



low hanging fruit

vs.



micro-optimization

Creative Techniques for Loading Web Pages Faster

Trevor Parscal - @trevorparscal
Roan Kattouw - @catrope



OSCON 2011

WIKIMEDIA 

- We work for Wikimedia Foundation
- We created a JS/CSS loading system for Wikipedia
- We want to share what we did, and what we learned
- Questions throughout or at the end are OK

People



OSCON 2011



- Roan: Robot
- Trevor: Human

People



Roan

- Roan: Robot
- Trevor: Human

People



Roan

robot

- Roan: Robot
- Trevor: Human

People



Roan

robot



Trevor

- Roan: Robot
- Trevor: Human

People



Roan

robot



Trevor
human

- Roan: Robot
- Trevor: Human

People



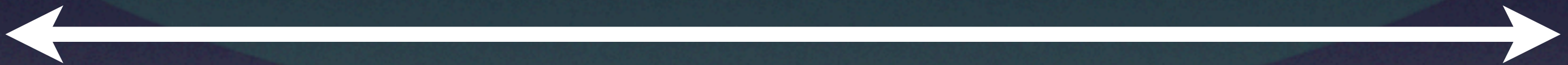
- Lots of crossover
- Understand the problem of client-rich development
- Know how to make things scale well

People



Back-end

Front-end



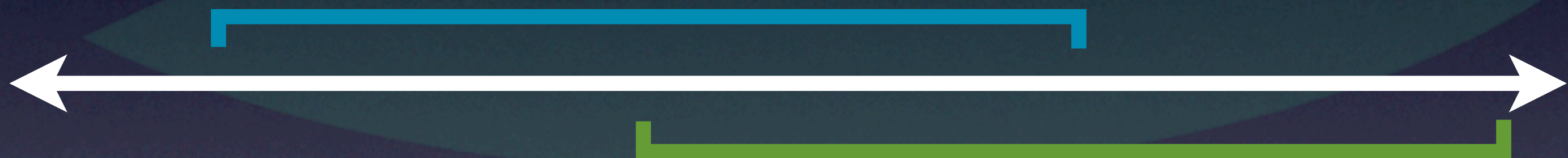
- Lots of crossover
- Understand the problem of client-rich development
- Know how to make things scale well

People



Back-end

Front-end



- Lots of crossover
- Understand the problem of client-rich development
- Know how to make things scale well

Software

WIKIMEDIA 

- MediaWiki: wiki application that powers Wikimedia's sites
- Open source!
- Linux, Apache, MySQL, PHP, Memcached, Squid, Varnish
- Wikimedia sites are the 5th most visited sites in the world
- The largest Wikimedia site is Wikipedia

Software



MediaWiki
platform

- MediaWiki: wiki application that powers Wikimedia's sites
- Open source!
- Linux, Apache, MySQL, PHP, Memcached, Squid, Varnish
- Wikimedia sites are the 5th most visited sites in the world
- The largest Wikimedia site is Wikipedia

Software



MediaWiki

platform



Wikipedia

project

- MediaWiki: wiki application that powers Wikimedia's sites
- Open source!
- Linux, Apache, MySQL, PHP, Memcached, Squid, Varnish
- Wikimedia sites are the 5th most visited sites in the world
- The largest Wikimedia site is Wikipedia

Trevor Parscal My talk My preferences My watchlist My contributions Log out

Main Page Discussion Read Edit View history Search

Welcome to Wikipedia,
the free encyclopedia that anyone can edit.
3,694,628 articles in English

- Arts
- History
- Society
- Biography
- Mathematics
- Technology
- Geography
- Science
- All portals

Today's featured article

Jack Warner (1892–1978) was a Canadian-born American film executive who was the president and driving force behind the Warner Bros. Studios in Hollywood, Los Angeles, California. Warner's 45-year career was longer than that of any other traditional Hollywood studio mogul. He worked with his brother, Sam Warner, to procure the technology for the film industry's first talking picture. Although Warner was feared by many of his employees and inspired ridicule with his uneven attempts at humor, he earned respect for his shrewd instincts and toughmindedness. He recruited many of Warner Bros.' top stars and promoted the hard-edged social dramas for which the studio became known. Although he was a

In the news

- Asteroid **2010 TK7** is confirmed as the first Earth trojan asteroid discovered.
- Truong Tan Sang** becomes the new President of Vietnam and nominates Nguyen Tan Dung to another term as Prime Minister.
- In cycling, **Cadel Evans** (pictured) wins the **2011 Tour de France**, becoming the first Australian Tour de France winner.
- Protests** against rising house prices in Israel continue, with thousands gathering in Tel Aviv and Jerusalem.
- Thousands of protesters encounter

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact Wikipedia

Toolbox

Print/export

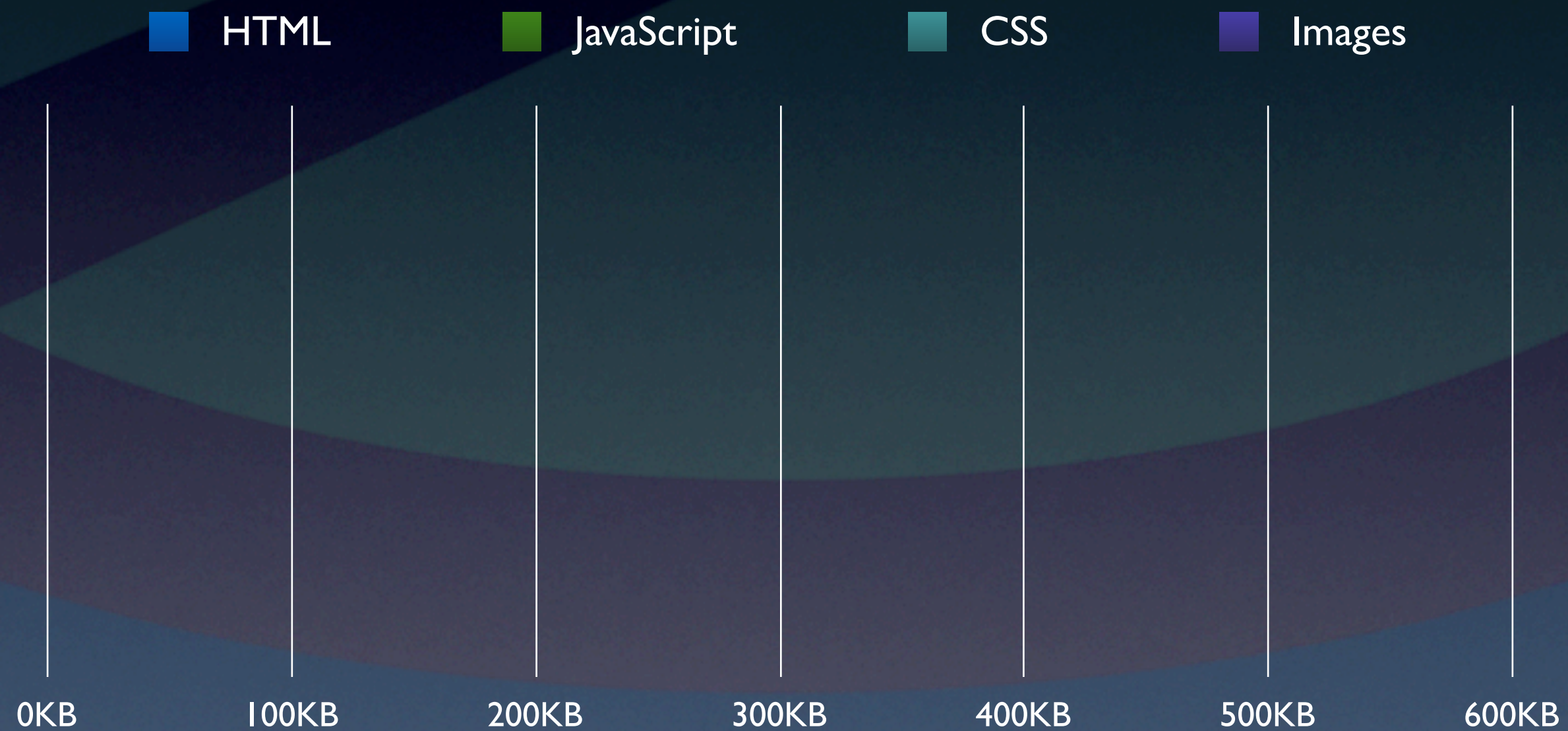
Languages
Simple English

OSCON 2011

WIKIMEDIA

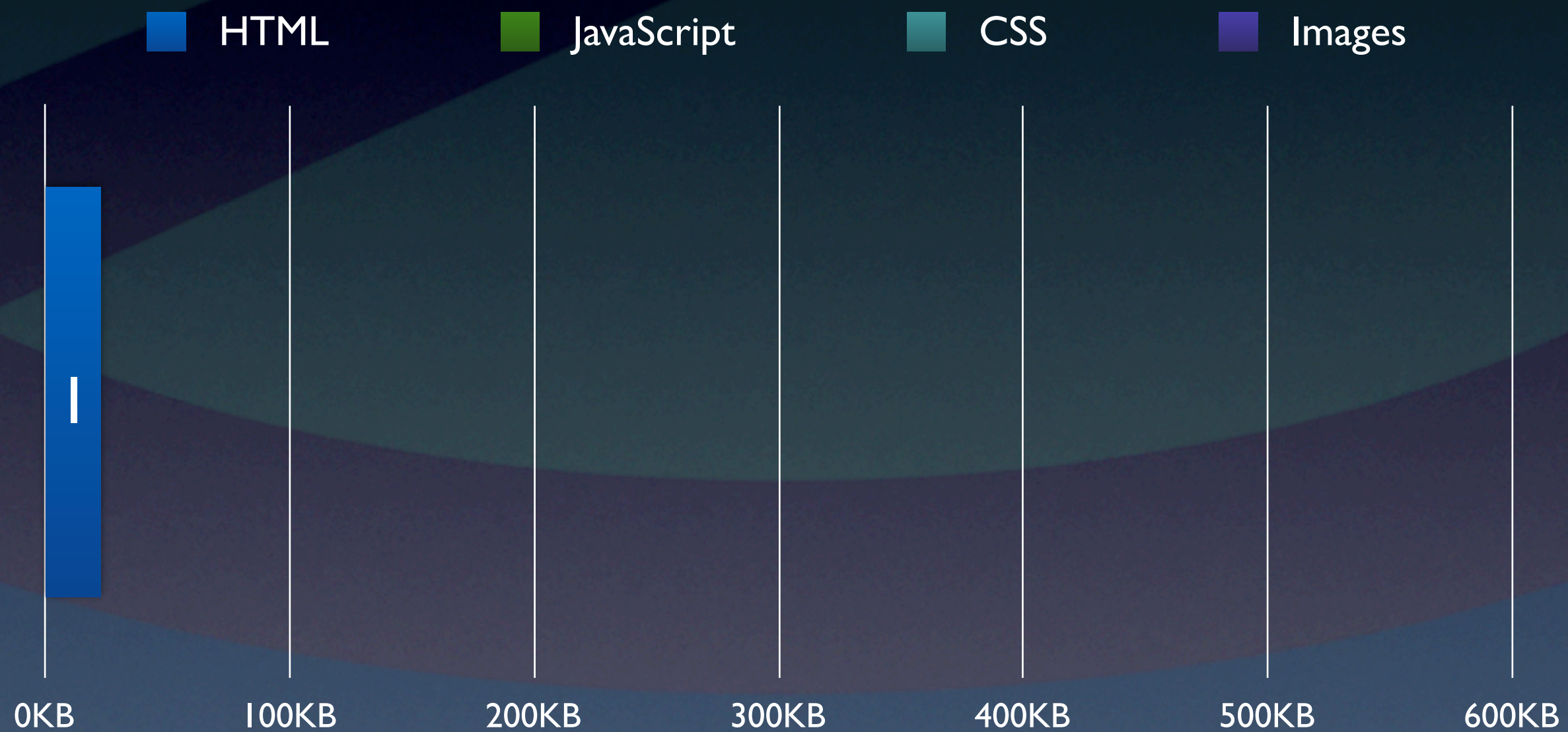
- This is a Wikipedia page
- It's got lots of resources that make it load slowly

Resources



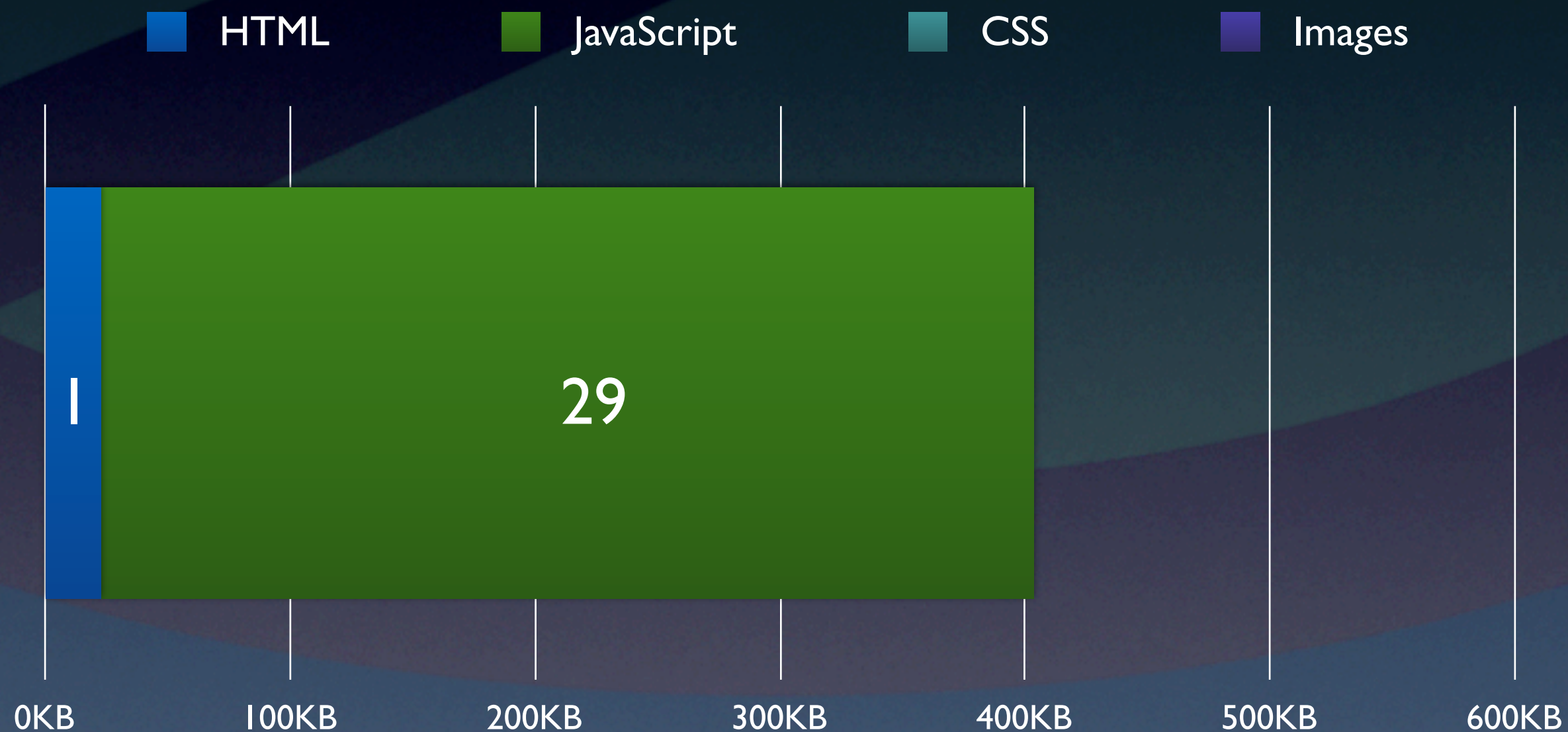
- HTML is a tiny part of the problem
- JS is where most of the gains are
- CSS is large-ish, but at least only a few requests
- Images are small, but lots of requests are costly

Resources



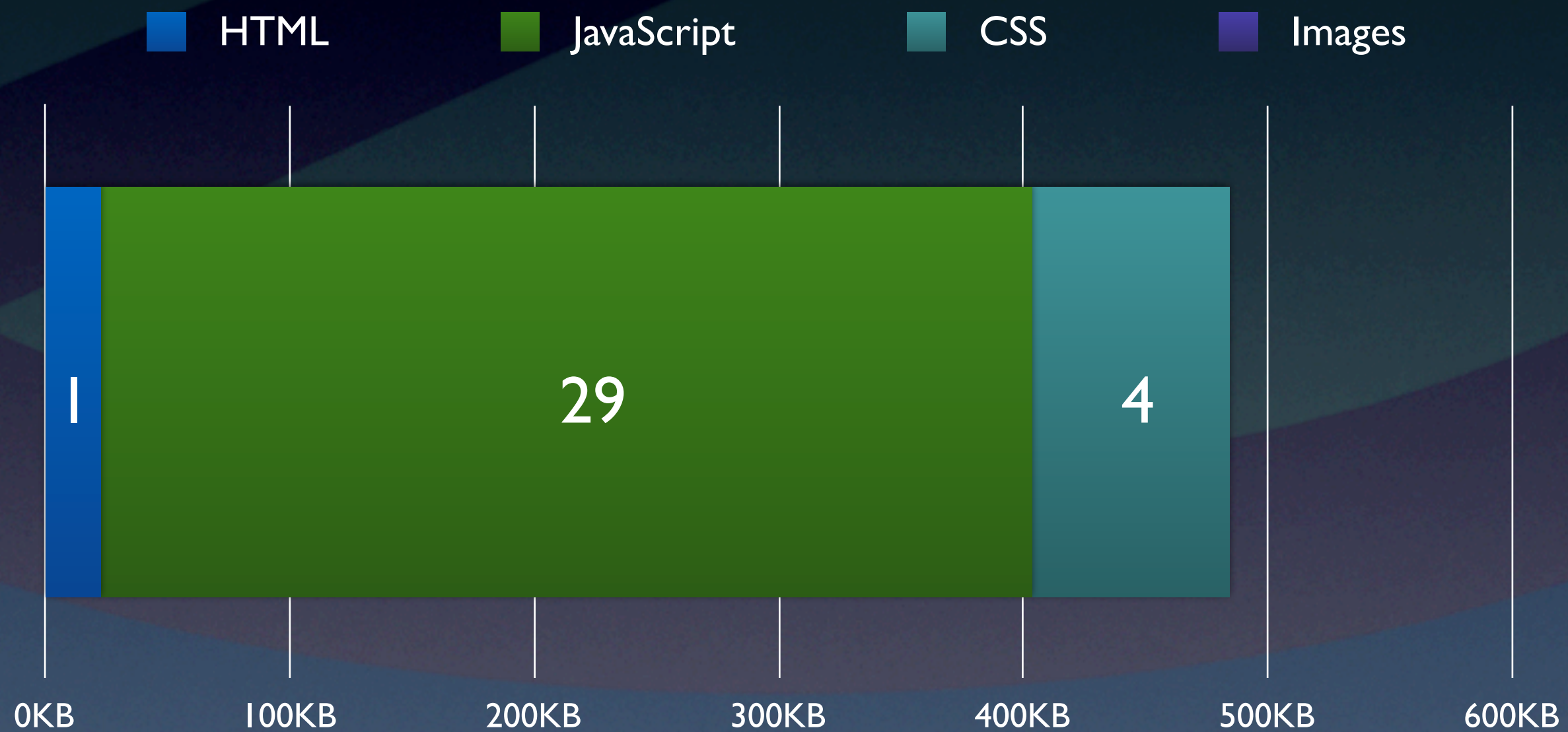
- HTML is a tiny part of the problem
- JS is where most of the gains are
- CSS is large-ish, but at least only a few requests
- Images are small, but lots of requests are costly

Resources



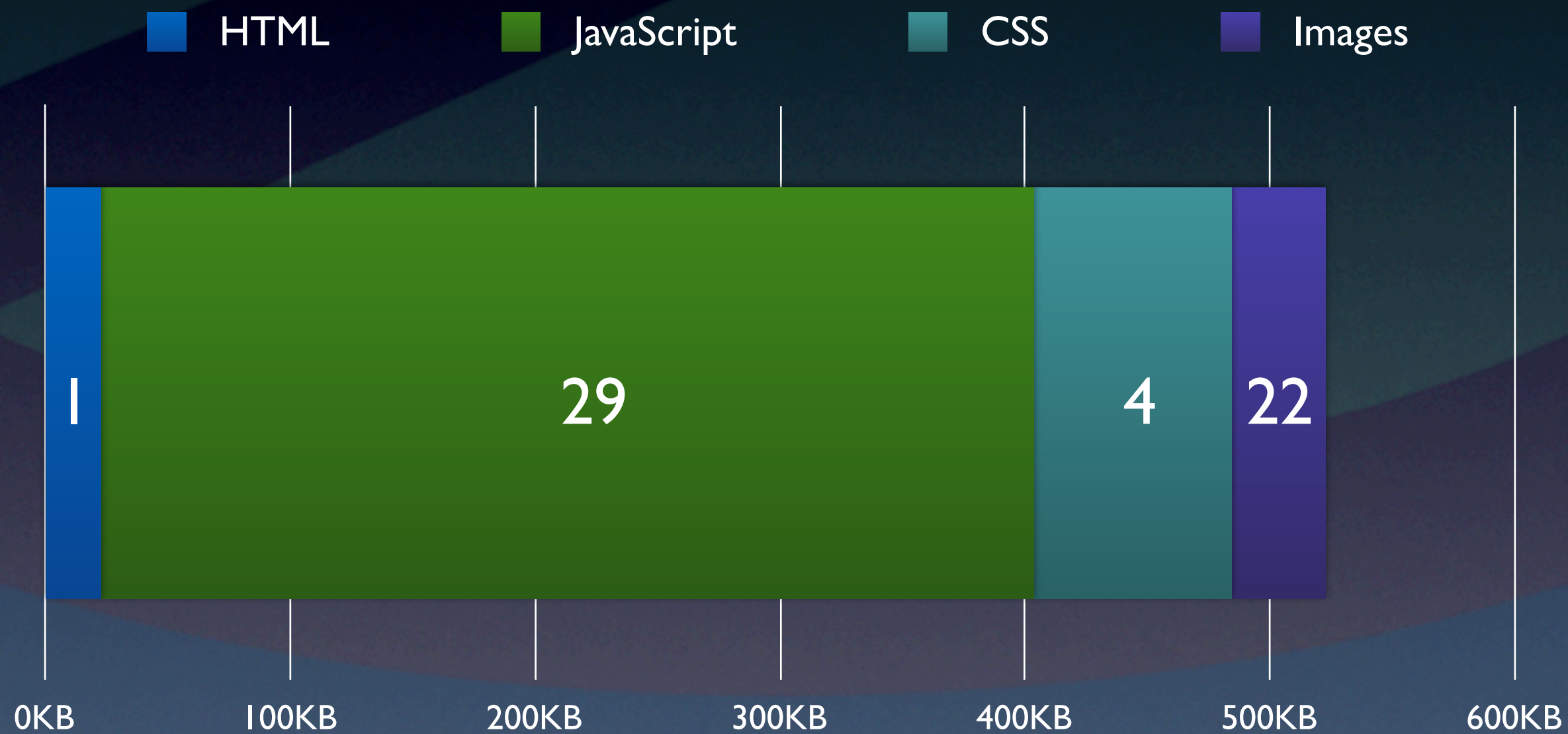
- HTML is a tiny part of the problem
- JS is where most of the gains are
- CSS is large-ish, but at least only a few requests
- Images are small, but lots of requests are costly

Resources



- HTML is a tiny part of the problem
- JS is where most of the gains are
- CSS is large-ish, but at least only a few requests
- Images are small, but lots of requests are costly

Resources



- HTML is a tiny part of the problem
- JS is where most of the gains are
- CSS is large-ish, but at least only a few requests
- Images are small, but lots of requests are costly

Resources

- What are all these things doing?
- Scripts power rich user interfaces – we are doing more of this now
- Styles make things look right – this is getting complex too
- We also need translated messages on the client!

Resources



Scripts

javascript

- What are all these things doing?
- Scripts power rich user interfaces – we are doing more of this now
- Styles make things look right – this is getting complex too
- We also need translated messages on the client!

Resources



Scripts

javascript



Styles

css

- What are all these things doing?
- Scripts power rich user interfaces – we are doing more of this now
- Styles make things look right – this is getting complex too
- We also need translated messages on the client!

Resources



Scripts

javascript



Styles

css



Translations

json

- What are all these things doing?
- Scripts power rich user interfaces – we are doing more of this now
- Styles make things look right – this is getting complex too
- We also need translated messages on the client!

Origins

WIKIMEDIA 

- Where do these resources come from?
- Core software (filesystem)
- Extension software (filesystem)
- User scripts and styles (database)

Origins



Core

skins & common stuff

- Where do these resources come from?
- Core software (filesystem)
- Extension software (filesystem)
- User scripts and styles (database)

Origins



Core

skins & common stuff



Extensions

most everything else

- Where do these resources come from?
- Core software (filesystem)
- Extension software (filesystem)
- User scripts and styles (database)

Origins



Core

skins & common stuff



Extensions

most everything else



Users

`Math.random();`

WIKIMEDIA 

- Where do these resources come from?
- Core software (filesystem)
- Extension software (filesystem)
- User scripts and styles (database)

Optimizations

- Minification wasn't used commonly
- Nothing was combined – no way to know what to combine
- Sprites were not used much, limited functionality & laziness
- We were using gzip at least

Optimizations



Minification

whitespace removal

- Minification wasn't used commonly
- Nothing was combined – no way to know what to combine
- Sprites were not used much, limited functionality & laziness
- We were using gzip at least

Optimizations



Minification

whitespace removal



Combination

file concatenation

- Minification wasn't used commonly
- Nothing was combined – no way to know what to combine
- Sprites were not used much, limited functionality & laziness
- We were using gzip at least

Optimizations



Minification

whitespace removal



Combination

file concatenation



Sprites

multi-graphic images

- Minification wasn't used commonly
- Nothing was combined – no way to know what to combine
- Sprites were not used much, limited functionality & laziness
- We were using gzip at least

Caching



OSCON 2011

WIKIMEDIA 

- Global style version increment on update -- global, so all or nothing
- Must be global because resources are not always known in advance
- Embedded in HTML
- Sucks for purging, version number stuck in Squid-cached HTML

Caching



Versioning

one version for everything

- Global style version increment on update -- global, so all or nothing
- Must be global because resources are not always known in advance
- Embedded in HTML
- Sucks for purging, version number stuck in Squid-cached HTML

Caching



Versioning

one version for everything



Query String

versions in the html

- Global style version increment on update -- global, so all or nothing
- Must be global because resources are not always known in advance
- Embedded in HTML
- Sucks for purging, version number stuck in Squid-cached HTML

Caching



<http://bits.wikimedia.org/skins-1.5/vector/main-ltr.css?123>

Versioning

one version for everything

Query String

versions in the html

- Global style version increment on update – global, so all or nothing
- Must be global because resources are not always known in advance
- Embedded in HTML
- Sucks for purging, version number stuck in Squid-cached HTML

Caching



Versioning

one version for everything



Query String

versions in the html

- Global style version increment on update -- global, so all or nothing
- Must be global because resources are not always known in advance
- Embedded in HTML
- Sucks for purging, version number stuck in Squid-cached HTML

Caching



Versioning

one version for everything



Query String

versions in the html



Purging

wait for cache expiry

- Global style version increment on update -- global, so all or nothing
- Must be global because resources are not always known in advance
- Embedded in HTML
- Sucks for purging, version number stuck in Squid-cached HTML

Being a developer sucked



OSCON 2011

WIKIMEDIA 

- Just getting scripts style and messages to the client was a chore
- RTL had to be done manually
- Localization was awkward and clunky
- Optimization was a pain to maintain
- All of this distracted us from making software better
- User experience was bad, things were slow

Being a developer sucked



Creating good luck!

- Just getting scripts style and messages to the client was a chore
- RTL had to be done manually
- Localization was awkward and clunky
- Optimization was a pain to maintain
- All of this distracted us from making software better
- User experience was bad, things were slow

Being a developer sucked



Creating

good luck!



Maintaining

messy and boring

- Just getting scripts style and messages to the client was a chore
- RTL had to be done manually
- Localization was awkward and clunky
- Optimization was a pain to maintain
- All of this distracted us from making software better
- User experience was bad, things were slow

Being a developer sucked



Creating

good luck!



Maintaining

messy and boring



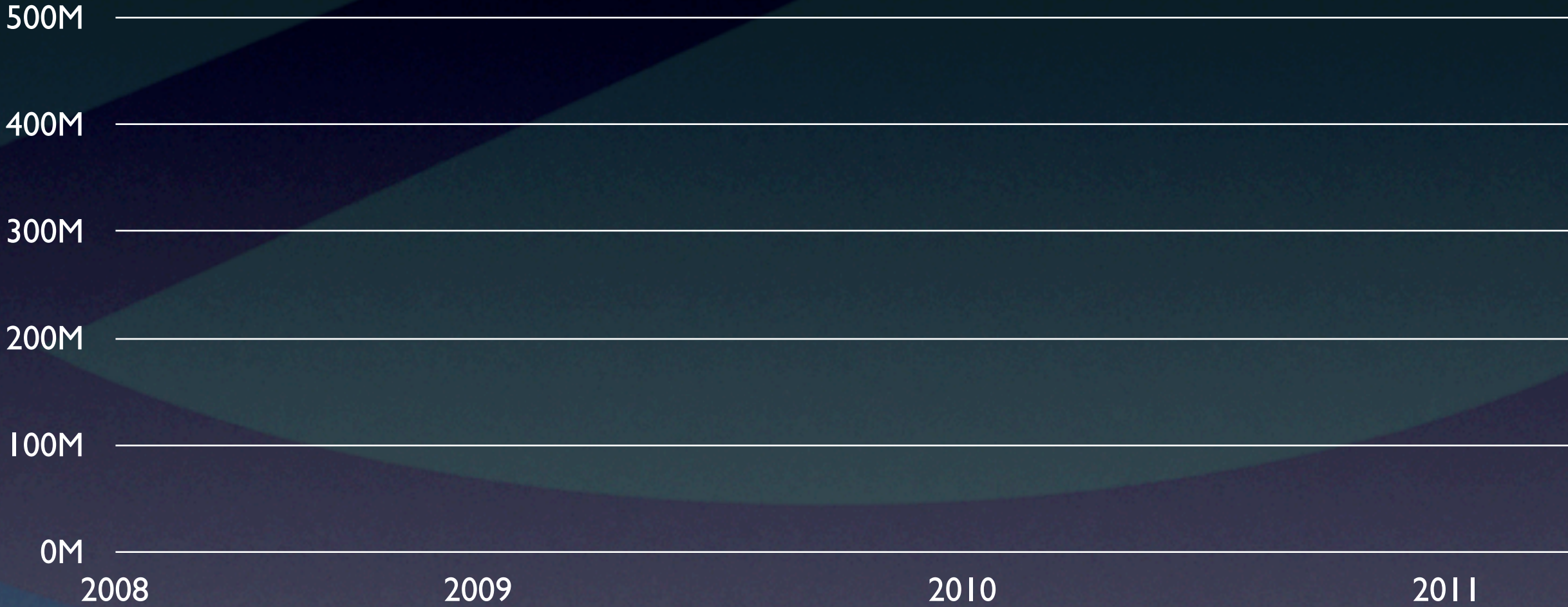
Using

clunky and slow

WIKIMEDIA 

- Just getting scripts style and messages to the client was a chore
- RTL had to be done manually
- Localization was awkward and clunky
- Optimization was a pain to maintain
- All of this distracted us from making software better
- User experience was bad, things were slow

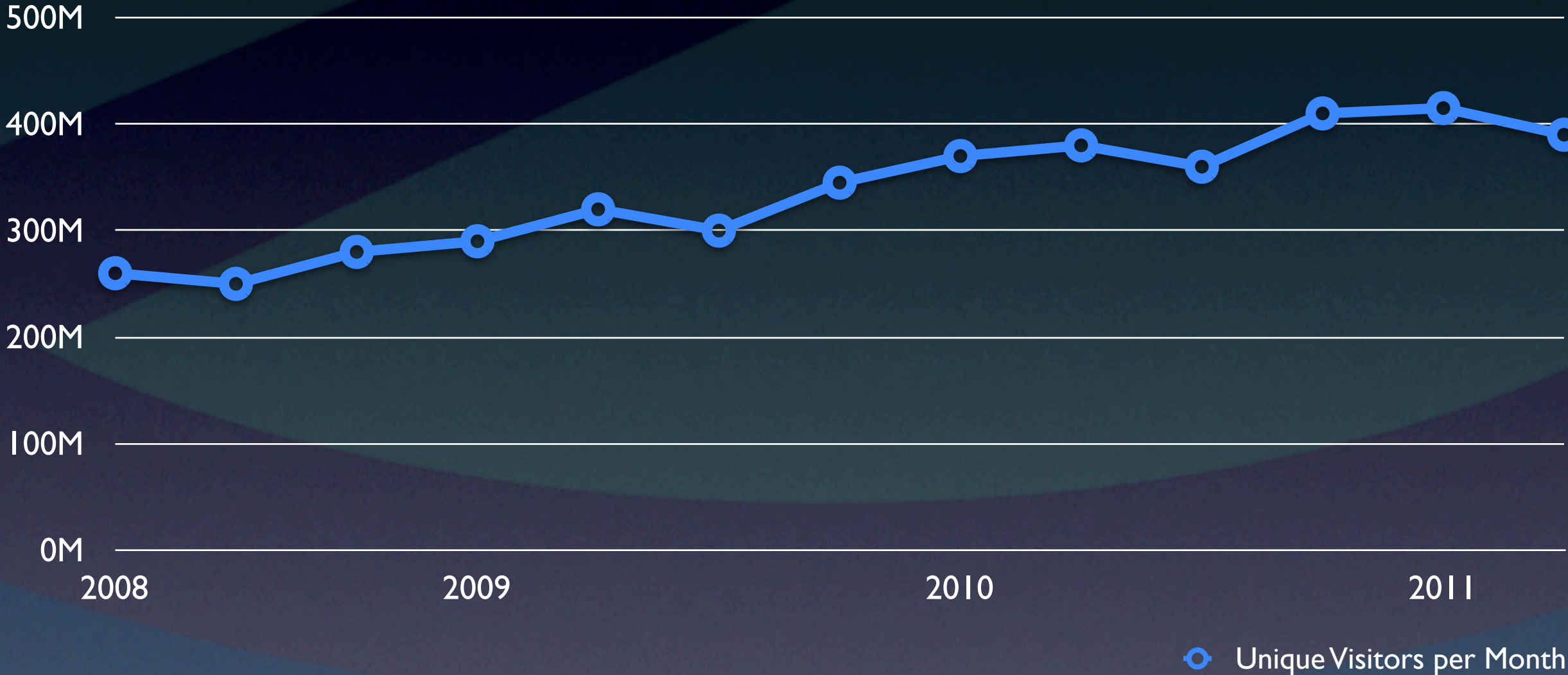
Traffic



WIKIMEDIA 

- These are in MILLIONS
- Per month

Traffic



WIKIMEDIA 

- These are in MILLIONS
- Per month

Cache Invalidation

- User scripts and styles can be changed at any time
 - They only affect that user
- Site scripts and styles can be changed at any time
 - They affect everyone
- Localizations are updated every day

user-specific resources

- User scripts and styles can be changed at any time
 - They only affect that user
- Site scripts and styles can be changed at any time
 - They affect everyone
- Localizations are updated every day

Cache Invalidation



Users

user-specific resources



Admins

site-wide resources

- User scripts and styles can be changed at any time
 - They only affect that user
- Site scripts and styles can be changed at any time
 - They affect everyone
- Localizations are updated every day

Cache Invalidation



Users

user-specific resources



Admins

site-wide resources



Translators

site-wide messages

- User scripts and styles can be changed at any time
 - They only affect that user
- Site scripts and styles can be changed at any time
 - They affect everyone
- Localizations are updated every day

Approach



OSCON 2011

WIKIMEDIA 

- Easy 80%, not chasing down the final 20%
- Requests and client-side caching are more important than server load
- Making it harder to make software makes software worse
- NEXT: This is how it works...

Approach



Easy Gains

no micro-optimization

- Easy 80%, not chasing down the final 20%
- Requests and client-side caching are more important than server load
- Making it harder to make software makes software worse
- NEXT: This is how it works...

Approach



Easy Gains

no micro-optimization



Client Focus

it's where the magic is

- Easy 80%, not chasing down the final 20%
- Requests and client-side caching are more important than server load
- Making it harder to make software makes software worse
- NEXT: This is how it works...

Approach



Easy Gains

no micro-optimization



Client Focus

it's where the magic is



Ease of Use

developers use this stuff

- Easy 80%, not chasing down the final 20%
- Requests and client-side caching are more important than server load
- Making it harder to make software makes software worse
- NEXT: This is how it works...

Modules



OSCON 2011



- Scripts styles and messages can be packaged together
- Or components of modules can be access individually

Modules

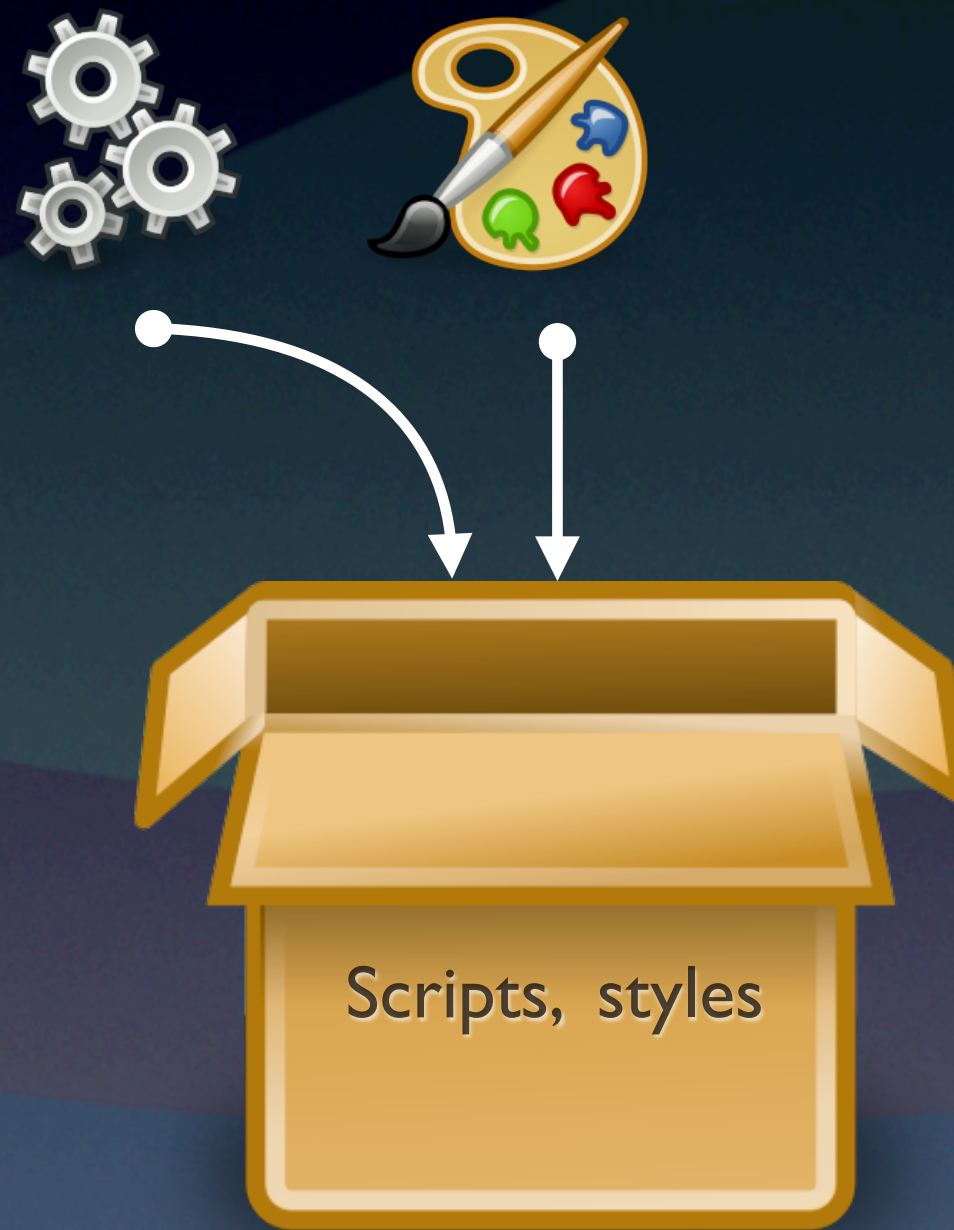


OSCON 2011



- Scripts styles and messages can be packaged together
- Or components of modules can be access individually

Modules



- Scripts styles and messages can be packaged together
- Or components of modules can be access individually

Modules



- Scripts styles and messages can be packaged together
- Or components of modules can be access individually

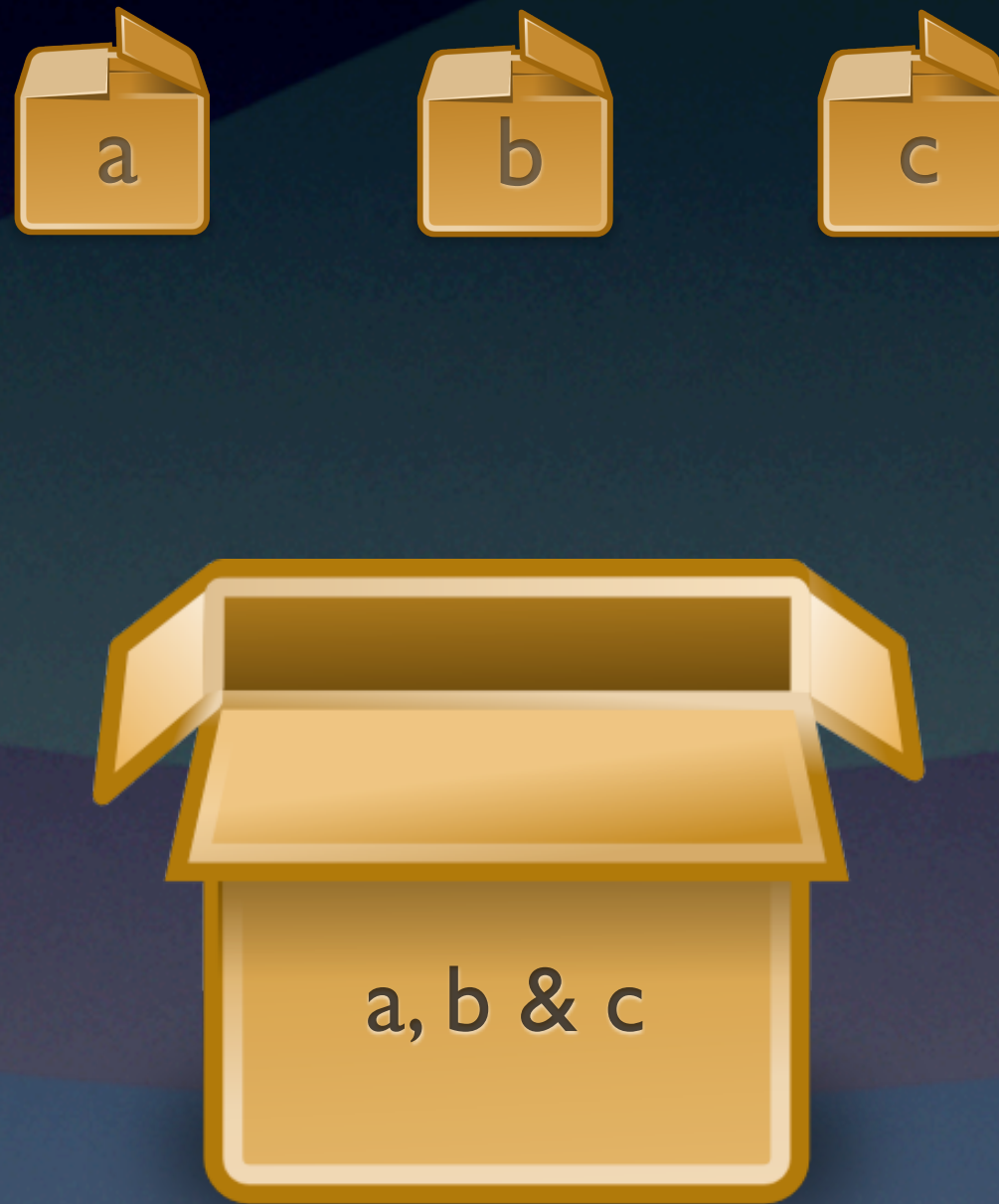
A diagram illustrating the inputs to a system. At the top, there are three icons: a set of three interlocking grey gears, a yellow artist's palette with a paintbrush and three blobs of paint (blue, green, red), and two waving flags (the Spanish flag and the United States flag). Three white curved arrows point from each of these icons down to a large, open, yellow cardboard box. Inside the box, the text "Scripts, styles & messages" is written in a black, sans-serif font.

Modules



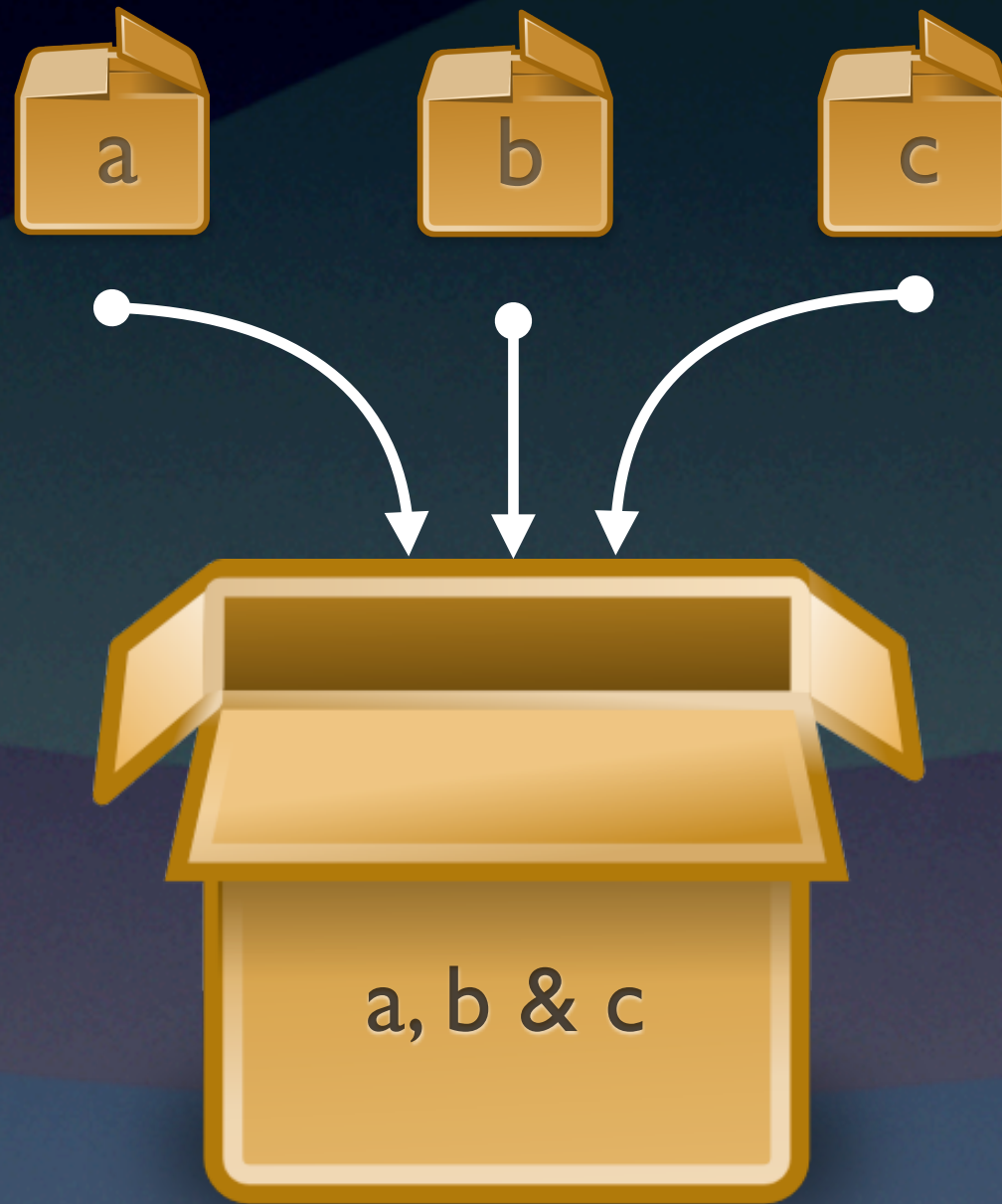
OSCON 2011

Modules



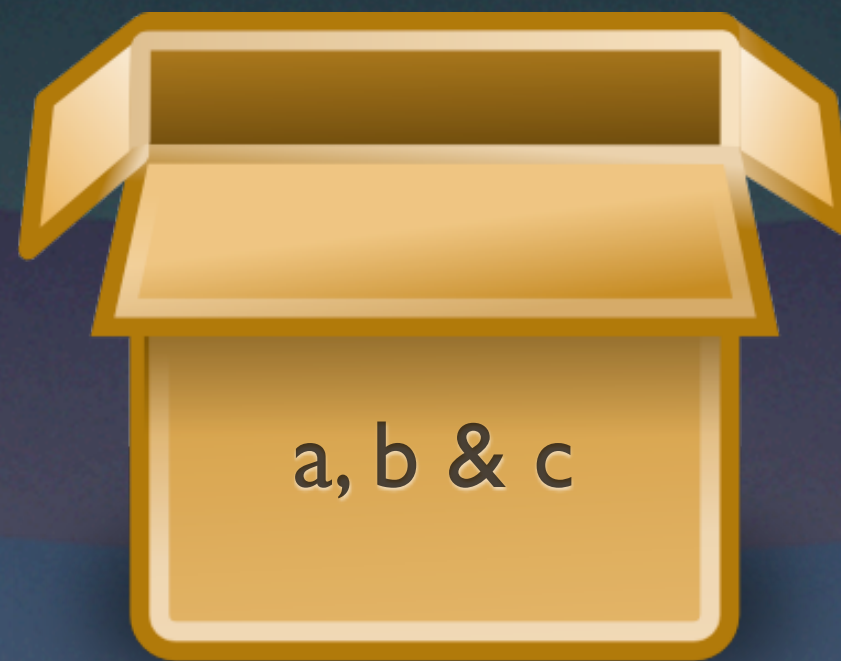
- Any number of modules can be bundled together in a single request

Modules



- Any number of modules can be bundled together in a single request

Modules

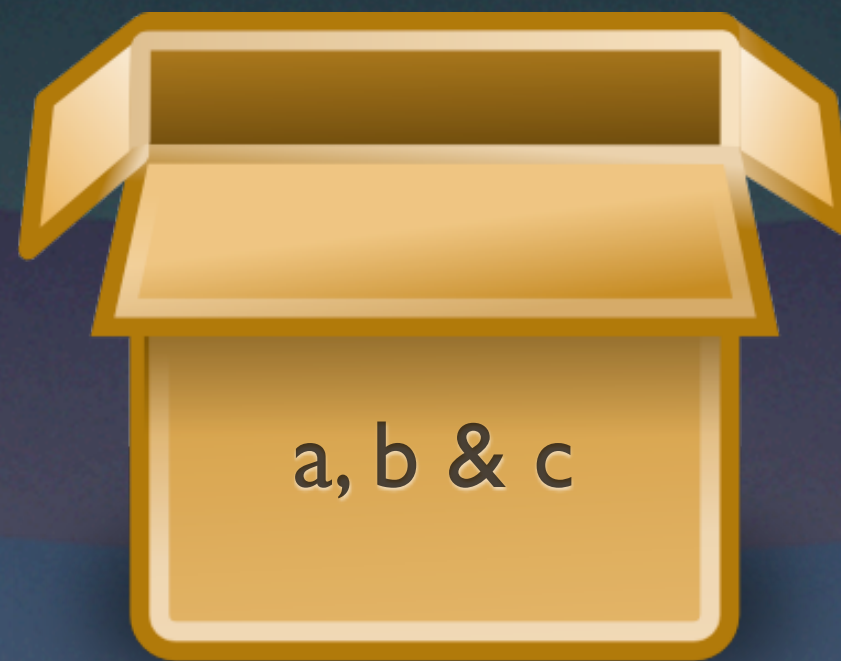


OSCON 2011

WIKIMEDIA 

- Any number of modules can be bundled together in a single request

Modules



OSCON 2011

Modules



OSCON 2011

Scripts



OSCON 2011

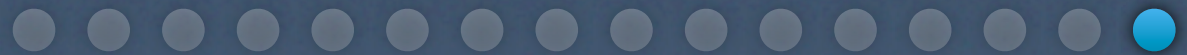


Scripts



Wrapping

delayed execution



OSCON 2011

WIKIMEDIA 

Scripts



Wrapping

delayed execution



Minification

whitespace removal

Scripts



Wrapping

delayed execution



Minification

whitespace removal



Conditions

debug, skin or language

Styles

- Who's used CSS sprites before?
- Explains sprites (good and bad)
- It's all magical and transparent!
- Remapping makes injected styles still work
- Flipping (more about that later)

Styles



Embedding

data uri sweetness

- Who's used CSS sprites before?
- Explains sprites (good and bad)
- It's all magical and transparent!
- Remapping makes injected styles still work
- Flipping (more about that later)

Styles

```
background-image: url(data:image/png;base64,iVBORw0KGgoA...FTkSuQmCC);
```

Embedding

data uri sweetness

- Who's used CSS sprites before?
- Explains sprites (good and bad)
- It's all magical and transparent!
- Remapping makes injected styles still work
- Flipping (more about that later)

Styles



Embedding

data uri sweetness

- Who's used CSS sprites before?
- Explains sprites (good and bad)
- It's all magical and transparent!
- Remapping makes injected styles still work
- Flipping (more about that later)

Styles



Embedding

data uri sweetness



Remapping

relative urls still work

- Who's used CSS sprites before?
- Explains sprites (good and bad)
- It's all magical and transparent!
- Remapping makes injected styles still work
- Flipping (more about that later)

Styles



Embedding

data uri sweetness



Remapping

relative urls still work



Flipping

free rtl support

- Who's used CSS sprites before?
- Explains sprites (good and bad)
- It's all magical and transparent!
- Remapping makes injected styles still work
- Flipping (more about that later)

Styles



Bundling

one request

- Who's used CSS sprites before?
- Explains sprites (good and bad)
- It's all magical and transparent!
- Remapping makes injected styles still work
- Flipping (more about that later)

Styles



Bundling

one request



OSCON 2011

WIKIMEDIA 

- Sent together with JS
- Whitespace is gone
- Skin-specific styles is good

Styles



Bundling

one request



Minification

whitespace removal



OSCON 2011

WIKIMEDIA 

- Sent together with JS
- Whitespace is gone
- Skin-specific styles is good

Styles



Bundling

one request



Minification

whitespace removal



Conditions

skin specific styles

- Sent together with JS
- Whitespace is gone
- Skin-specific styles is good

Messages



OSCON 2011



Messages



Bundling

one request



OSCON 2011

Messages



Bundling

one request



Conditions

language

Startup Module

- Bail on old-school browsers
- Provide module manifest for dependency resolution
- Tack on site-wide configuration

Startup Module



Sanity check

```
if ( IE5 ) { giveUp(); }
```

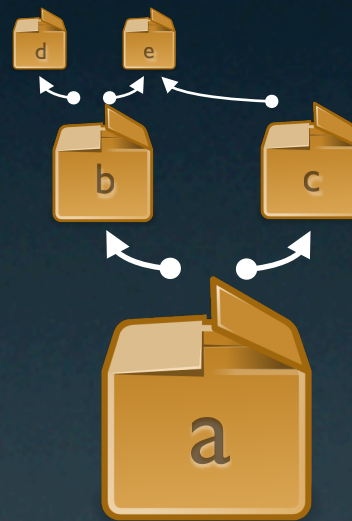
- Bail on old-school browsers
- Provide module manifest for dependency resolution
- Tack on site-wide configuration

Startup Module



Sanity check

```
if ( IE5 ) { giveUp(); }
```



Dependencies

module manifest

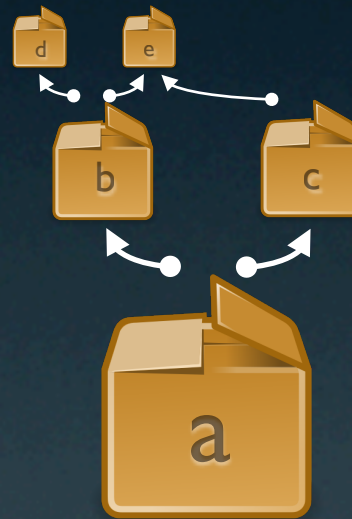
- Bail on old-school browsers
- Provide module manifest for dependency resolution
- Tack on site-wide configuration

Startup Module



Sanity check

`if (IE5) { giveUp(); }`



Dependencies

module manifest



Configuration

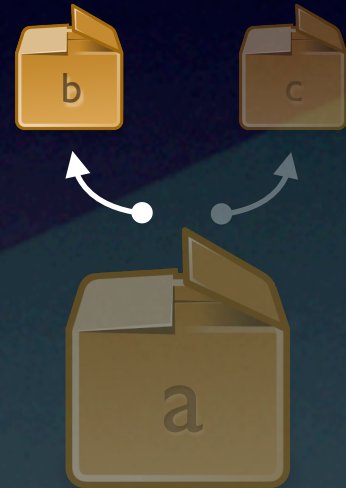
site-wide settings

- Bail on old-school browsers
- Provide module manifest for dependency resolution
- Tack on site-wide configuration

Client-side Loader

- Automatic resolution, never loads something twice
- Module are requested in batches
- Modules (scripts, styles, messages) are executed in proper order

Client-side Loader

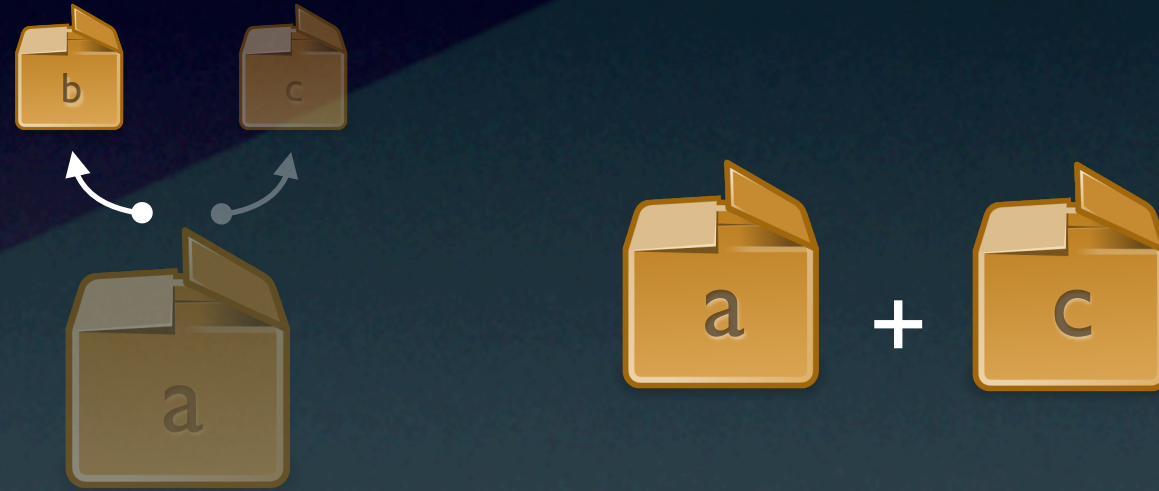


Resolution

calculate dependencies

- Automatic resolution, never loads something twice
- Module are requested in batches
- Modules (scripts, styles, messages) are executed in proper order

Client-side Loader



Resolution

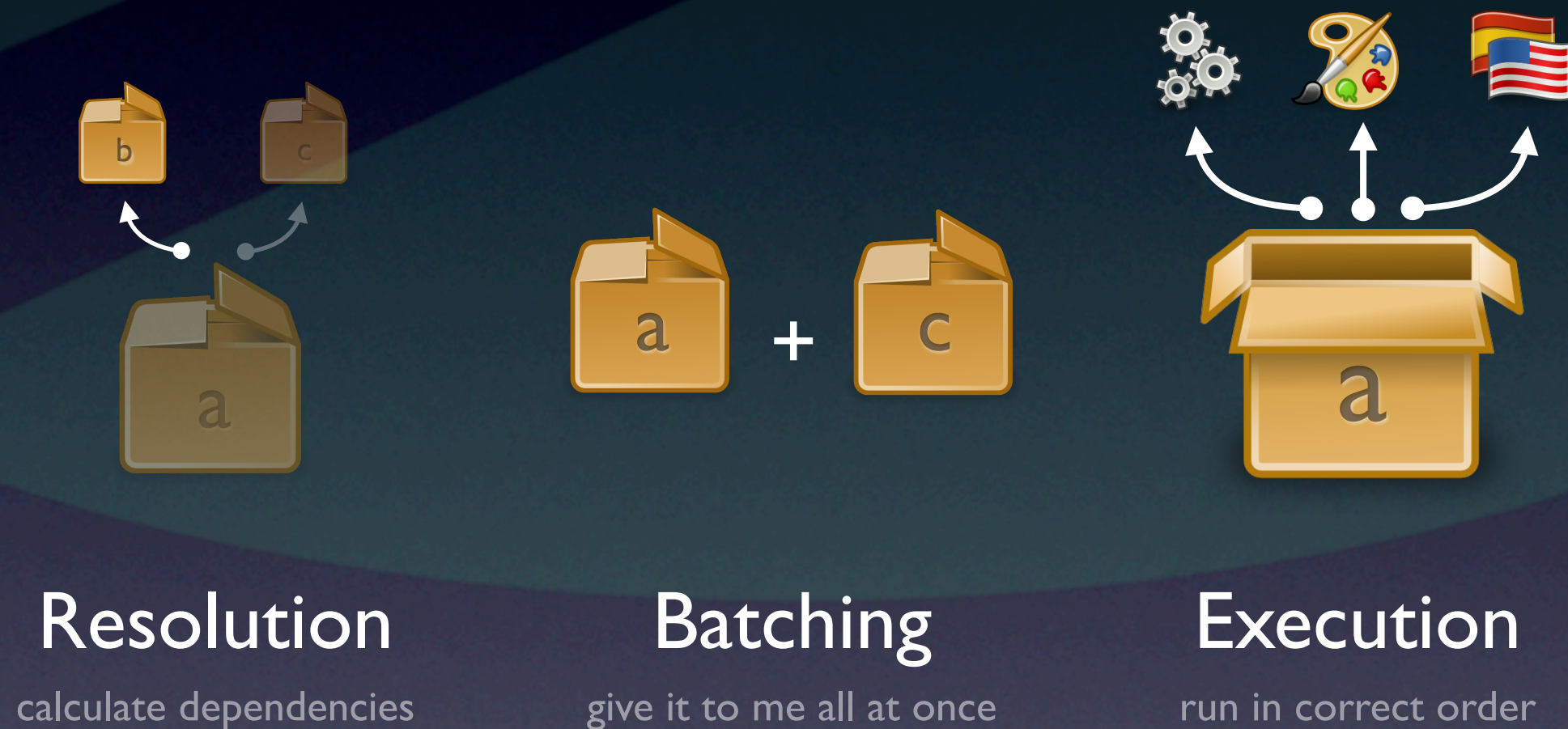
calculate dependencies

Batching

give it to me all at once

- Automatic resolution, never loads something twice
- Module are requested in batches
- Modules (scripts, styles, messages) are executed in proper order

Client-side Loader

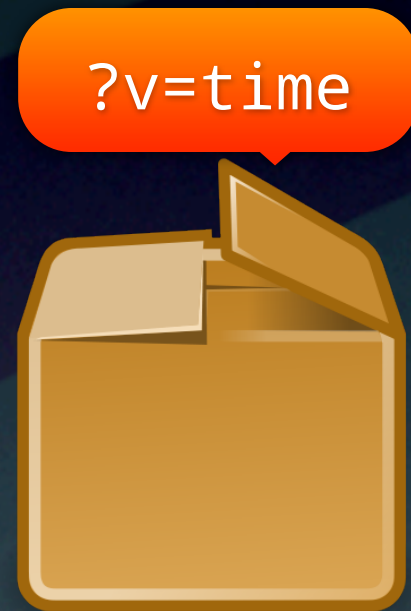


- Automatic resolution, never loads something twice
- Module are requested in batches
- Modules (scripts, styles, messages) are executed in proper order

Caching

- We use timestamps, based on latest modified time of resources in module
- Startup module is cached for 5 min (talk about caching)
- Resources are cached for a month, could be forever

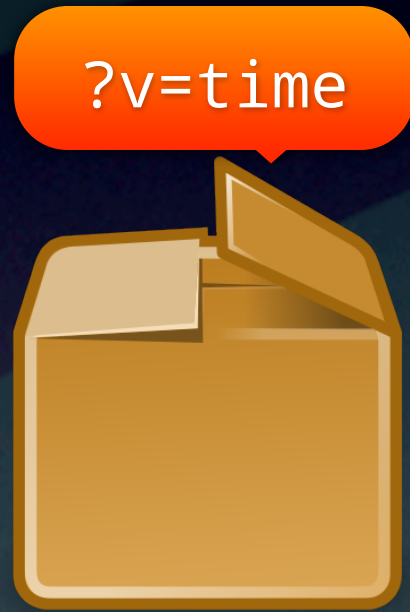
Caching



Versioning
per-module

- We use timestamps, based on latest modified time of resources in module
- Startup module is cached for 5 min (talk about caching)
- Resources are cached for a month, could be forever

Caching



Versioning

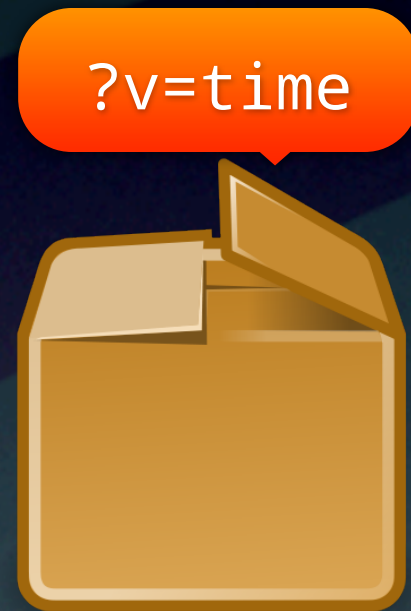
per-module

Startup Module

5 minutes

- We use timestamps, based on latest modified time of resources in module
- Startup module is cached for 5 min (talk about caching)
- Resources are cached for a month, could be forever

Caching



Versioning
per-module



Startup Module
5 minutes



Resources
30 days

- We use timestamps, based on latest modified time of resources in module
- Startup module is cached for 5 min (talk about caching)
- Resources are cached for a month, could be forever

So, it turns out...

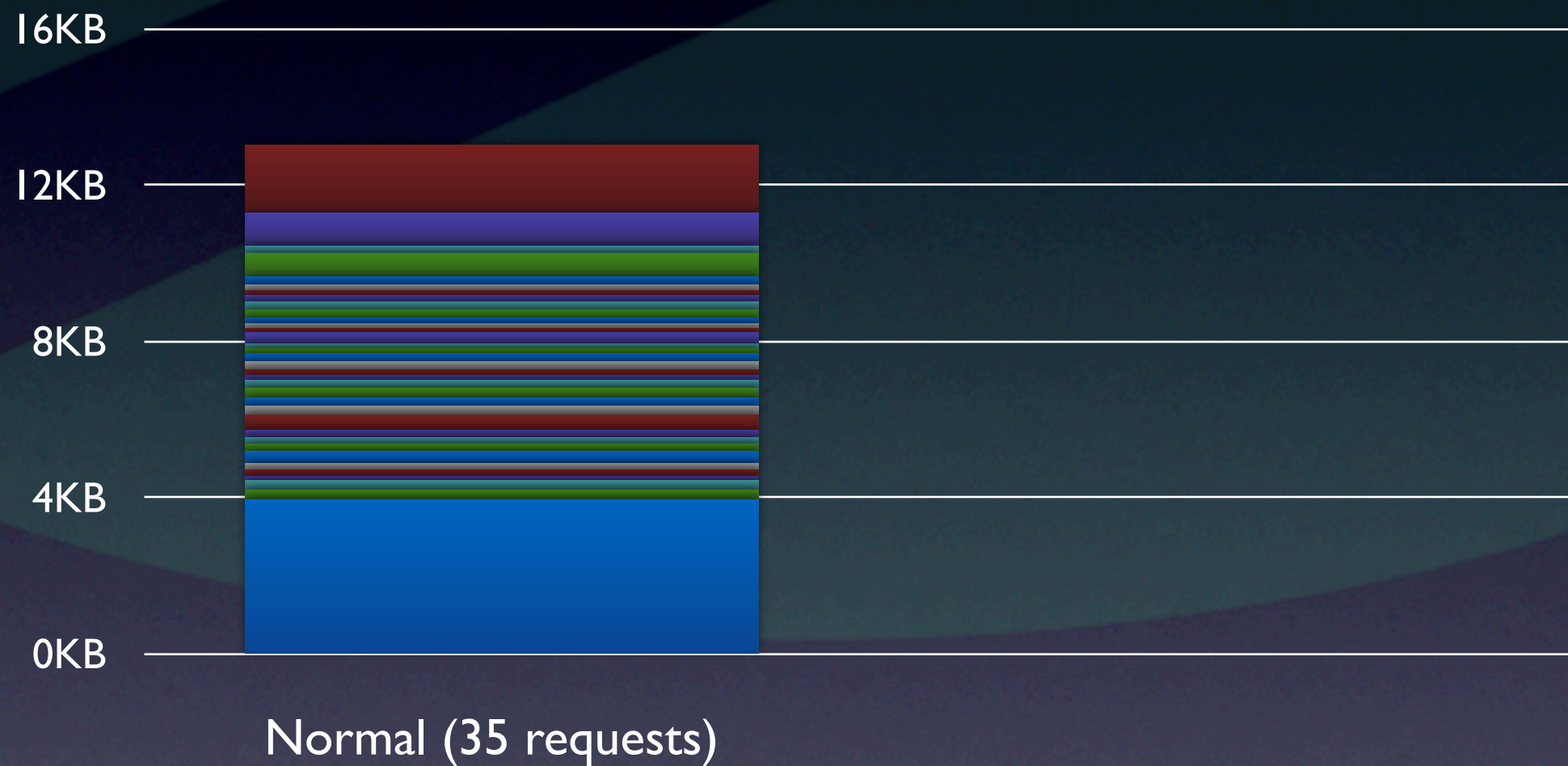
Embedding is sweet

- Vector normally would have 35 requests, mostly images
- Despite base64 inflation, size is smaller after gzip
- Gzip gets a chance to combine common PNG headers and pixel data
- Plus, sprites are a pain to get right, to maintain, can't always be used
- Embedding is idiot-proof: just type `/* @embed */`

Embedding is sweet

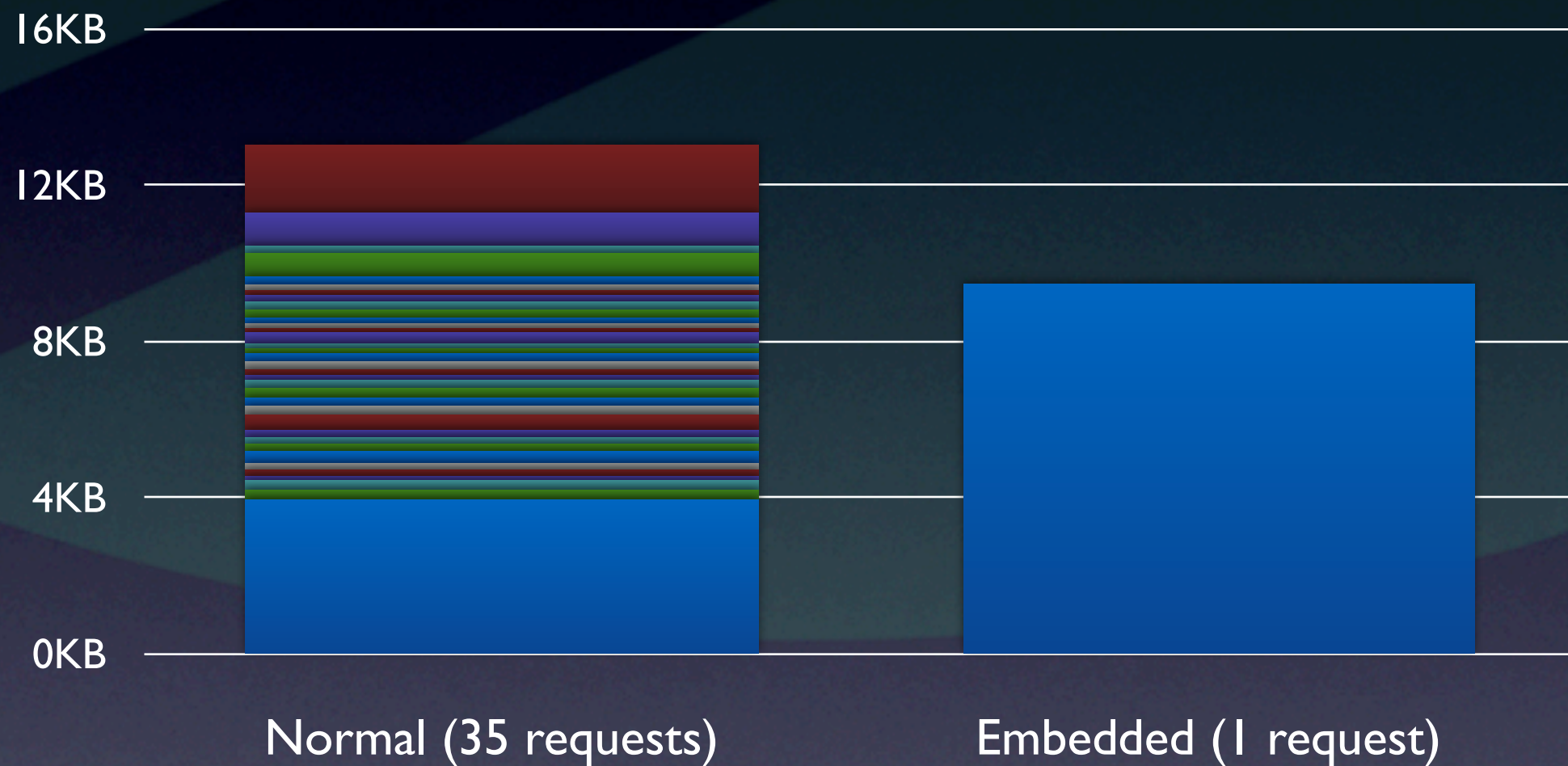
- Vector normally would have 35 requests, mostly images
- Despite base64 inflation, size is smaller after gzip
- Gzip gets a chance to combine common PNG headers and pixel data
- Plus, sprites are a pain to get right, to maintain, can't always be used
- Embedding is idiot-proof: just type `/* @embed */`

Embedding is sweet



- Vector normally would have 35 requests, mostly images
- Despite base64 inflation, size is smaller after gzip
- Gzip gets a chance to combine common PNG headers and pixel data
- Plus, sprites are a pain to get right, to maintain, can't always be used
- Embedding is idiot-proof: just type `/* @embed */`

Embedding is sweet



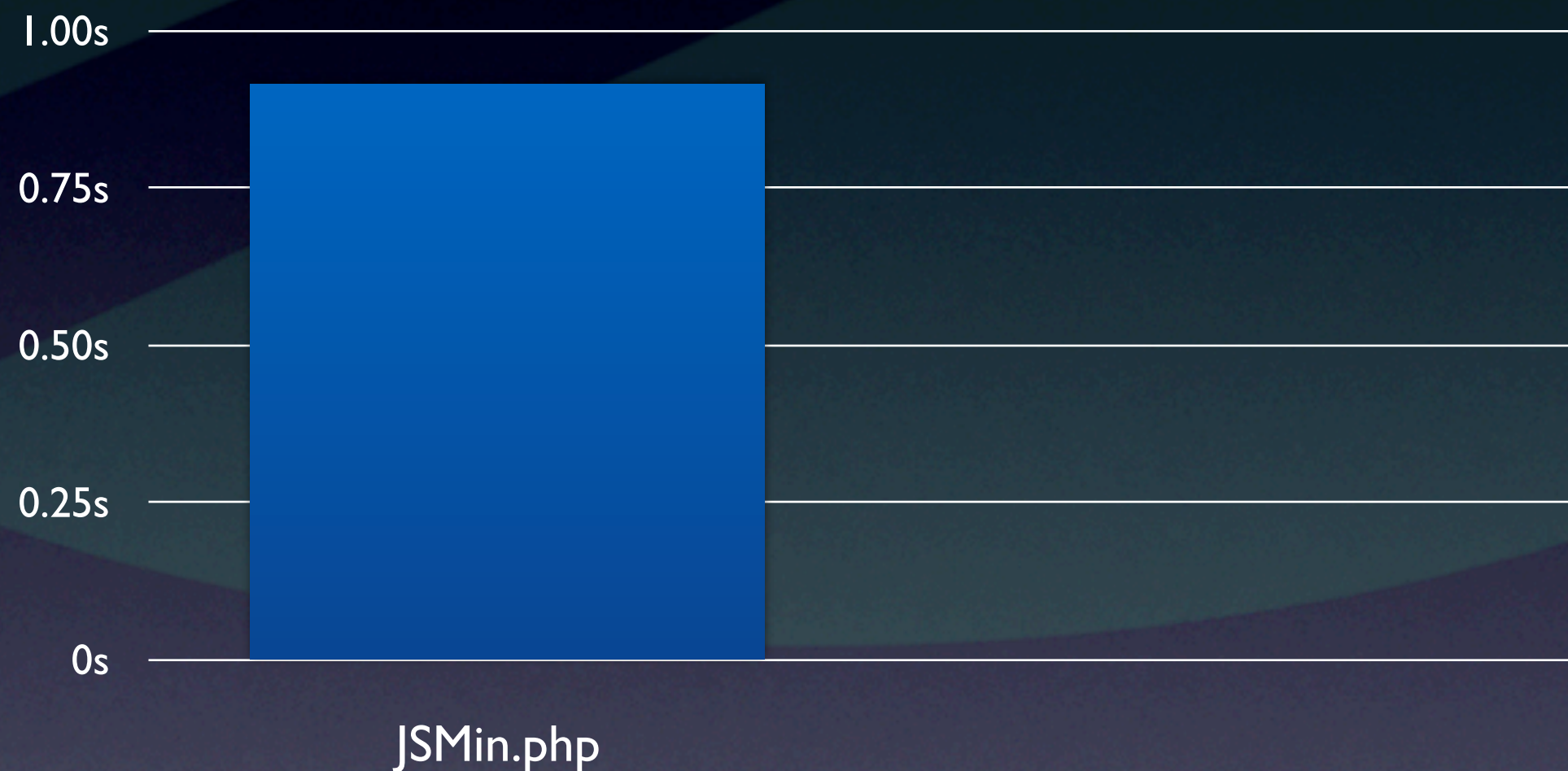
CSSMin: <http://tinyurl.com/CSSMin-php>

- Vector normally would have 35 requests, mostly images
- Despite base64 inflation, size is smaller after gzip
- Gzip gets a chance to combine common PNG headers and pixel data
- Plus, sprites are a pain to get right, to maintain, can't always be used
- Embedding is idiot-proof: just type `/* @embed */`

JSTMin.php is slow

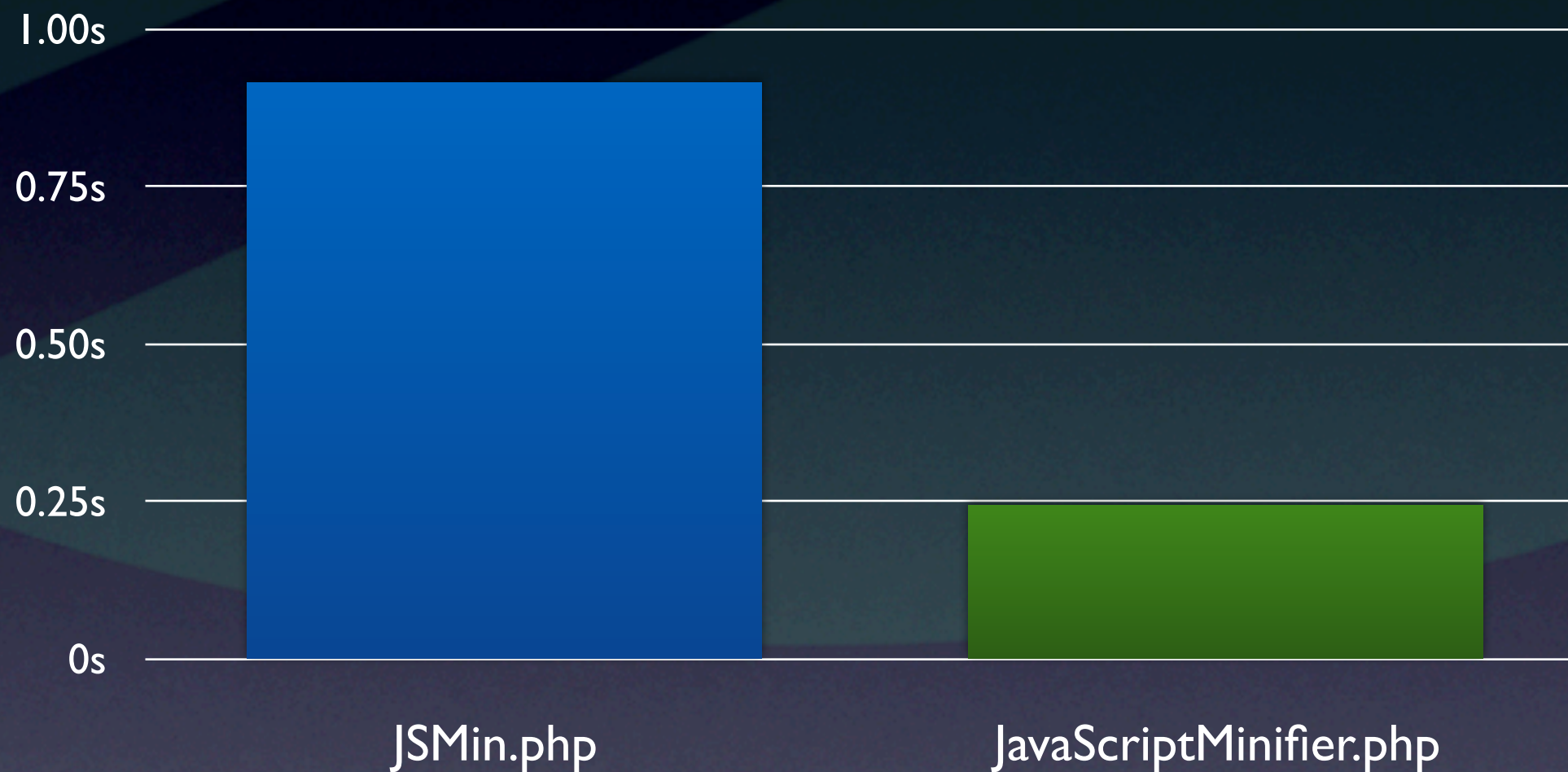
- Written by Paul Copperman (volunteer)
- 0.5% difference
- More correct than JSMIn!
- Stand-alone code, download and use!

JSMin.php is slow



- Written by Paul Copperman (volunteer)
- 0.5% difference
- More correct than JSMin!
- Stand-alone code, download and use!

JSMIn.php is slow



JavaScriptMinifier: <http://tinyurl.com/JavaScriptMinifier>

- Written by Paul Copperman (volunteer)
- 0.5% difference
- More correct than JSMIn!
- Stand-alone code, download and use!

CSS Janus is Awesome

- RTL for free!
- Can use @noflip when needed
- Even just blindly flipping styles, results are amazing

CSS Janus is Awesome



Ported

from python



\$humans--;

seriously, go robots!

CSSJanus: <http://tinyurl.com/CSSJanus>

WIKIMEDIA 

- RTL for free!
- Can use @noflip when needed
- Even just blindly flipping styles, results are amazing

Balance is important

- Overzealous batching causes cache fragmentation
- Added grouping to intentionally split requests
- Groups are defined in module manifest
- Typical value for b: JUI

duplicate data may be sent

- 

The diagram illustrates the difference between a single join and a join with a filter. On the left, two tables are joined, resulting in two identical output documents. On the right, three tables are joined with a filter, resulting in two identical output documents.

duplicate data may be sent

controlled fragmentation

- Overzealous batching causes cache fragmentation
- Added grouping to intentionally split requests
- Groups are defined in module manifest
- Typical value for b: JUI

It Works!

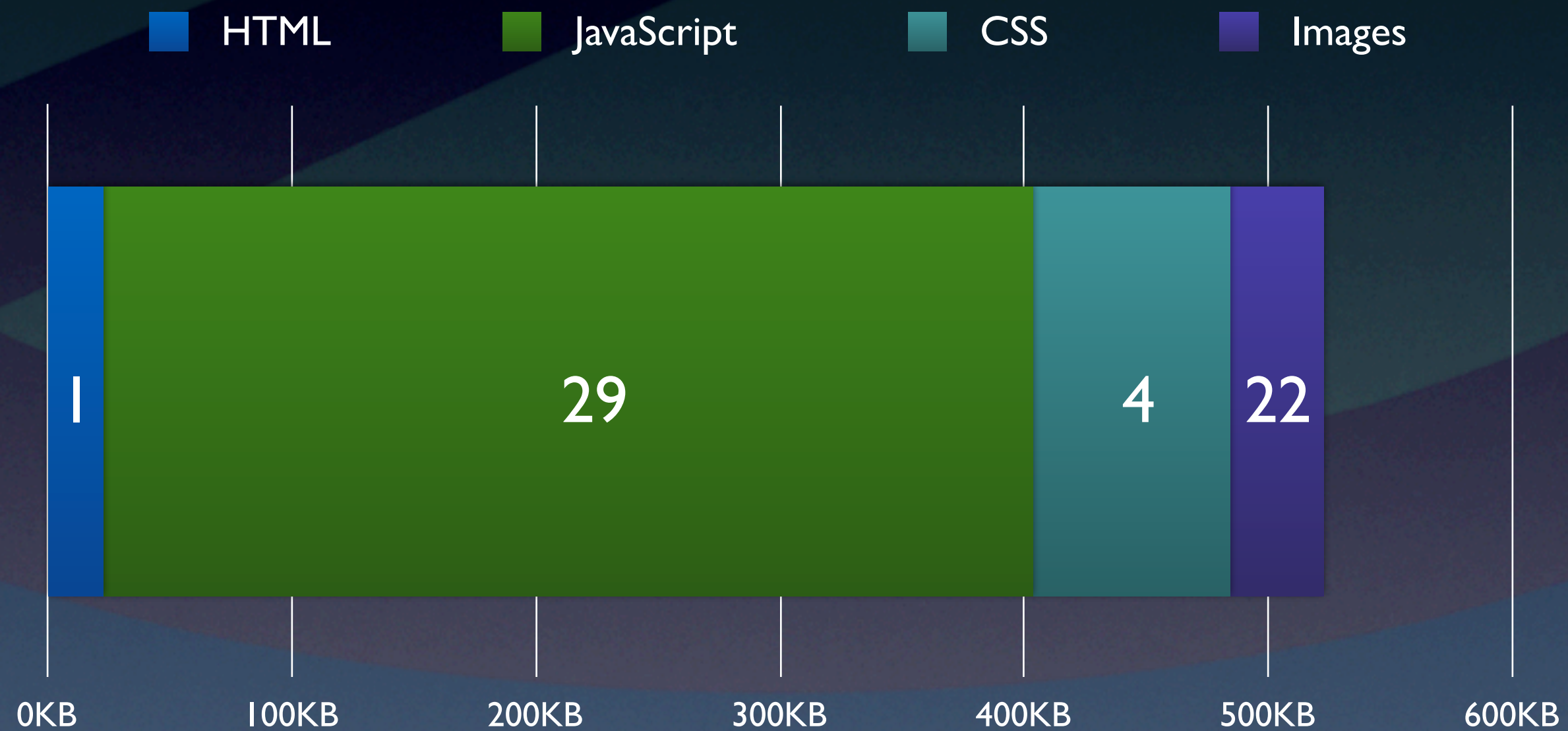


OSCON 2011



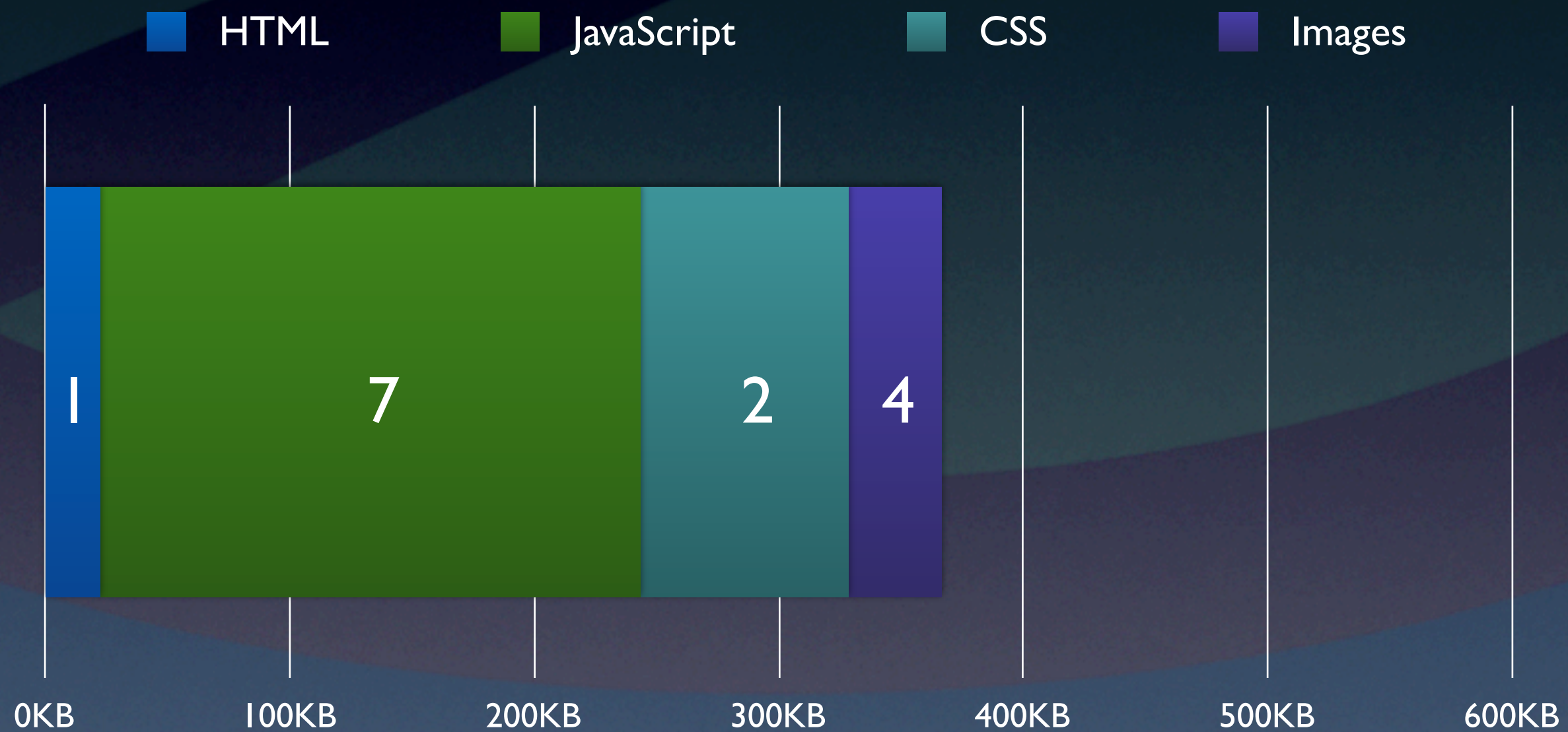
– You recall this chart?

It Works!



– You recall this chart?

It Works!



- Dramatic reduction in number of requests
- Less total data transfer, gzip does it's thing

It's Efficient!

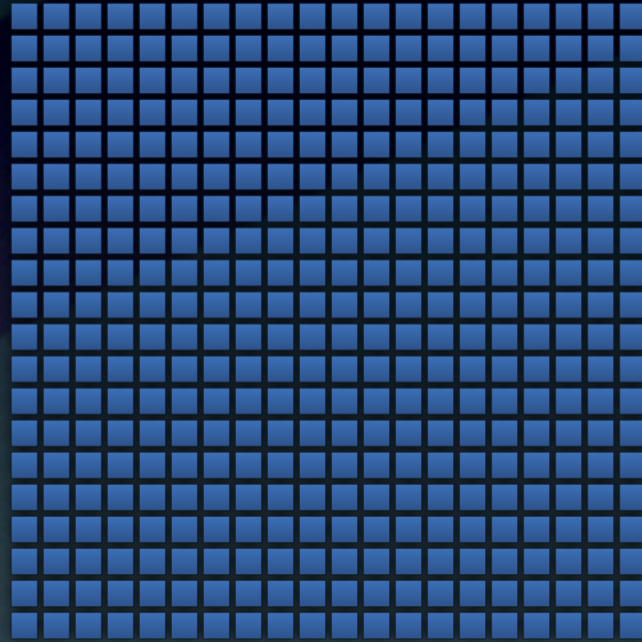
- Guess how many servers?
- 13 Resource servers (4 app, 9 cache)
- **Not even fully optimized**

It's Efficient!

Servers

- Guess how many servers?
- 13 Resource servers (4 app, 9 cache)
- **Not even fully optimized**

It's Efficient!

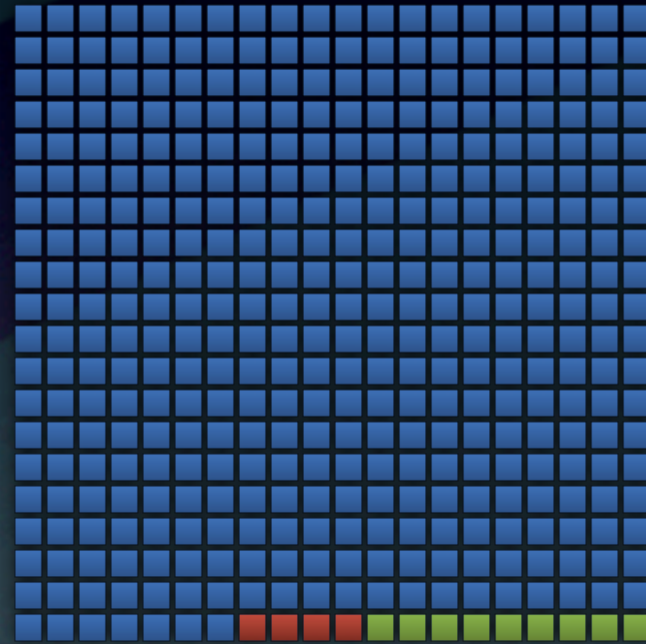


Servers

~400 servers

- Guess how many servers?
- 13 Resource servers (4 app, 9 cache)
- **Not even fully optimized**

It's Efficient!



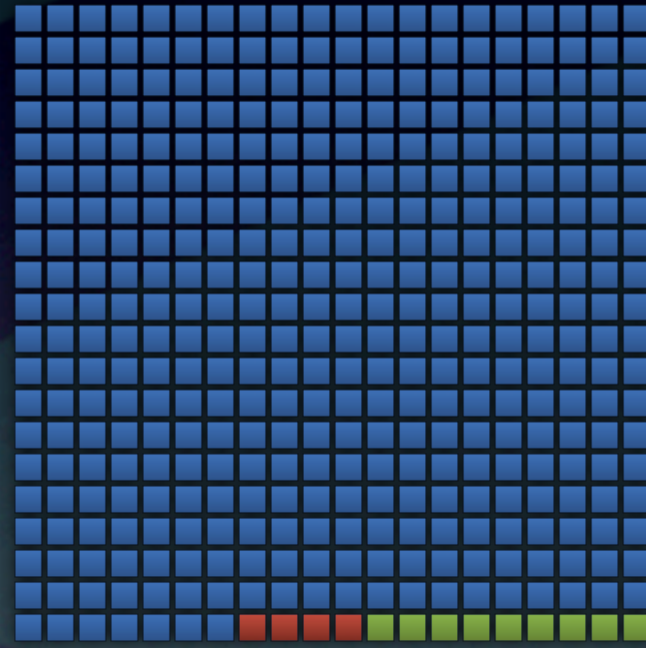
Servers

~400 servers

For resources: 4 app, 9 cache

- Guess how many servers?
- 13 Resource servers (4 app, 9 cache)
- **Not even fully optimized**

It's Efficient!



Servers

~400 servers

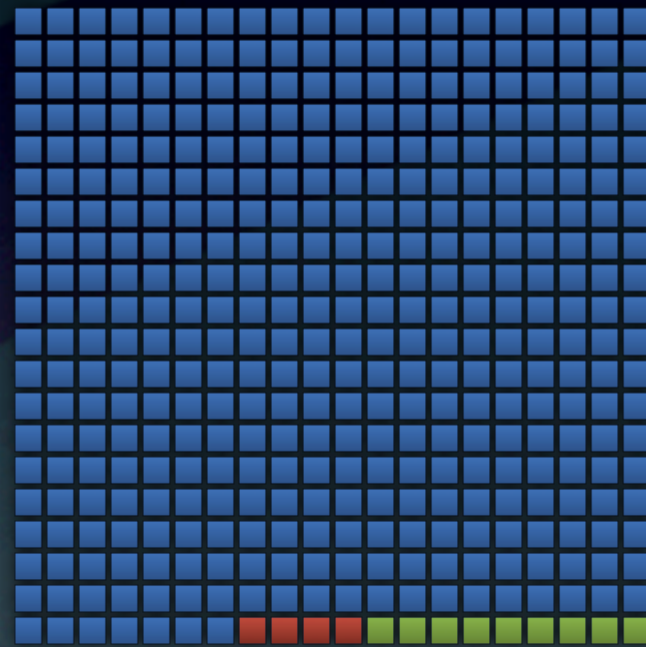
For resources: 4 app, 9 cache

Requests

90k req/s peak load

- Guess how many servers?
- 13 Resource servers (4 app, 9 cache)
- **Not even fully optimized**

It's Efficient!



Servers

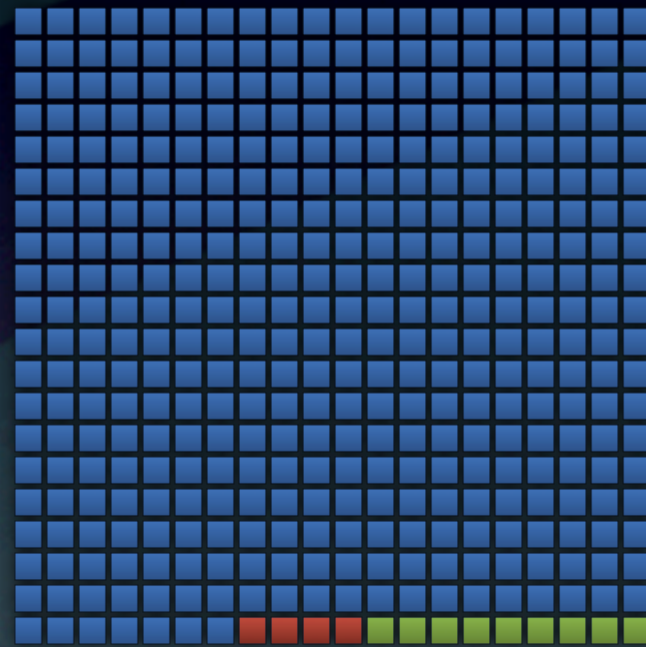
~400 servers
For resources: 4 app, 9 cache

Requests

90k req/s peak load
of which 40k are for resources

- Guess how many servers?
- 13 Resource servers (4 app, 9 cache)
- **Not even fully optimized**

It's Efficient!



Servers

~400 servers
For resources: 4 app, 9 cache



Requests

90k req/s peak load
of which 40k are for resources
and 73 are cache misses
cache hit rate: 98.2%

- Guess how many servers?
- 13 Resource servers (4 app, 9 cache)
- **Not even fully optimized**

ResourceLoader

- To recap, we employed these techniques, focusing on:
 - Easy gains
 - The client
 - Developer friendliness

ResourceLoader



Easy Gains

no micro-optimization

- To recap, we employed these techniques, focusing on:
 - Easy gains
 - The client
 - Developer friendliness

ResourceLoader



Easy Gains

no micro-optimization



Client Focus

it's where the magic is

- To recap, we employed these techniques, focusing on:
 - Easy gains
 - The client
 - Developer friendliness

ResourceLoader



Easy Gains

no micro-optimization



Client Focus

it's where the magic is



Ease of Use

developers use this stuff

WIKIMEDIA 

- To recap, we employed these techniques, focusing on:
 - Easy gains
 - The client
 - Developer friendliness

ResourceLoader



Easy to Make

happy developers

- To recap, we employed these techniques, focusing on:
 - Easy gains
 - The client
 - Developer friendliness

ResourceLoader



Easy to Make

happy developers

WIKIMEDIA 

- Development is easier
- Servers are happier
- Users are happier
- Everyone wins

ResourceLoader



Easy to Make

happy developers

Lightweight

happy servers

WIKIMEDIA 

- Development is easier
- Servers are happier
- Users are happier
- Everyone wins

ResourceLoader



Easy to Make

happy developers



Lightweight

happy servers



Snappy Pages

happy people

WIKIMEDIA 

- Development is easier
- Servers are happier
- Users are happier
- Everyone wins

Thanks!

<http://wikitech.wikimedia.org/view/Presentations>

<http://www.mediawiki.org/wiki/ResourceLoader>

Trevor Parscal - @trevorparscal
Roan Kattouw - @catrope

● OSCON 2011

WIKIMEDIA 

- Download our presentation
- Download our code
- Build on our techniques
- Make the web a better place

Work @ Wikimedia



<http://jobs.wikimedia.org>

● OSCON 2011

WIKIMEDIA 

- Point to people they can talk to
- Tech-related and non-tech jobs too
- GO BACK TO PREVIOUS SLIDE NOW PLZ!