HTTP Caching

SensioLabs

Introduction to HTTP caching

HTTP Caching

The best way to improve the **performance** of an application is probably to **cache its output** and bypass it altogether.

HTTP Caching

Of course, this is very difficult for **highly dynamic websites**!

HTTP Caching

Symfony cache system relies on the simplicity and power of the **HTTP cache** as defined in the HTTP specification (RFC2616).

HTTP Caching

Basically, if you already know **HTTP validation** and **expiration** caching models, you are ready to use most of the Symfony caching layer.

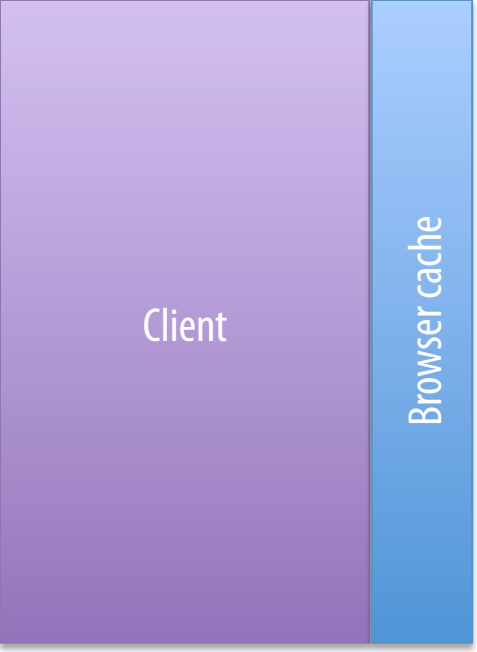
HTTP Caching - Types of caches

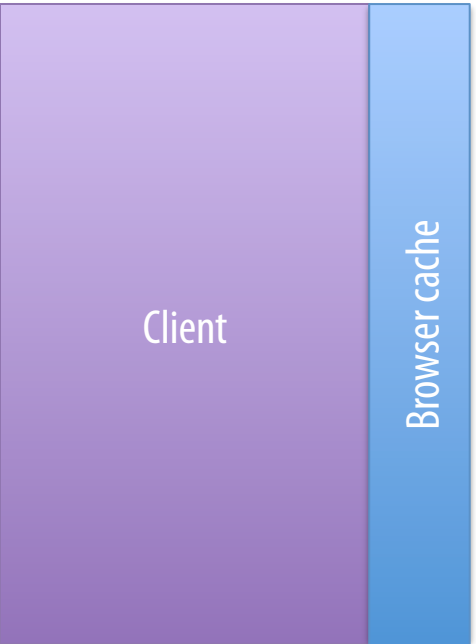
- **Browser caches**: Every browser comes with its own local cache that is mainly useful for when you hit "back" or when images are reused throughout a website;

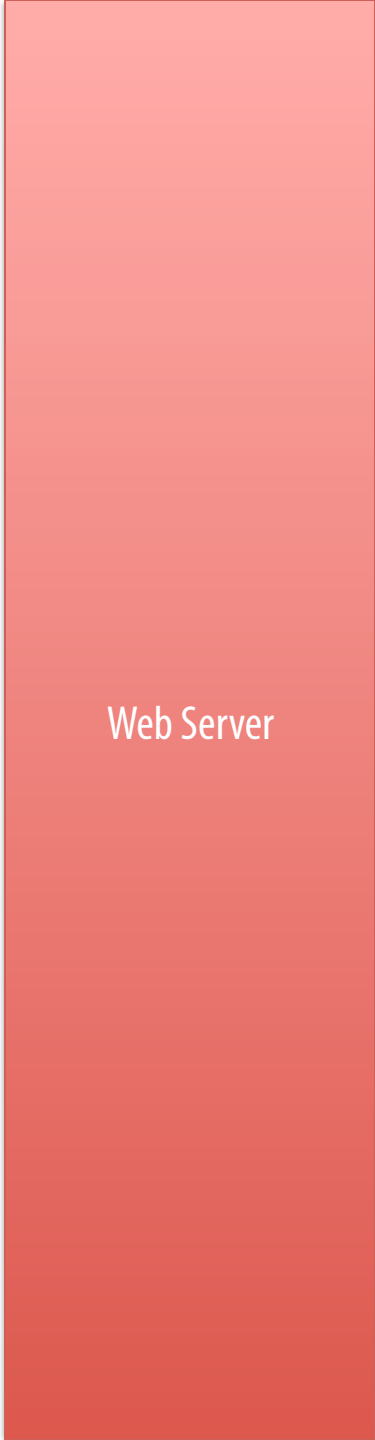
- **Proxy caches**: A proxy is a shared cache as many people can be behind a single one. It's usually installed by large corporations and ISPs to reduce latency and network traffic.

- **Gateway caches**: Like a proxy, it's also a shared cache but on the server side. Installed by network administrators, it makes websites more scalable, reliable and performing better (CDNs like Akamaï are gateway caches).

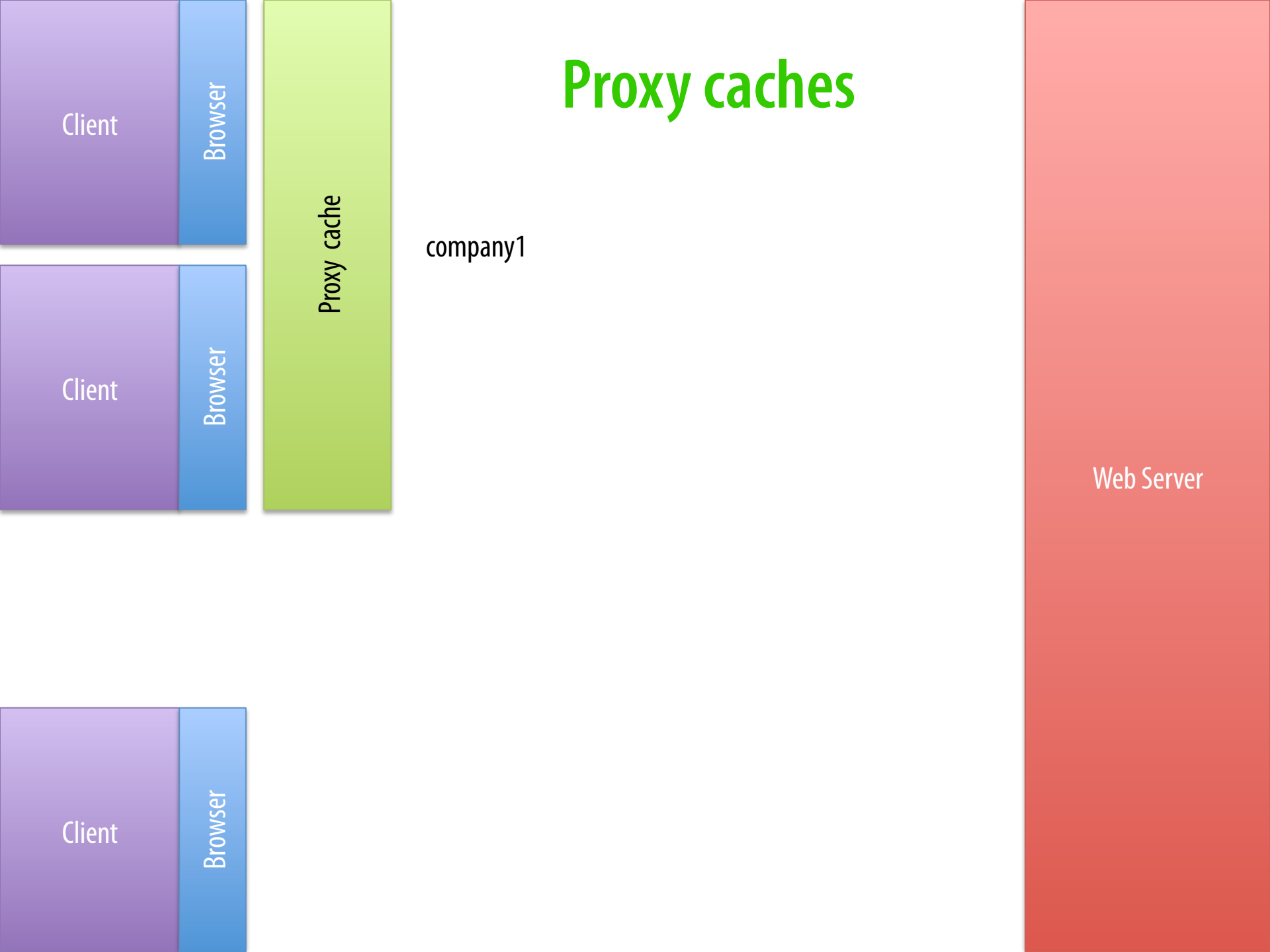
Browser caches



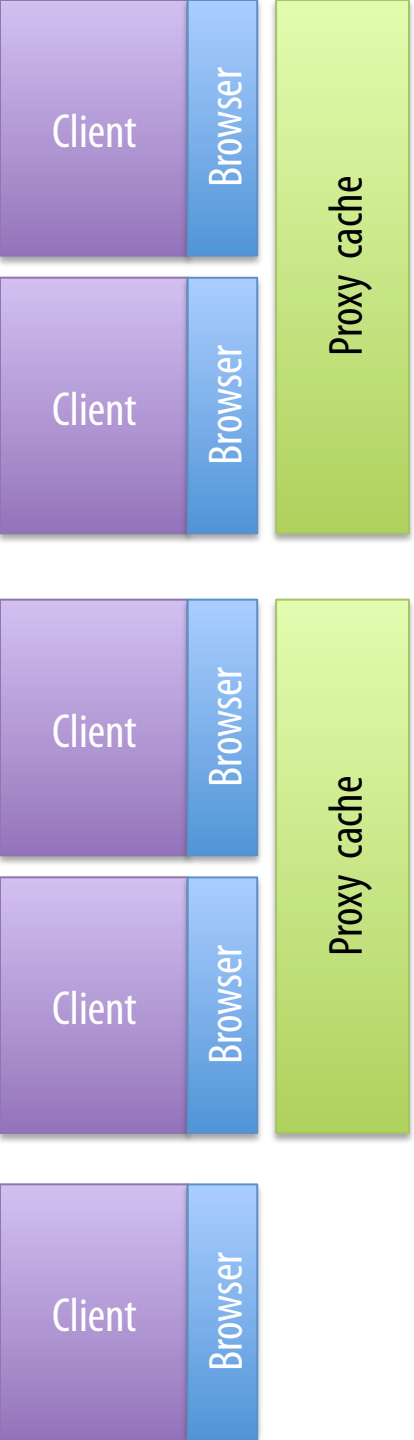


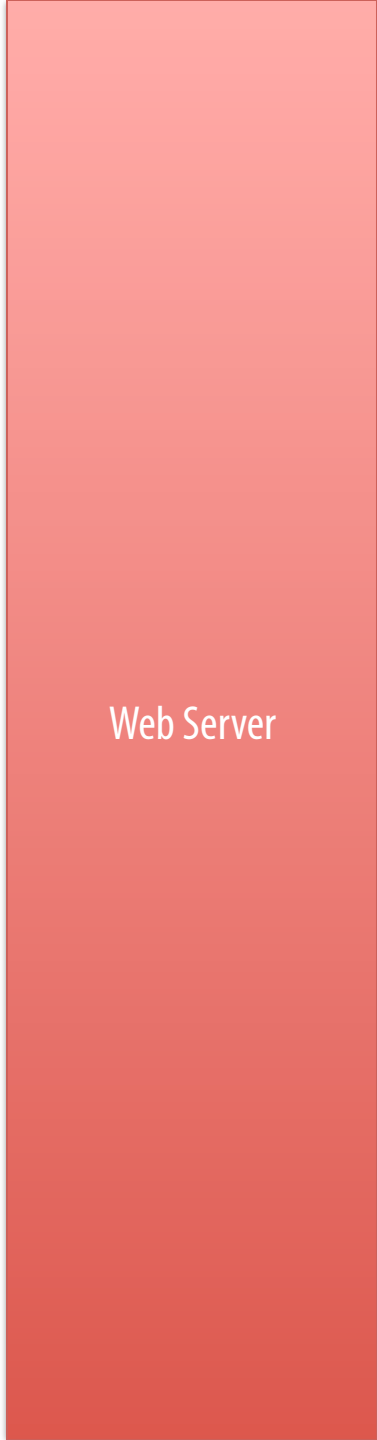


Proxy caches

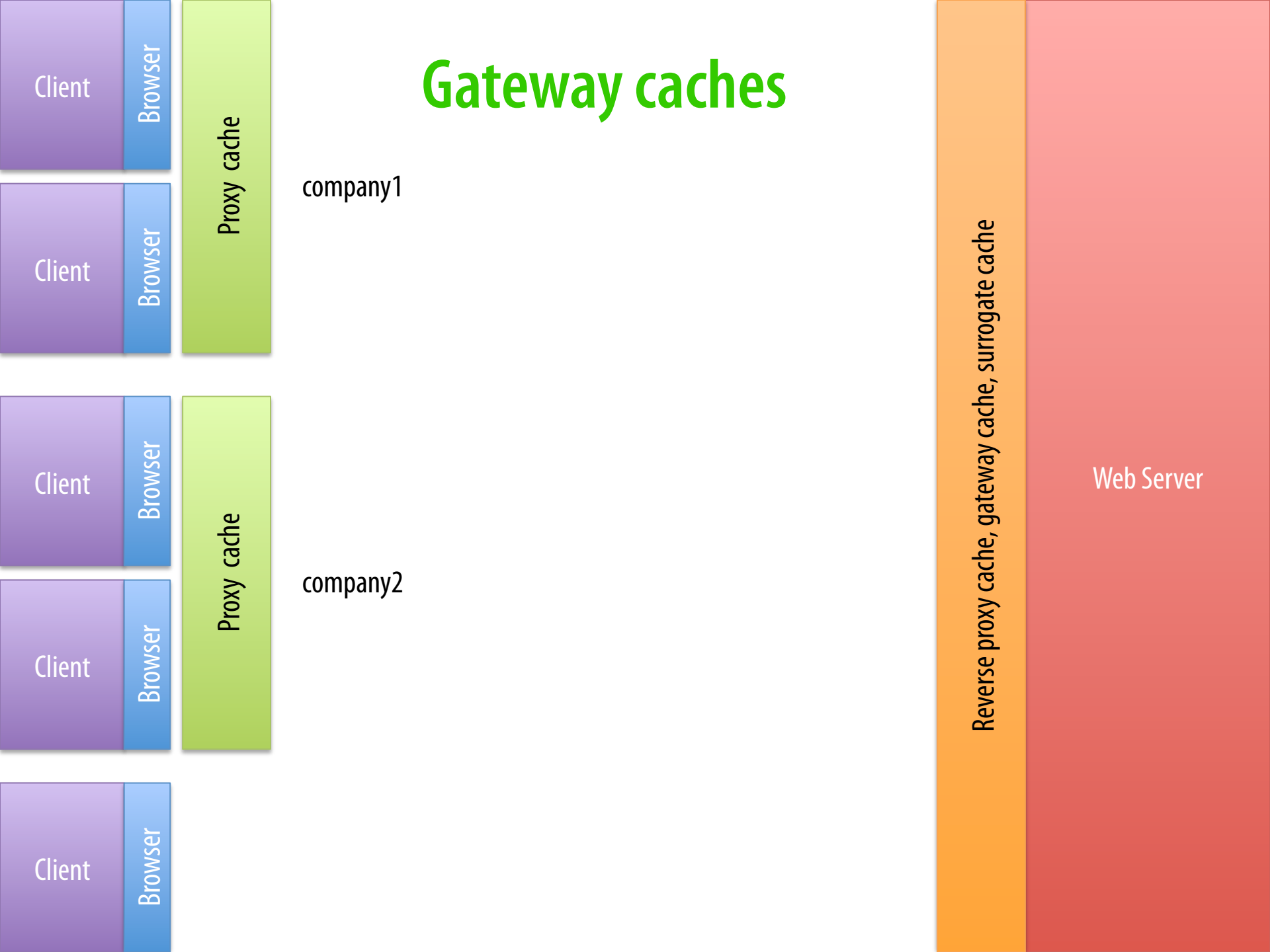


Proxy caches





Gateway caches



What can be cahced?

Only « safe » methods like **GET** and **HEAD** can be cached as they don’t change the state of the resource.

Don’t expect to cache a resource accessible from a **PUT**, **DELETE** or **POST** method.

HTTP Caching strategies

HTTP cache expiration strategy

The goal is to specify how long a **response** should be **considered** « **fresh** » by including a **Cache-Control** and/or an **Expires** header.

Caches that understand expiration will not make the same request until the cached version reaches its **expiration time** and becomes « **stale** ».

Advantage: saves some CPU resources

HTTP cache validation strategy

When some pages are really dynamic, the validation model uses a **unique identifier** and/or a **timestamp** to check if the page changed since the last request.

Identifiers are defined with the **Etag** header whereas timestamps are defined with the **Last-Modified** response header field.

Advantage: reduces bandwidth usage

HTTP Caching

The goal of both models is to **never generate the same response twice**.

HTTP Cache Specification

- **RFC 2616** is being rewritten for clarity.
- **URL:** http://tools.ietf.org/wg/httpbis/
- **Sections to read:**
  - **P4** – Conditional Requests
  - **P6** – Caching: browser and intermediary caches

Default Caching Strategy in Symfony

By default, Symfony asks browsers to not cache the page at all (« **no-cache** »).

× Headers Preview Response Timing

▶ General

▼ Response Headers view source

Cache-Control: no-cache
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
Date: Tue, 22 Mar 2016 11:56:29 GMT
Keep-Alive: timeout=5, max=200
Server: Apache/2.4.16 (Unix) PHP/5.6.14
Transfer-Encoding: chunked
X-Debug-Token: 10c577
X-Debug-Token-Link: /app_dev.php/_profiler/10c577
X-Powered-By: PHP/5.6.14

▼ Request Headers view source

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en,es;q=0.8,eu;q=0.6
Cache-Control: max-age=0
Connection: keep-alive
DNT: 1
Host: symfony-demo.dev
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2623.87 Safari/537.36

Expiration model

Expires HTTP Header field

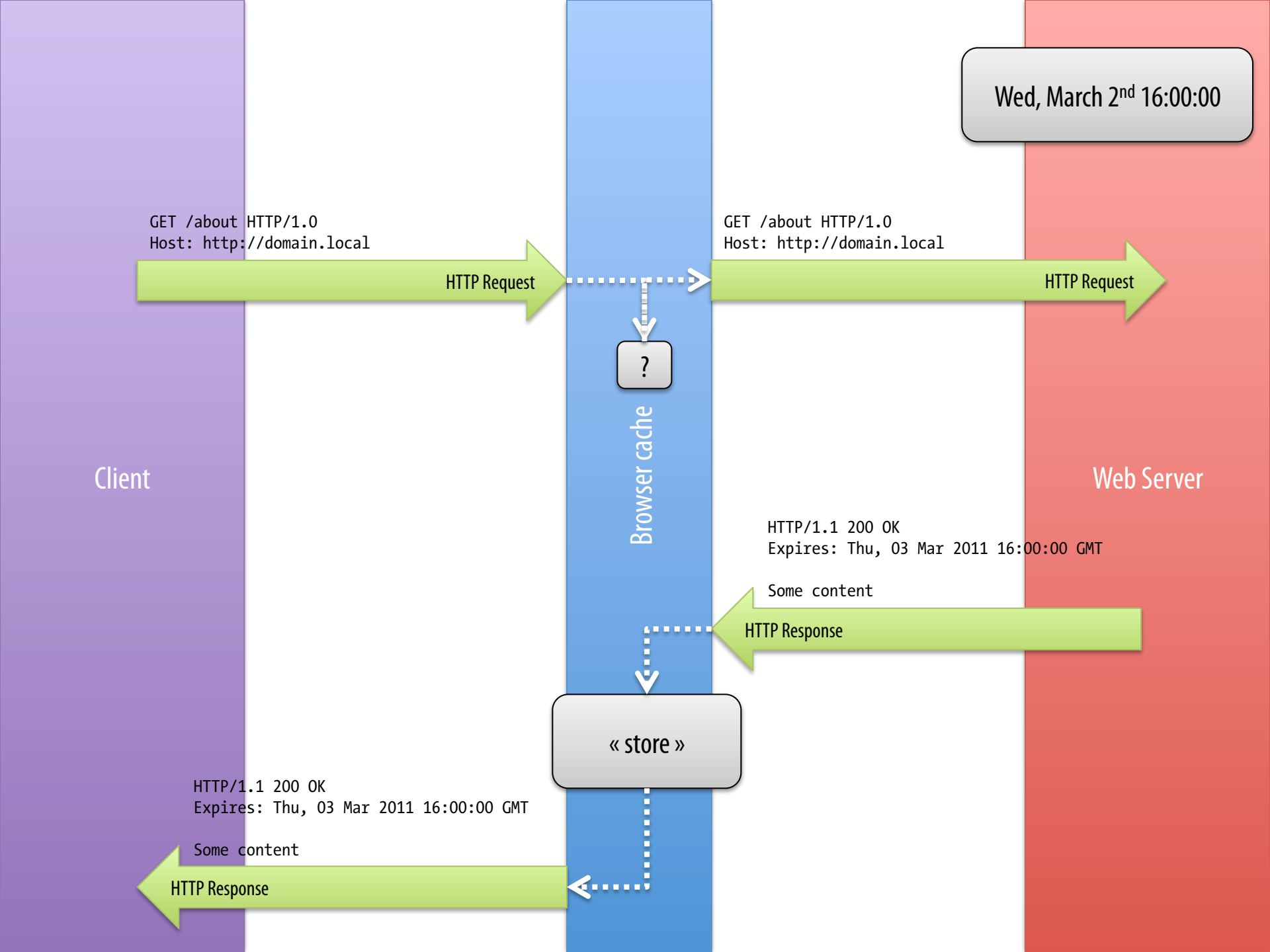
« The **Expires** header gives the date/time after which the response is considered **stale** ».

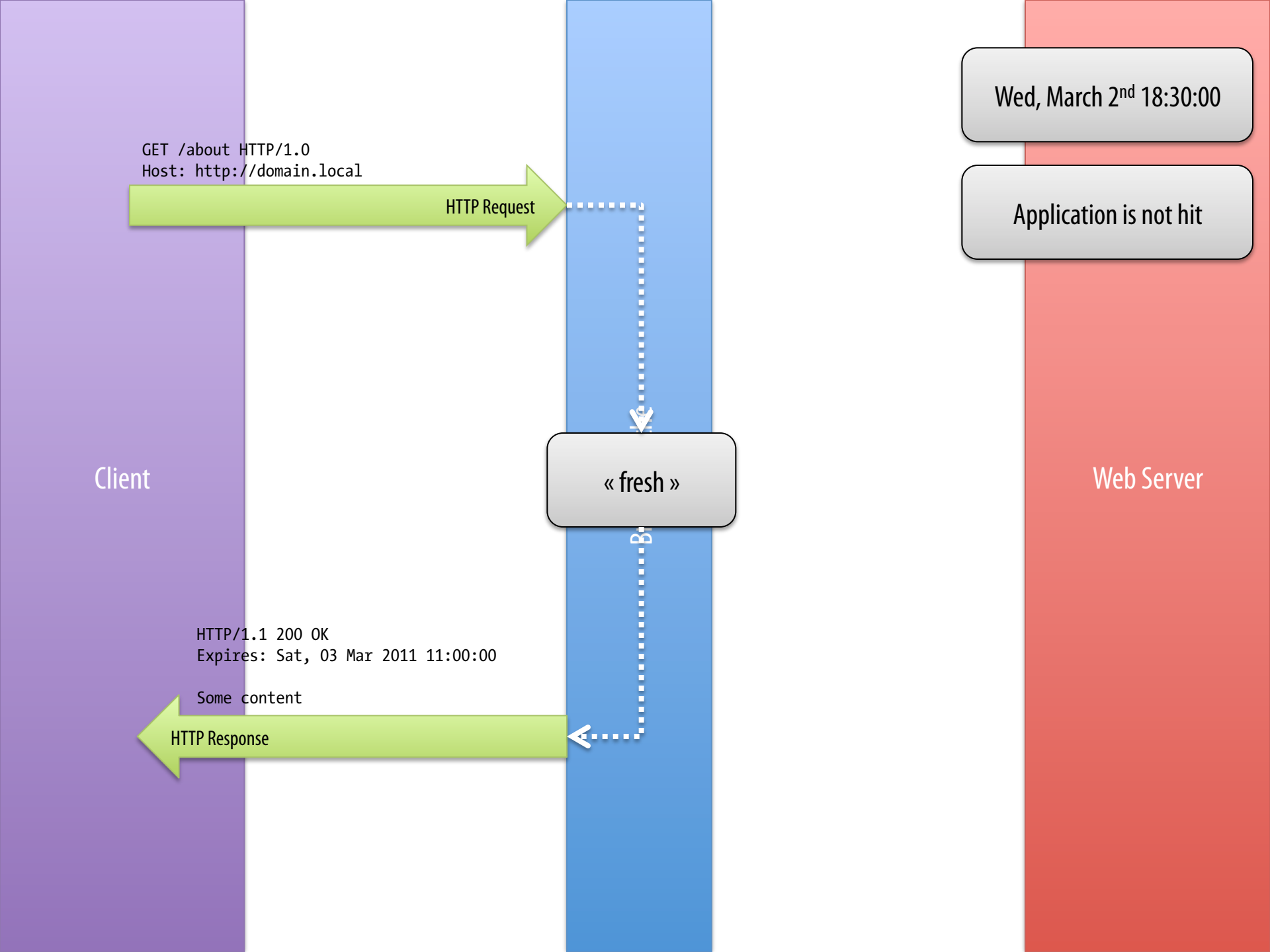
```
$response->setExpires(
  new \DateTime('+600 seconds')
);
```

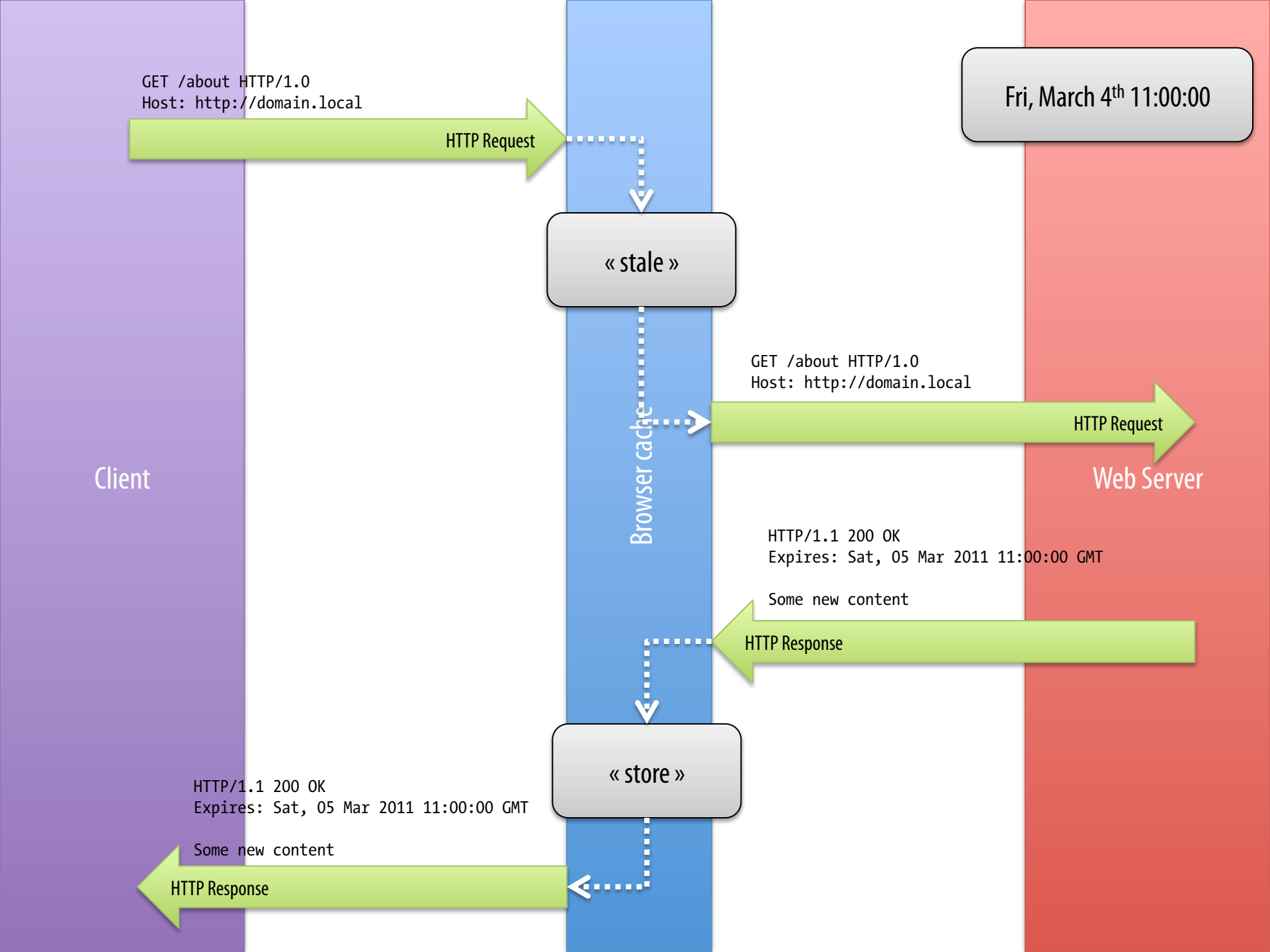
The expiration date must be a valid **GMT date** fomat.

Expires HTTP Header field

```
/**
 * @Route("/about", name="app_about")
 * @Cache(expires="+6 hours")
 */
public function aboutAction()
{
    // ...
}
```


Wed, March 2nd 16:00:00
GET /about HTTP/1.0
Host: http://domain.local
HTTP Request
GET /about HTTP/1.0
Host: http://domain.local
HTTP Request
?
Client
Browser cache
Web Server
HTTP/1.1 200 OK
Expires: Thu, 03 Mar 2011 16:00:00 GMT
Some content
HTTP Response
« store »
HTTP/1.1 200 OK
Expires: Thu, 03 Mar 2011 16:00:00 GMT
Some content
HTTP Response


Wed, March 2nd 18:30:00
Application is not hit
GET /about HTTP/1.0
Host: http://domain.local
HTTP Request
Client
« fresh »
Web Server
HTTP/1.1 200 OK
Expires: Sat, 03 Mar 2011 11:00:00
Some content
HTTP Response


GET /about HTTP/1.0
Host: http://domain.local
HTTP Request
Fri, March 4th 11:00:00
« stale »
GET /about HTTP/1.0
Host: http://domain.local
HTTP Request
Client
Browser cache
Web Server
HTTP/1.1 200 OK
Expires: Sat, 05 Mar 2011 11:00:00 GMT
Some new content
HTTP Response
« store »
HTTP/1.1 200 OK
Expires: Sat, 05 Mar 2011 11:00:00 GMT
Some new content
HTTP Response

Expires HTTP Header field

- **First limitation:** Both clocks of the browser and the server must be synchronized.

- **Second limitation:** The specification states that servers should not send « Expires » dates more than one year in the future.

Cache-Control HTTP Header field

The HTTP 1.1 protocol introduces the « **Cache-Control** » header field that is responsible to define caching strategy by specifying several directives.

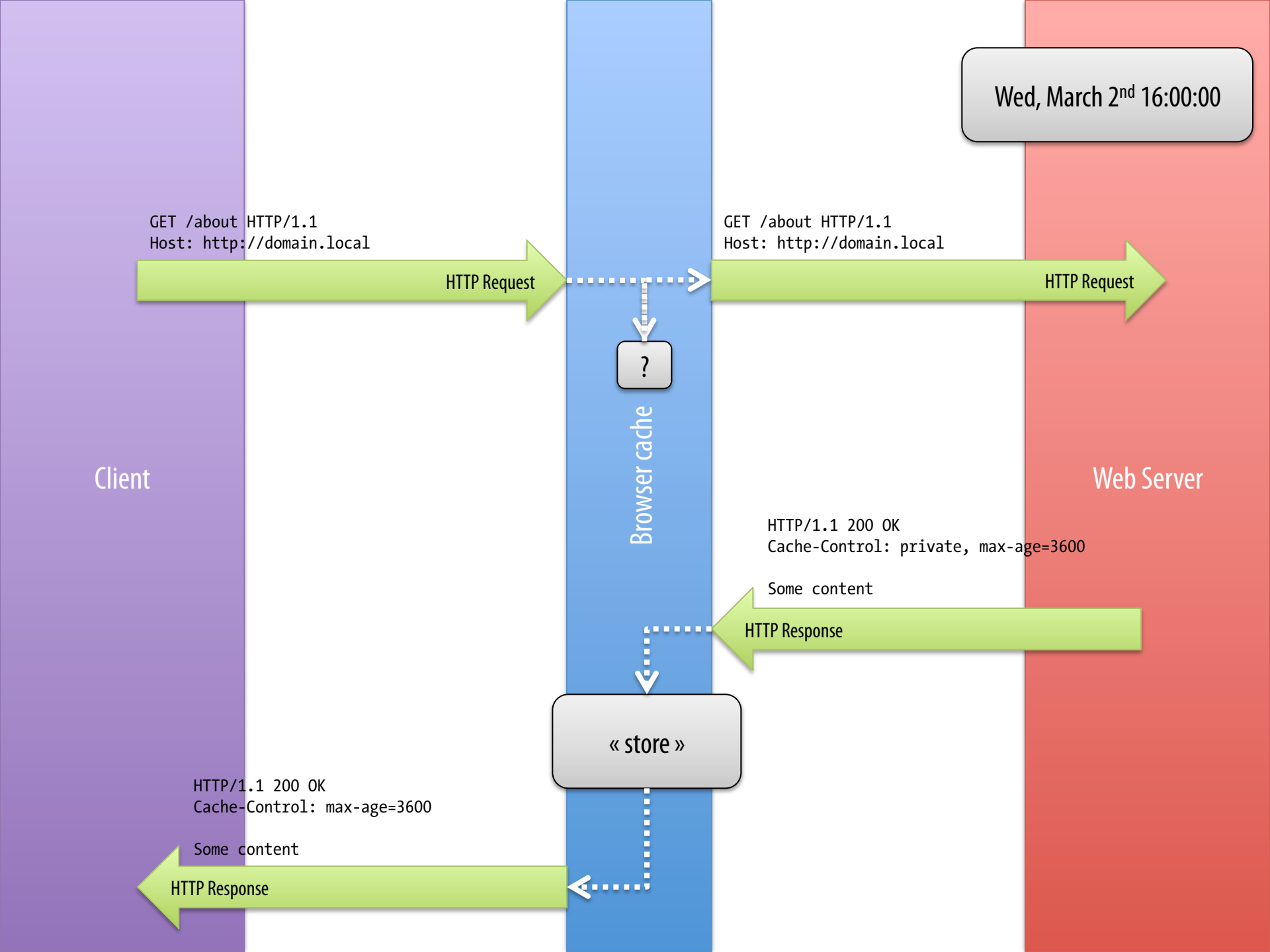
For expiration, there are « **max-age** » and « **s-maxage** » directives that consider a resource « fresh » for a number of seconds since the date/time the response was generated.

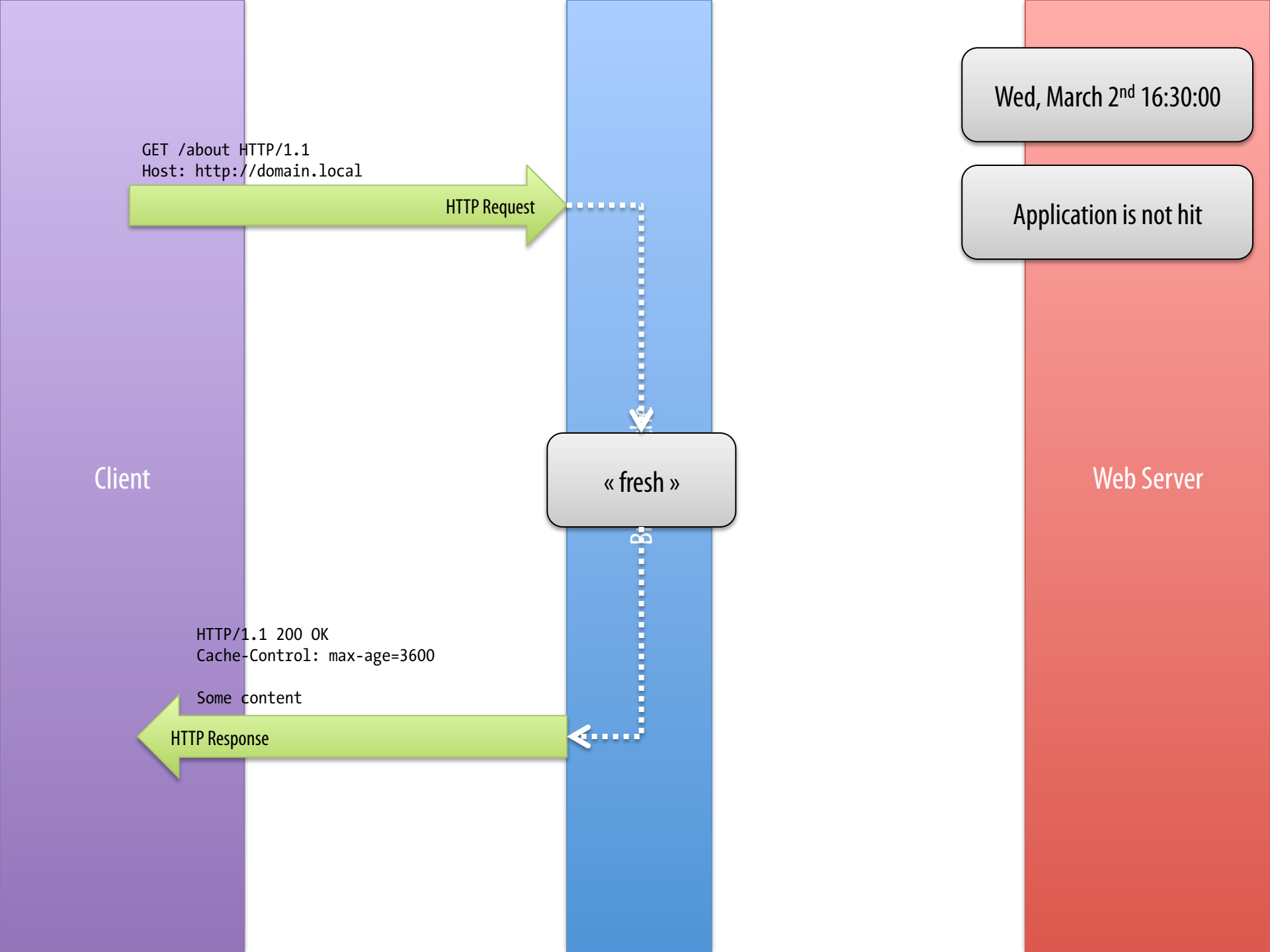
```
Cache-Control: private, max-age=60
```

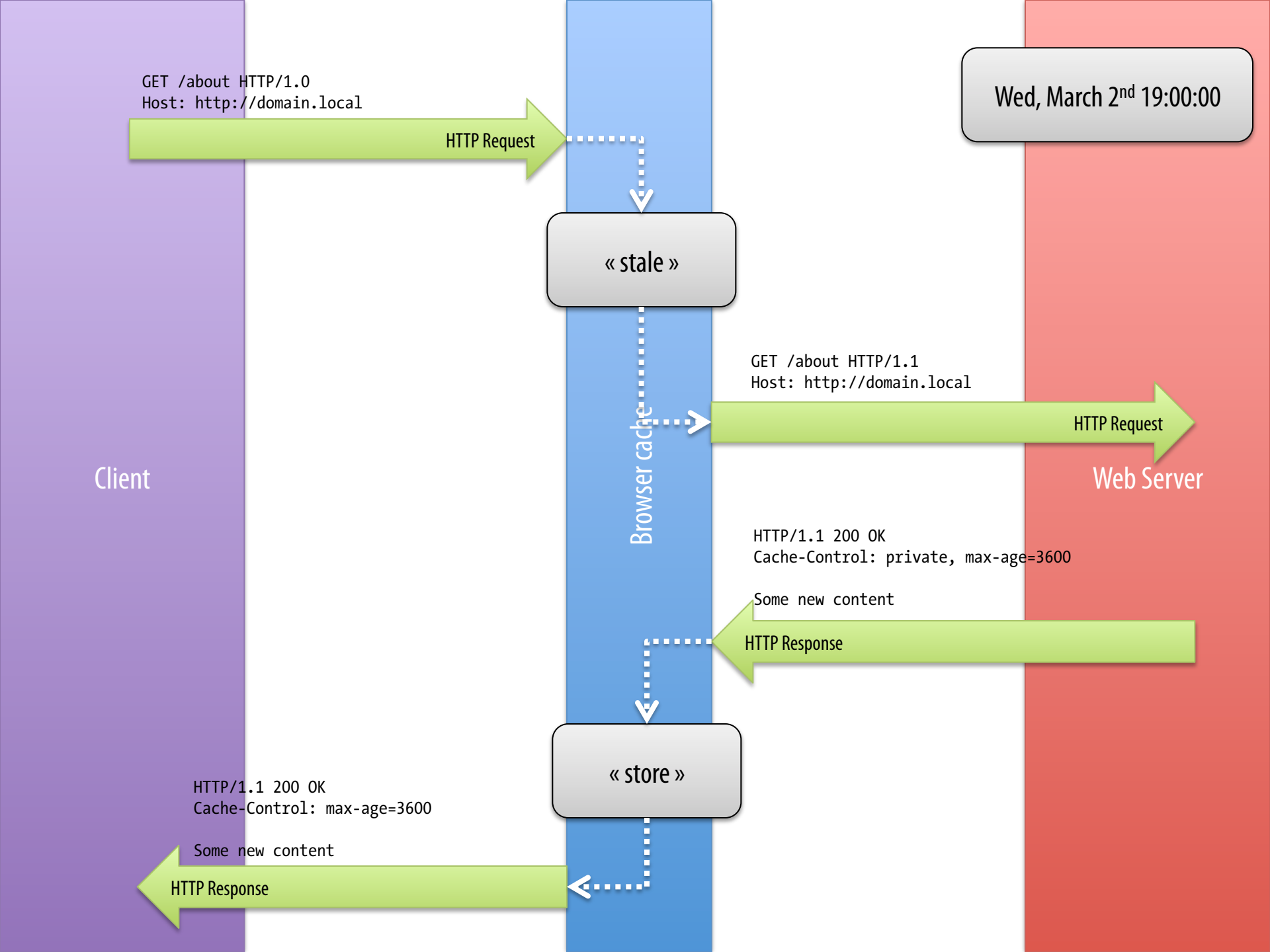
Browser-side and server-side caching

```
// Browser-side caching
$response->setMaxAge(600);

/**
 * @Cache(maxage=600)
 * ...
 */
```

```
// Server-side caching
$response->setSharedMaxAge(600);

/**
 * @Cache(smaxage=600)
 * ...
 */
```


Wed, March 2nd 16:00:00
GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
?
Client
Browser cache
Web Server
HTTP/1.1 200 OK
Cache-Control: private, max-age=3600
Some content
HTTP Response
« store »
HTTP/1.1 200 OK
Cache-Control: max-age=3600
Some content
HTTP Response


Wed, March 2nd 16:30:00
Application is not hit
GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
« fresh »
Client
Web Server
HTTP/1.1 200 OK
Cache-Control: max-age=3600
Some content
HTTP Response


GET /about HTTP/1.0
Host: http://domain.local
HTTP Request
Wed, March 2nd 19:00:00
« stale »
Browser cache
GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
Client
Web Server
HTTP/1.1 200 OK
Cache-Control: private, max-age=3600
Some new content
HTTP Response
« store »
HTTP/1.1 200 OK
Cache-Control: max-age=3600
Some new content
HTTP Response

Validation model

Validation in Practice

200

HTTP **OK**

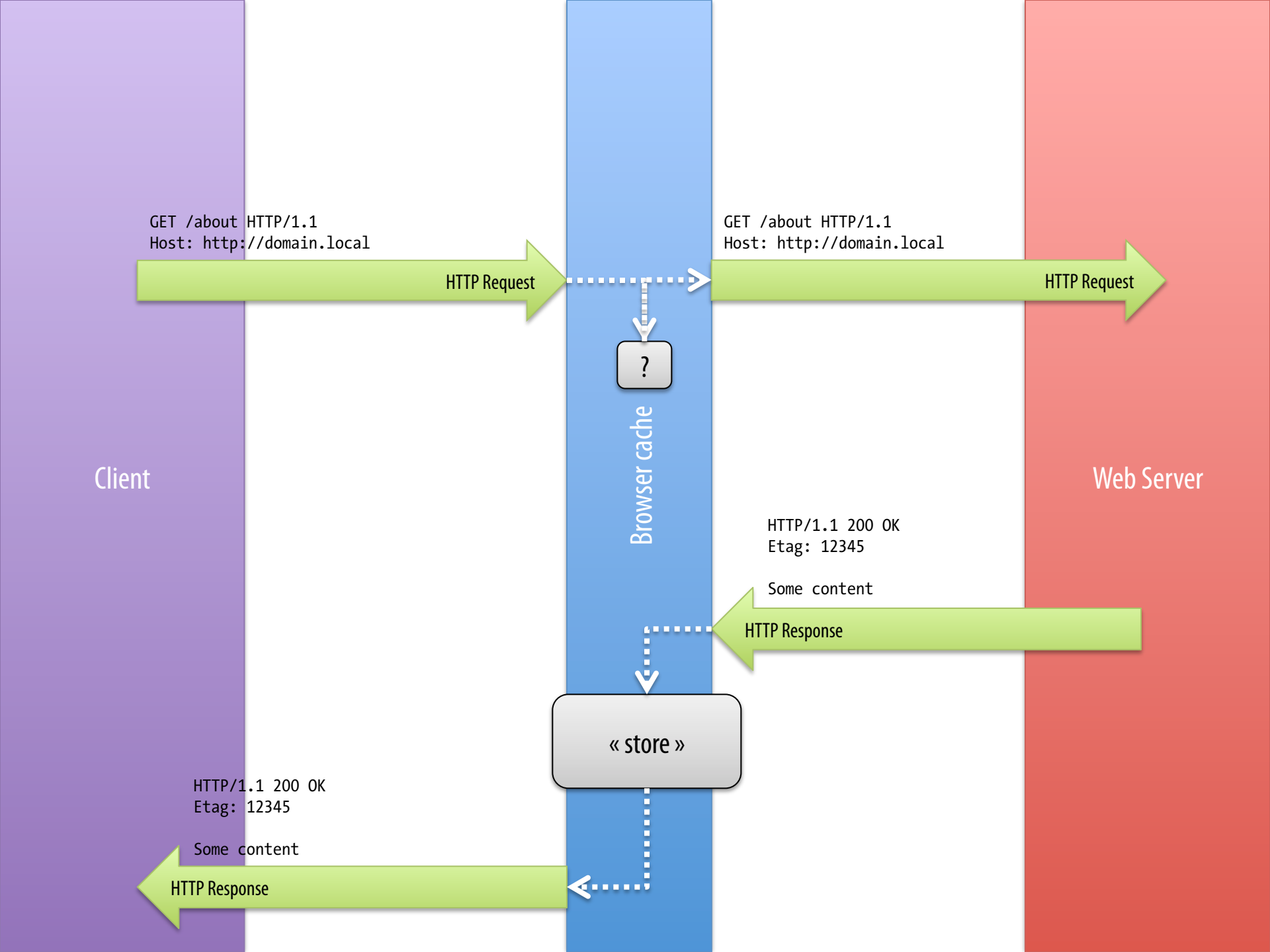
304

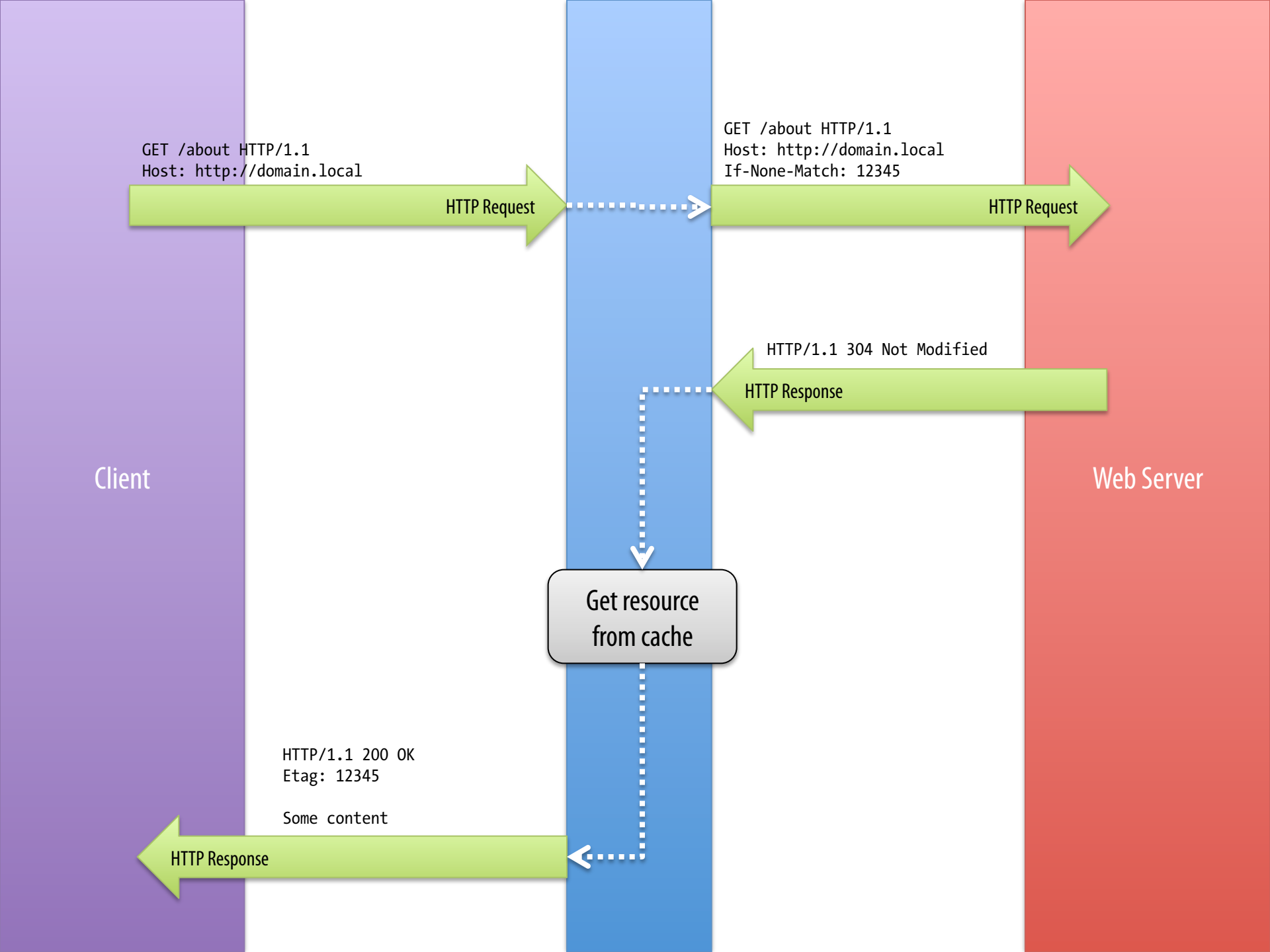
HTTP **NOT MODIFIED**

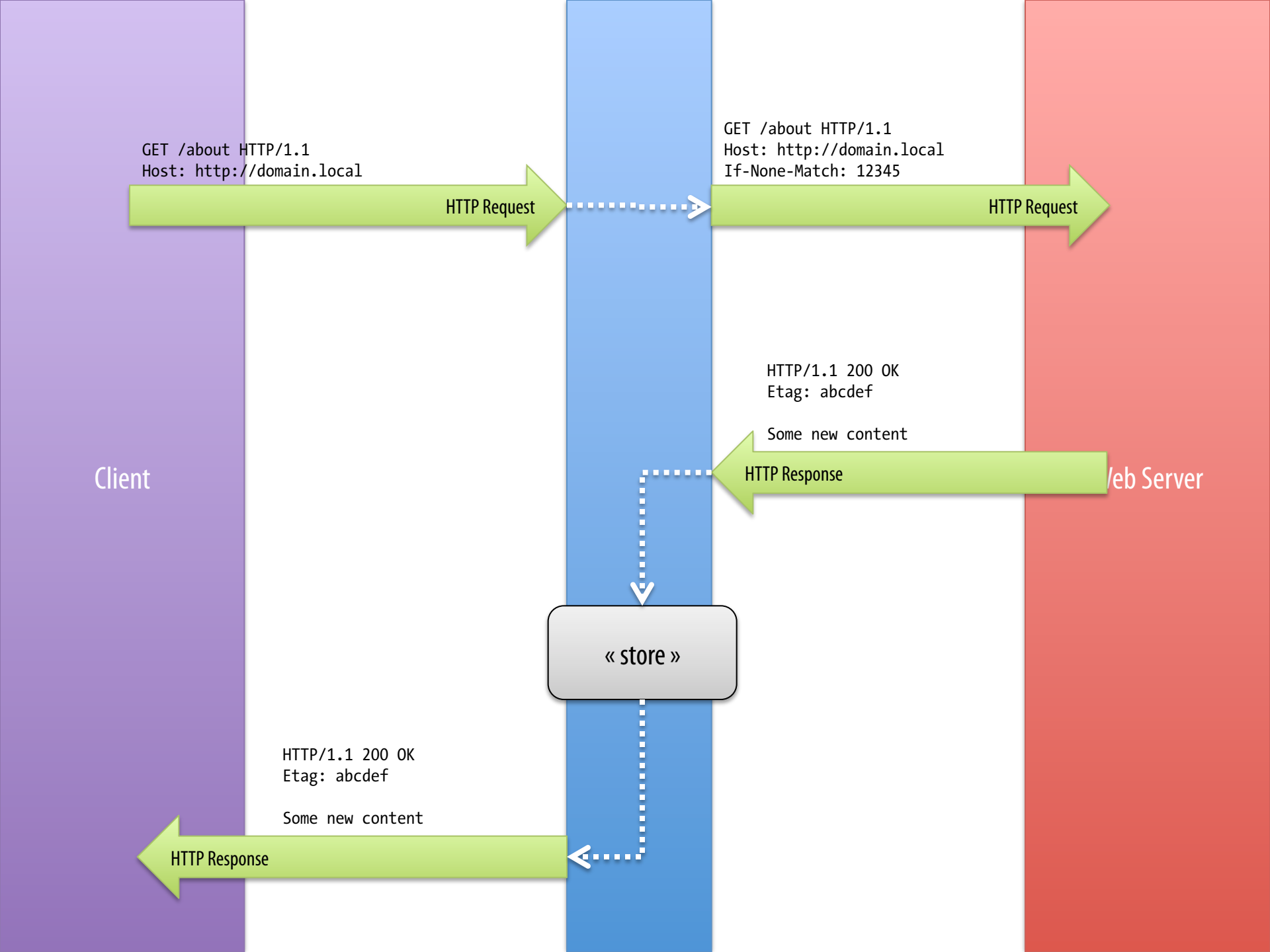
Etag HTTP Header Field

The « **Etag** » response HTTP header field provides the current value of the entity-tag for one representation of the target resource.

```
$response->setETag('abc123456def');
```


GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
?
Client
Browser cache
Web Server
HTTP/1.1 200 OK
Etag: 12345
Some content
HTTP Response
« store »
HTTP/1.1 200 OK
Etag: 12345
Some content
HTTP Response

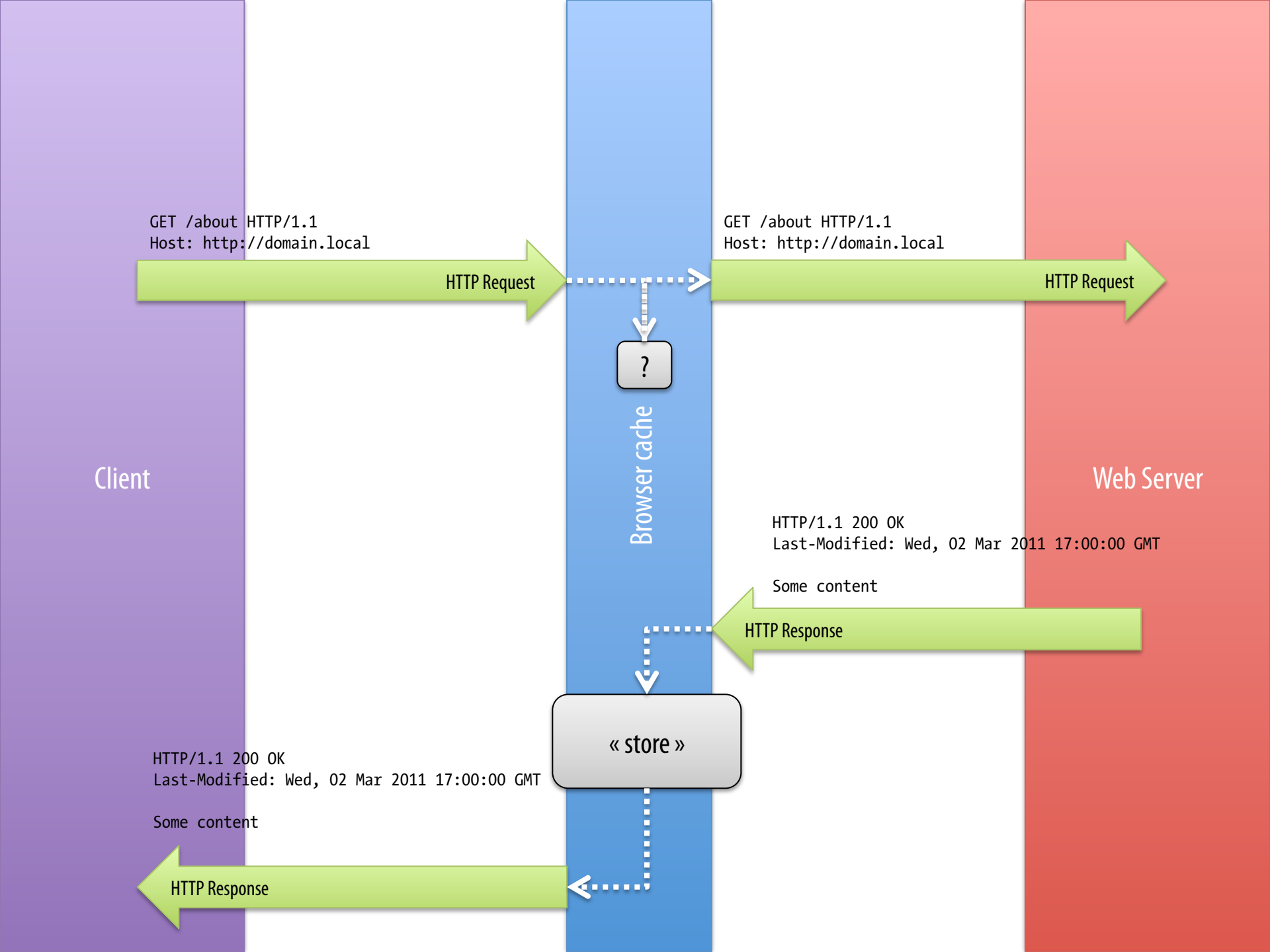

GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
GET /about HTTP/1.1
Host: http://domain.local
If-None-Match: 12345
HTTP Request
HTTP/1.1 304 Not Modified
HTTP Response
Client
Web Server
Get resource from cache
HTTP/1.1 200 OK
Etag: 12345
Some content
HTTP Response

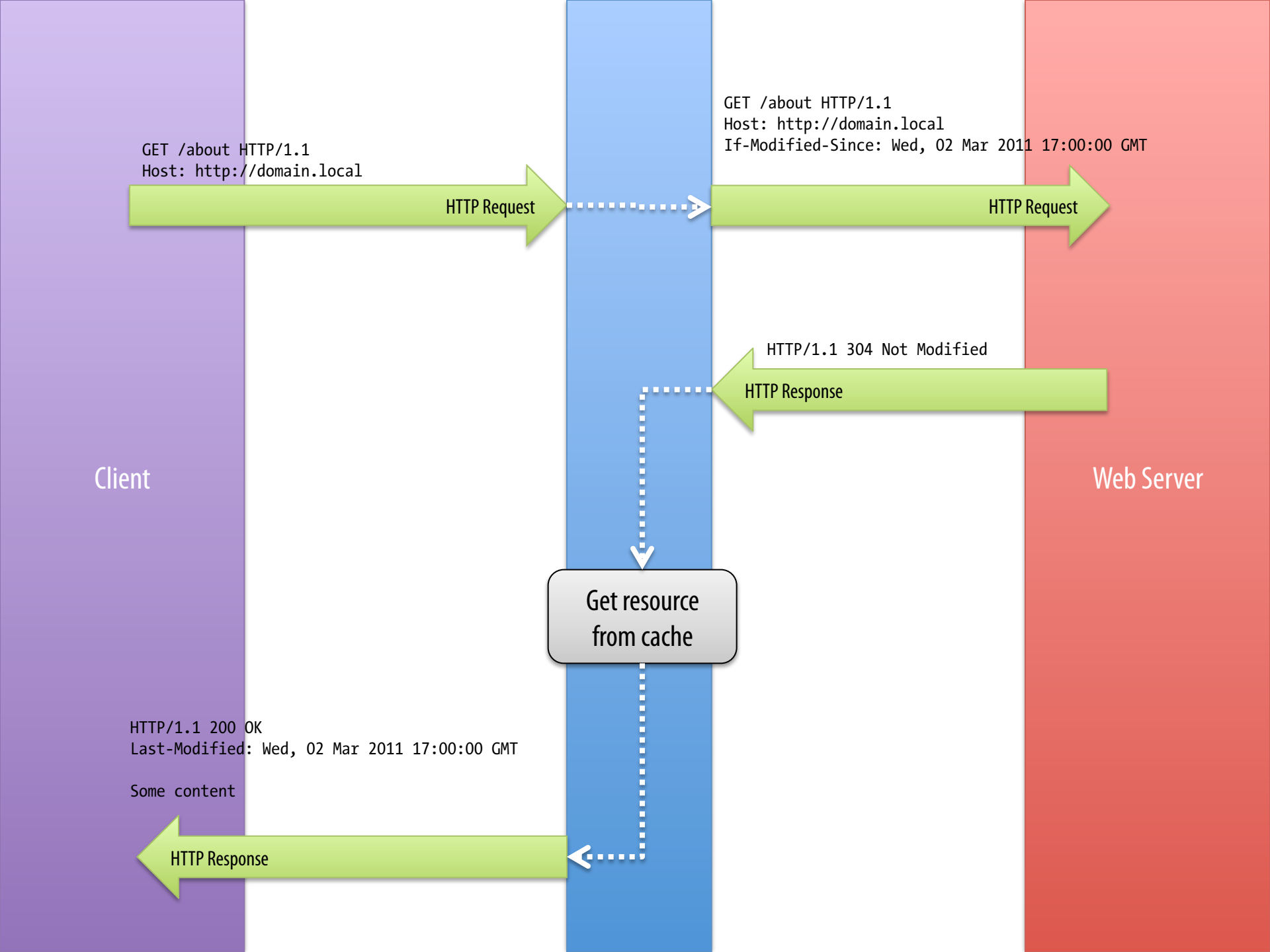

GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
GET /about HTTP/1.1
Host: http://domain.local
If-None-Match: 12345
HTTP Request
HTTP/1.1 200 OK
Etag: abcdef
Some new content
HTTP Response
Client
Web Server
« store »
HTTP/1.1 200 OK
Etag: abcdef
Some new content
HTTP Response

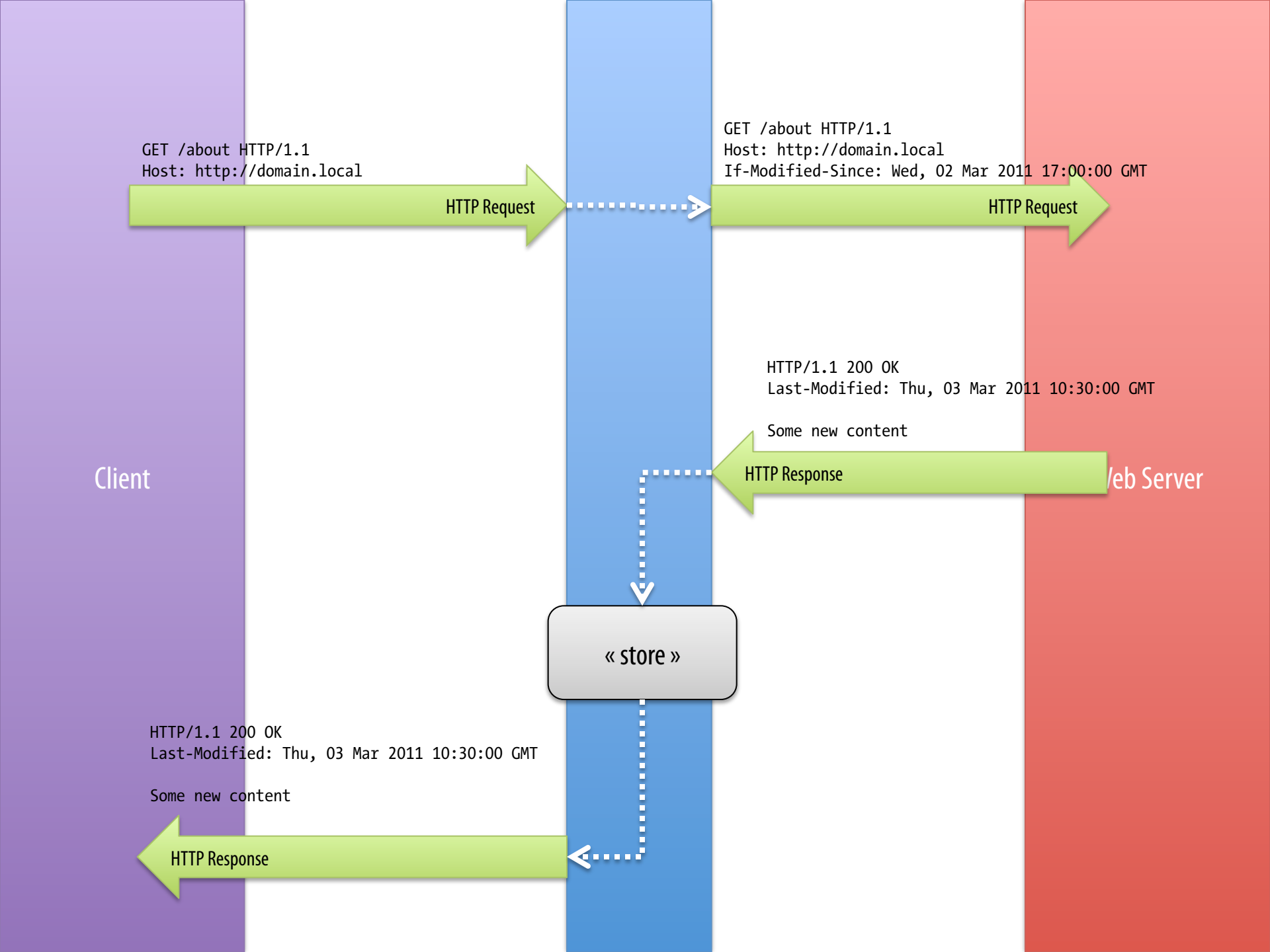
Last-Modified HTTP Header Field

The « **Last-Modified** response HTTP header field indicates the date and time at which the origin server believes the representation was last modified ».

```
$article = ArticleDAO::findById($id);
$response->setLastModified($article->getUpdatedAt());
```


GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
?
Client
Browser cache
Web Server
HTTP/1.1 200 OK
Last-Modified: Wed, 02 Mar 2011 17:00:00 GMT
Some content
HTTP Response
« store »
HTTP/1.1 200 OK
Last-Modified: Wed, 02 Mar 2011 17:00:00 GMT
Some content
HTTP Response


GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
GET /about HTTP/1.1
Host: http://domain.local
If-Modified-Since: Wed, 02 Mar 2011 17:00:00 GMT
HTTP Request
HTTP/1.1 304 Not Modified
HTTP Response
Client
Web Server
Get resource from cache
HTTP/1.1 200 OK
Last-Modified: Wed, 02 Mar 2011 17:00:00 GMT
Some content
HTTP Response


GET /about HTTP/1.1
Host: http://domain.local
HTTP Request
GET /about HTTP/1.1
Host: http://domain.local
If-Modified-Since: Wed, 02 Mar 2011 17:00:00 GMT
HTTP Request
HTTP/1.1 200 OK
Last-Modified: Thu, 03 Mar 2011 10:30:00 GMT
Some new content
HTTP Response
Client
eb Server
« store »
HTTP/1.1 200 OK
Last-Modified: Thu, 03 Mar 2011 10:30:00 GMT
Some new content
HTTP Response

How to return 304 or 200?

```
$article = ArticleDAO::findOneById($id);
$response = new Response();
$response->setLastModified($article->getUpdatedAt());

if ($response->isNotModified($request)) {
    return $response;
}

return $this->render(
    'blog/post.html.twig',
    ['blog' => $blog],
    $response
);
```

Expiration & Validation Together

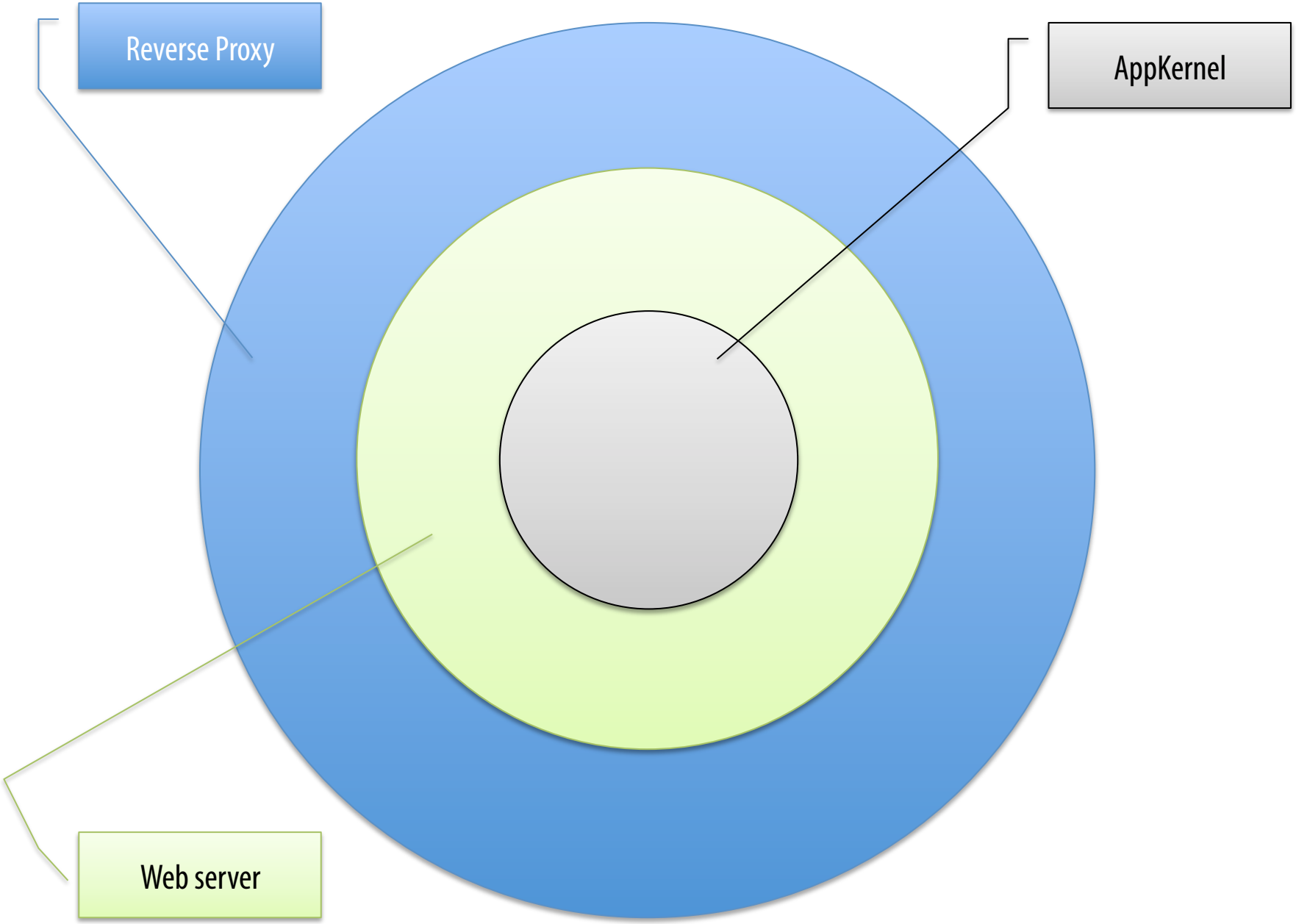
Both « expiration and validation » models can live together but it's important to remember that **expiration wins over validation**.

```
// Set cache settings in one call
$response->setCache([
    'etag'    => $etag,
    'max_age' => 10,
    'public'  => false
]);
```

Reverse Proxy Caches

What is a reverse proxy cache

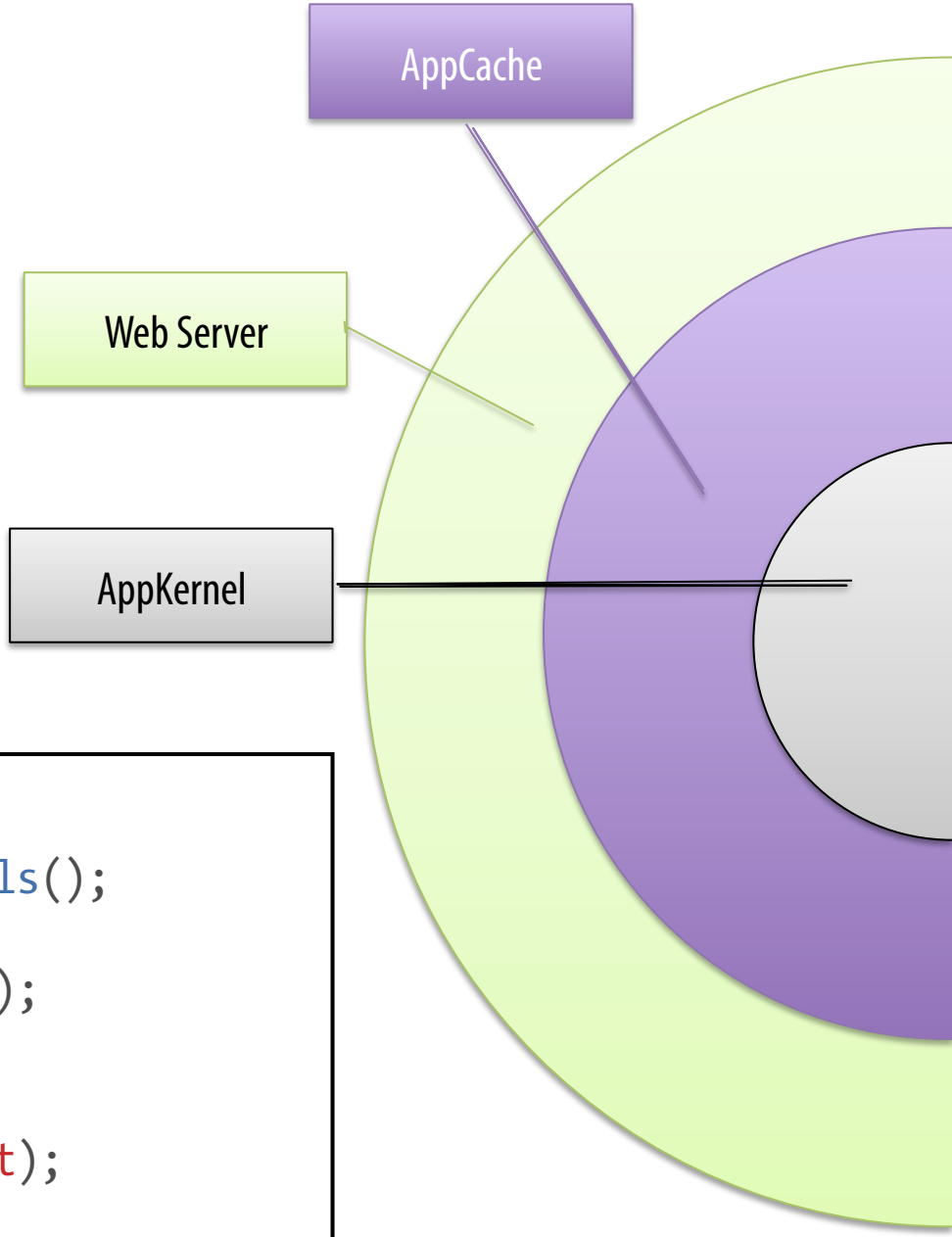
Reverse-Proxy cache sits in front of the application between the client and the web server.


Reverse Proxy
AppKernel
Web server

Open-Source Reverse Proxies

- Varnish Cache
  varnish-cache.org
- Squid Cache
  squid-cache.org
- HAProxy
  haproxy.org
- Symfony's PHP Reverse Proxy
  symfony.com/doc/current/book/http_cache.html

Enabling the Symfony reverse proxy is as simple as wrapping the AppKernel object with an AppCache instance in the front controller PHP script.




```
# web/app_dev.php
$request = Request::createFromGlobals();

$kernel = new AppKernel('dev', true);
$kernel = new AppCache($kernel);

$response = $kernel->handle($request);
$response->send();
```

Caching a response on the server side

The generated **Response** must include an **s-maxage Cache-Control** directive and must be **public**.

```
/**
 * @Route("/about", name="about")
 * @Cache(smaxage=120)
 */
public function aboutAction()
{
    // ...
}
```

Cache Debugging : page is not cached yet

The **x-symfony-cache** debug header field indicates the page is not yet cached (**miss**), so the reverse proxy caches it (**store**).

Console HTML CSS Script DOM Réseau Cookies Illuminations YSlow

Effacer Persistant Tous HTML CSS JS XHR Images Flash Média

URL	Statut	Domaine	Poids	Chronologie
GET about	200 OK	sandbox.local	14 B	

En-têtes Réponse HTML Cookies

Réponse voir le code source

Date	Fri, 01 Jul 2011 15:22:33 GMT
Server	Apache/2.2.19 (Unix) mod_ssl/2.2.19 OpenSSL/1.0.0d DAV/2 SVN/1.6.17 PHP/5.3.6
X-Powered-By	PHP/5.3.6
Cache-Control	public, s-maxage=120
x-wf-protocol-1	http://meta.wildfirehq.org/Protocol/JsonStream/0.2
x-wf-1-structure-1	http://meta.firephp.org/Wildfire/Structure/FirePHP/FirebugConsole/0.1
x-wf-1-plugin-1	http://meta.firephp.org/Wildfire/Plugin/FirePHP/Library-FirePHPCore/0.3
x-wf-1-1-1-1	260|[{"Type":"INFO","File":"","Line":"","Label":"request"},"Matched route \"sensio_training_default_index \" (parameters: \"_controller\": \"Sensio\\TrainingBundle\\Controller\\DefaultController::indexAction \", \"_route\": \"sensio_training_default_index\") [] []"]|
x-debug-token	4e0de6396e6e7
x-content-digest	en8b617517986534332e38521d59fef976e7c10bd8
Content-Length	14
Age	0
x-symfony-cache	GET /about: miss, store
Connection	close
Content-Type	text/html; charset=UTF-8

Cache Debugging : page is still valid

The **x-symfony-cache** debug header field indicates the page is still valid (**fresh**), so the reverse proxy gets it from its cache.

Console HTML CSS Script DOM Réseau Cookies Illuminations YSlow

Effacer Persistant Tous HTML CSS JS XHR Images Flash Média

URL	Statut	Domaine	Poids	Chronologie
GET about	200 OK	sandbox.local	14 B	14ms

En-têtes Réponse HTML Cookies

Réponse voir le code source

```
                Date  Fri, 01 Jul 2011 15:23:13 GMT
              Server  Apache/2.2.19 (Unix) mod_ssl/2.2.19 OpenSSL/1.0.0d DAV/2 SVN/1.6.17 PHP/5.3.6
        X-Powered-By  PHP/5.3.6
       Cache-Control  public, s-maxage=120
     x-wf-protocol-1  http://meta.wildfirehq.org/Protocol/JsonStream/0.2
  x-wf-1-structure-1  http://meta.firephp.org/Wildfire/Structure/FirePHP/FirebugConsole/0.1
     x-wf-1-plugin-1  http://meta.firephp.org/Wildfire/Plugin/FirePHP/Library-FirePHPCore/0.3
       x-wf-1-1-1-1  260|[{"Type":"INFO","File":"","Line":"","Label":"request"},"Matched route \"sensio_training_default_index
                     \" (parameters: \"_controller\": \"Sensio\\TrainingBundle\\Controller\\DefaultController::indexAction
                     \", \"_route\": \"sensio_training_default_index\") [] []"]|
      x-debug-token  4e0de6396e6e7
   x-content-digest  en8b617517986534332e38521d59fef976e7c10bd8
     Content-Length  14
                Age  40
    x-symfony-cache  GET /about: fresh
         Connection  close
       Content-Type  text/html; charset=UTF-8
```

Cache Debugging : page is stale

The **x-symfony-cache** debug header field indicates the page is no more fresh (**stale, invalid**), so the reverse proxy stores the new version in its cache (**store**).

Console HTML CSS Script DOM Réseau Cookies Illuminations YSlow

Effacer Persistant Tous HTML CSS JS XHR Images Flash Média

URL	Statut	Domaine	Poids	Chronologie
GET about	200 OK	sandbox.local	14 B	

En-têtes Réponse HTML Cookies

Réponse voir le code source

Date	Fri, 01 Jul 2011 15:25:35 GMT
Server	Apache/2.2.19 (Unix) mod_ssl/2.2.19 OpenSSL/1.0.0d DAV/2 SVN/1.6.17 PHP/5.3.6
X-Powered-By	PHP/5.3.6
Cache-Control	public, s-maxage=120
x-wf-protocol-1	http://meta.wildfirehq.org/Protocol/JsonStream/0.2
x-wf-1-structure-1	http://meta.firephp.org/Wildfire/Structure/FirePHP/FirebugConsole/0.1
x-wf-1-plugin-1	http://meta.firephp.org/Wildfire/Plugin/FirePHP/Library-FirePHPCore/0.3
x-wf-1-1-1-1	260|[{"Type":"INFO","File":"","Line":"","Label":"request"},"Matched route \"sensio_training_default_index \" (parameters: \"_controller\": \"Sensio\\TrainingBundle\\Controller\\DefaultController::indexAction \", \"_route\": \"sensio_training_default_index\") [] []"]|
x-debug-token	4e0de6efeccad
x-content-digest	en8b617517986534332e38521d59fef976e7c10bd8
Content-Length	14
Age	0
x-symfony-cache	GET /about: stale, invalid, store
Connection	close
Content-Type	text/html; charset=UTF-8

Edge Side Includes

An ESI tag

```
<esi:include
 src="http://..." />
```

Symfony

a SensioLabs product

DOWNLOAD NOW

What is Symfony? | Get started | Documentation | Marketplace | Community | Services | Blog | About

Home » Marketplace

Marketplace

Under construction...

No ESI

NEWS FROM THE BLOG

A week of symfony #241 (8->14 August 2011)
August 15, 2011

A week of symfony #240 (1->7 August 2011)
August 09, 2011

symfony 1.4.13 released
August 05, 2011

Visit Symfony's blog ›

IN THE NEWS

Symfony 2 trainings

Be the first to be trained

REGISTER TODAY

UPCOMING TRAINING SESSIONS

Getting Started with Symfony2
Köln – 2011-08-25

Getting Started with Symfony2
New York City – 2011-09-12

Mastering Symfony2
New York City – 2011-09-14

View all sessions ›

Symfony

a SensioLabs product → DOWNLOAD NOW

What is Symfony? | Get started | Documentation | Marketplace | Community | Services | Blog | About

Home » Marketplace

Marketplace

Under construction...

With ESI

NEWS FROM THE BLOG

A week of symfony #241 (8->14 August 2011)
August 15, 2011

A week of symfony #240 (1->7 August 2011)
August 09, 2011

symfony 1.4.13 released
August 05, 2011

Visit Symfony's blog ›

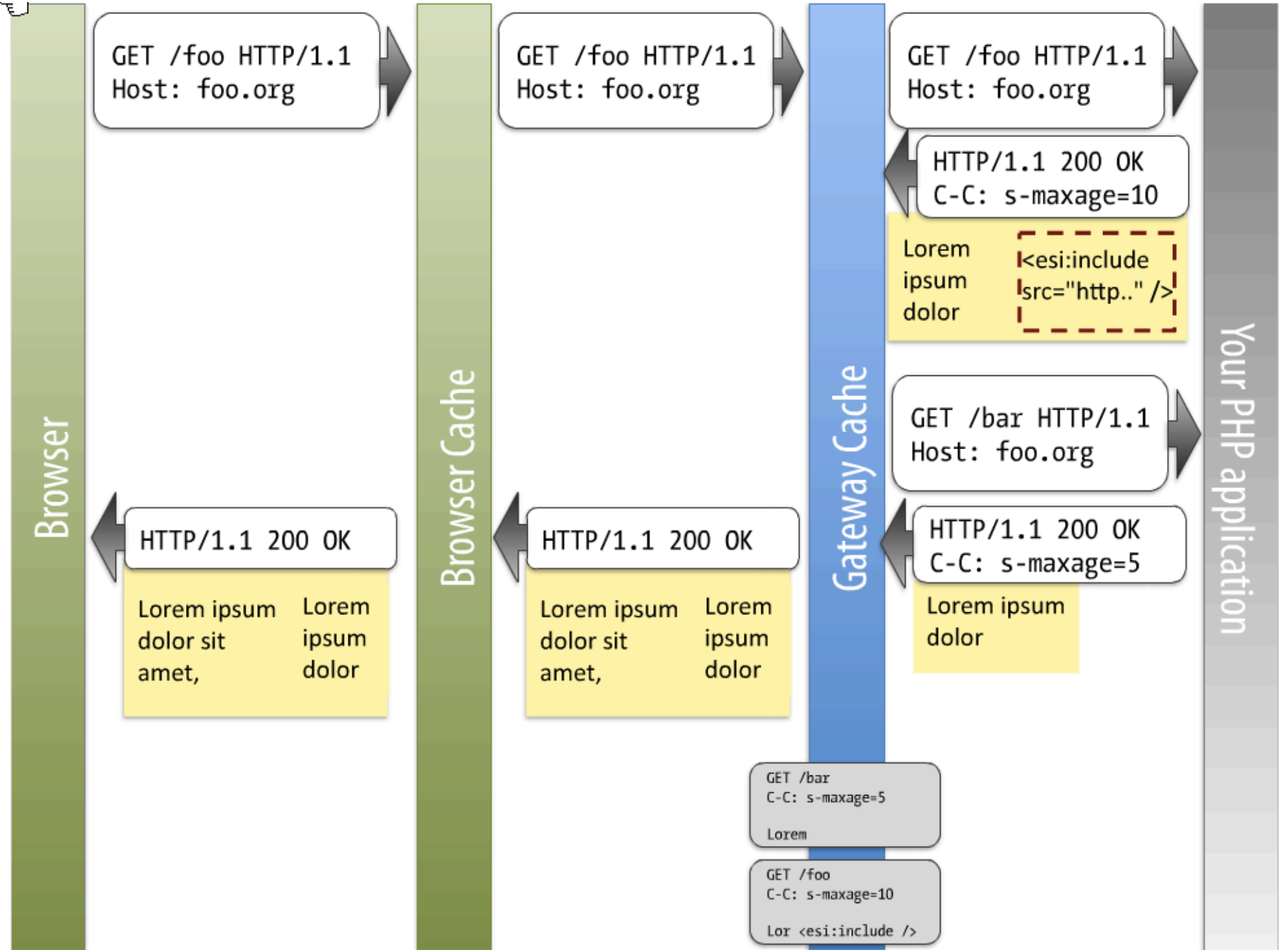
IN THE NEWS

Symfony 2 trainings

Be the first to be trained

REGISTER TODAY

```
<esi:include . . . />
```


Browser
Browser Cache
Gateway Cache
Your PHP application
GET /foo HTTP/1.1
Host: foo.org
GET /foo HTTP/1.1
Host: foo.org
GET /foo HTTP/1.1
Host: foo.org
HTTP/1.1 200 OK
C-C: s-maxage=10
Lorem ipsum dolor
<esi:include src="http.." />
GET /bar HTTP/1.1
Host: foo.org
HTTP/1.1 200 OK
C-C: s-maxage=5
Lorem ipsum dolor
HTTP/1.1 200 OK
Lorem ipsum dolor sit amet,
Lorem ipsum dolor
HTTP/1.1 200 OK
Lorem ipsum dolor sit amet,
Lorem ipsum dolor
GET /bar
C-C: s-maxage=5
Lorem
GET /foo
C-C: s-maxage=10
Lor <esi:include />

Enabling Edge Side Includes

Edge Side Includes must be enabled from the main **app/config/config.yml** configuration file.

```
framework:
    # ...
    esi: ~
```

Rendering an ESI tag

Symfony provides a **render_esi()** Twig function that generates an Edge Side Include tag. It takes a controller reference and an array of variables as its arguments.

```
{{ render_esi(controller(
    'AcmeBlogBundle:Blog:comments',
    { 'max': 10 }
)) }}
```

Rendering an ESI tag

You can also pass an absolute URI instead of a controller reference.

```
{{ render_esi(url(
    'last_comments',
    { 'max': 10 }
)) }}
```

Rendering an ESI response

```
/**
 * Generates a public response
 * to be cached in a shared cache.
 *
 * @Cache(smaxage=600)
 */
public function commentsAction($max = 10)
{
    return $this->render('blog/comments.html.twig', [
        'comments' => CommentDAO::findMostRecents($max),
    ]);
}
```

SensioLabs