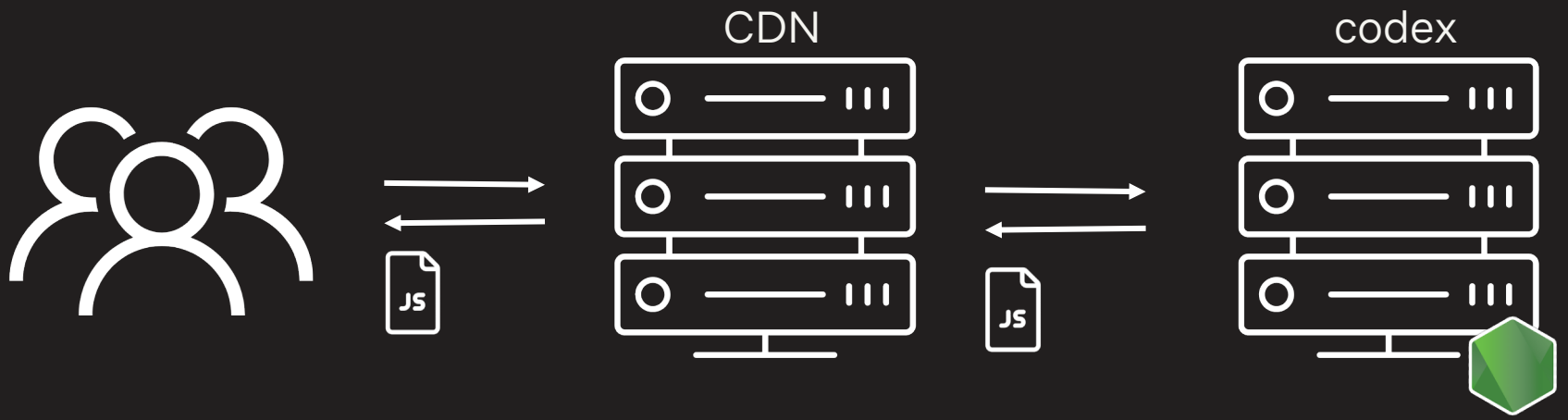

```
{
  newSearchExp: true
  smartPlayback: true
}
```

response time < 70ms



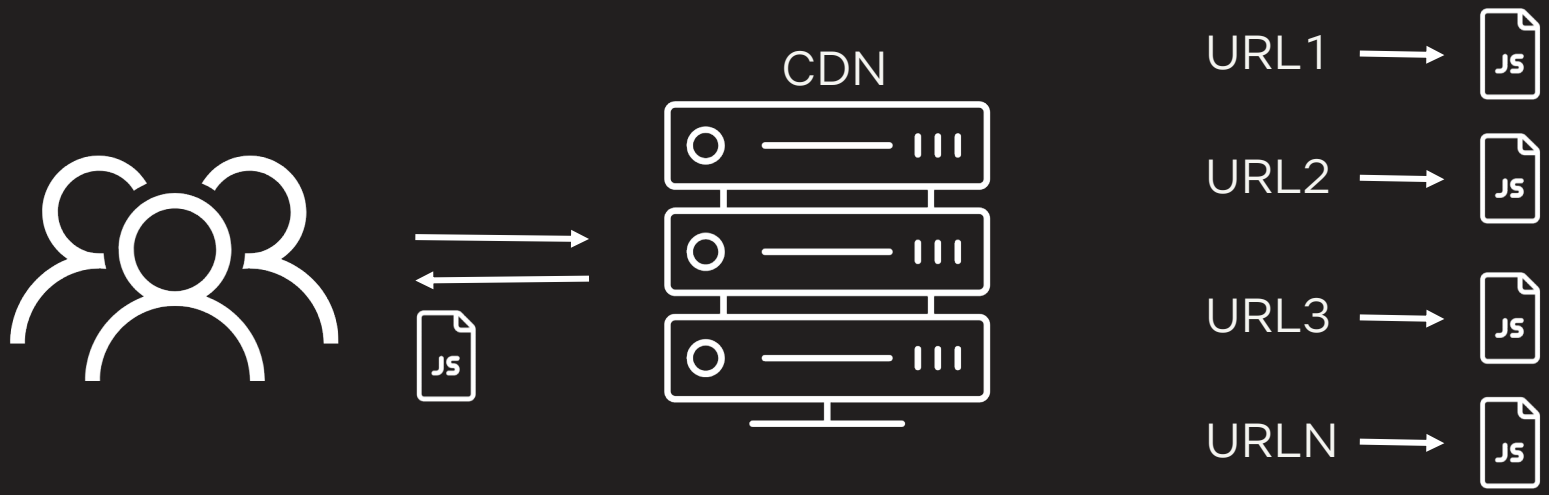
https://codex.nflx.com/webui/v3/app.js/newSearchExp,smartPlayback

https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**



https://codex.nflx.com/webui/v3/app.js/newSearchExp,smartPlayback

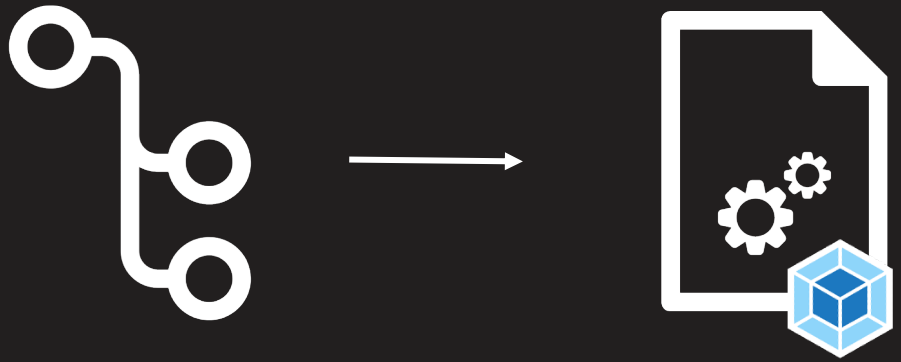
https://codex.nflx.com/ **{team}** / **{version}** / **{entrypoint}** / **{conditions}**

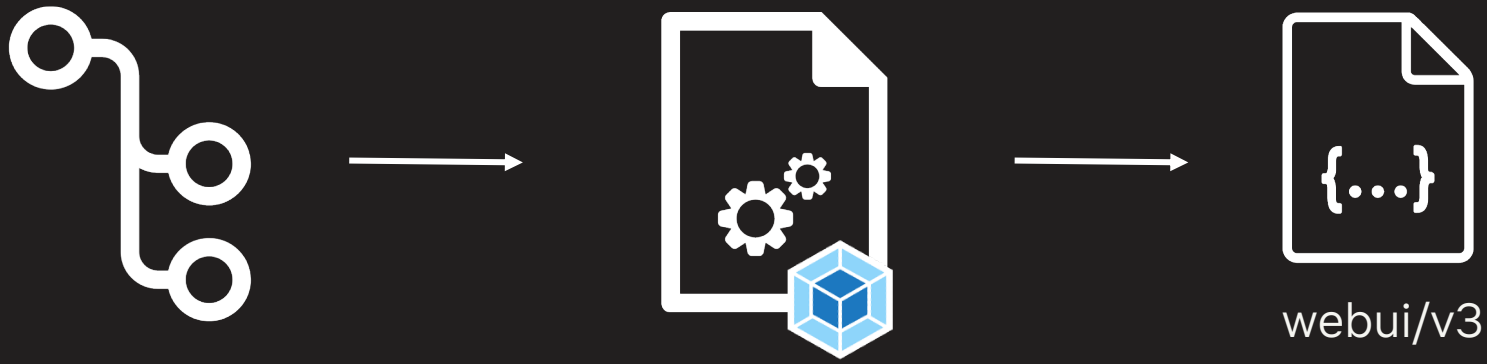


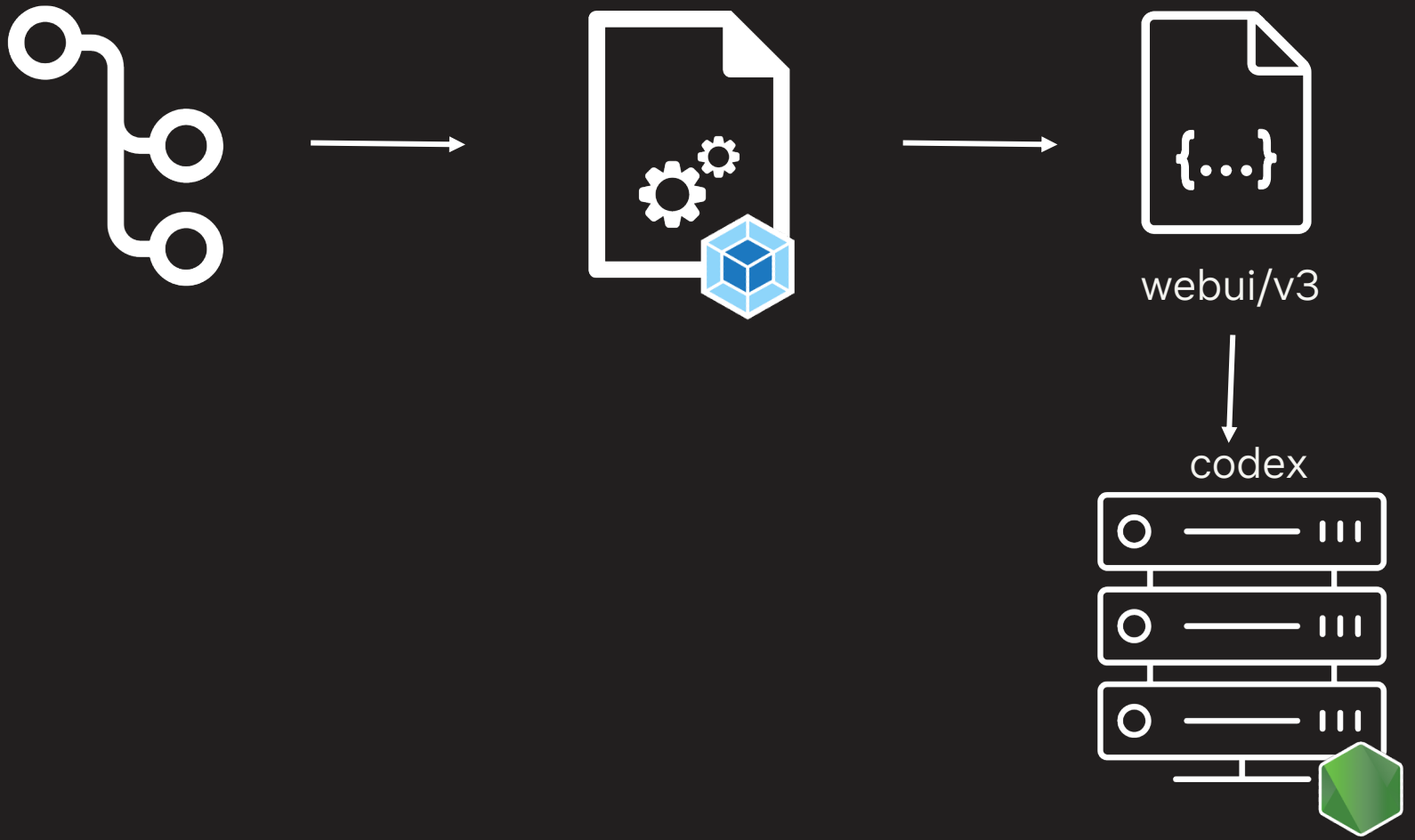
Summary

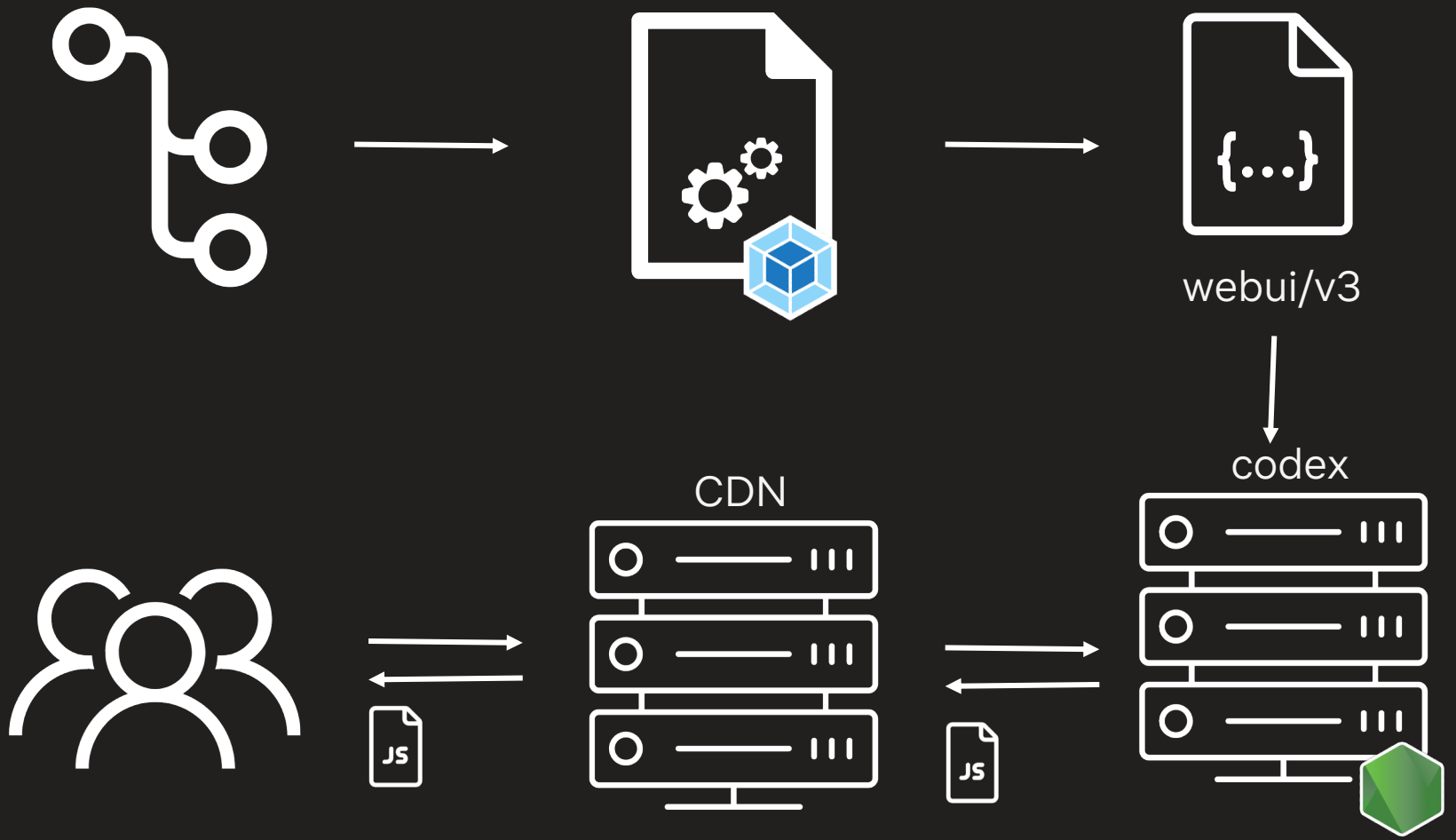
```
...
if ($$conditions$$.newSearchExp === true) {
    search = require('./newSearch');
}
else {
    search = require('./currentSearch');
}
// rest of the code
```

```
// @condition newSearchExp
import React, { Component } from 'react';
...
// rest of the code
```




{...}
webui/v3


{...}
webui/v3
codex


{...}
webui/v3
codex
CDN
JS
JS

The Future

Developer experience and Webpack

Managing the lifecycle of the artifacts

Thank you

Rajat Kumar
@rajatkumar