

Connecting web and desktop: WebDAV

ConFoo 2011

Tobias Schlitt <toby@qafoo.com>

March 10, 2011



Outline

Introduction

HTTP & WebDAV

Development challenges

Zeta Webdav component



About me

- ▶ Degree in computer science

More than 10 years of
experience with PHP

Open source enthusiast
of PHP, Joomla Components

and Joomla! content



About me

- ▶ Degree in computer science
- ▶ More than 10 years of professional PHP

enthusiast
of Components

and



About me

- ▶ Degree in computer science
- ▶ More than 10 years of professional PHP
- ▶ Open source enthusiast
 - ▶ Apache Zeta Components
 - ▶ Arbit
 - ▶ PHPUnit
 - ▶ ...



About me

- ▶ Degree in computer science
- ▶ More than 10 years of professional PHP
- ▶ Open source enthusiast
 - ▶ Apache Zeta Components
 - ▶ Arbit
 - ▶ PHPUnit
 - ▶ ...

Co-Founder of
Qafoo GmbH

<http://qafoo.com>



About me

- ▶ Degree in computer science
- ▶ More than 10 years of professional PHP
- ▶ Open source enthusiast
 - ▶ Apache Zeta Components
 - ▶ Arbit
 - ▶ PHPUnit
 - ▶ ...

Co-Founder of
Qafoo GmbH

<http://qafoo.com>



We help people to produce
high quality PHP code.

Outline

Introduction

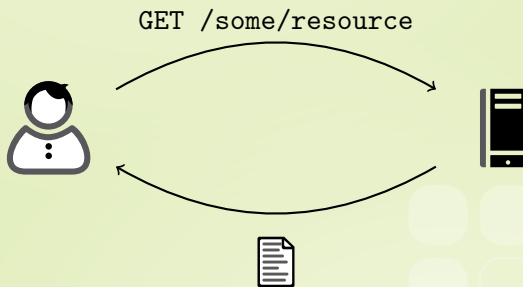
HTTP & WebDAV

Development challenges

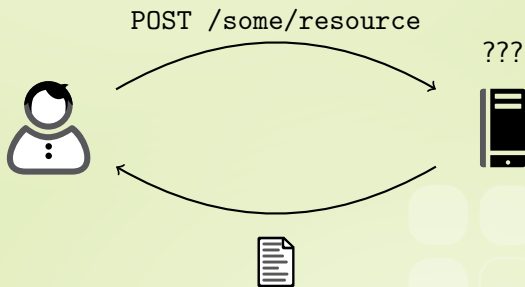
Zeta Webdav component



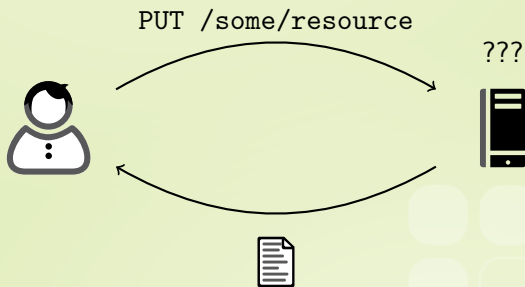
The concept



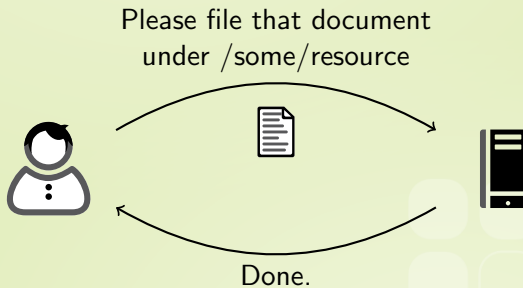
The concept



The concept

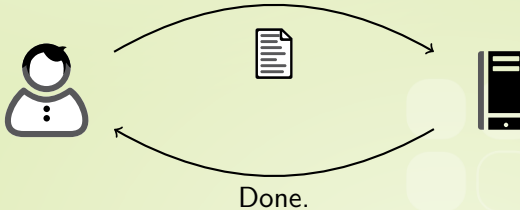


The concept



The concept

Please create a new collection of documents under /some/more



HTTP

- ▶ Network protocol driving the web

- ▶ Current version: 1.1

- ▶ RFC 2616 (June 1999)

<http://tools.ietf.org/html/rfc2616>

Created by Tim Berners-Lee

Application based

Client-server communication

Request / response

Headers / body

Formats / actions



HTTP

- ▶ Network protocol driving the web
- ▶ Current version: 1.1
- ▶ RFC 2616 (June 1999)
<http://tools.ietf.org/html/rfc2616>
- ▶ Inventor: Sir Tim Berners-Lee



HTTP

- ▶ Network protocol driving the web
- ▶ Current version: 1.1
- ▶ RFC 2616 (June 1999)
<http://tools.ietf.org/html/rfc2616>
- ▶ Inventor: Sir Tim Berners-Lee
- ▶ Client / server based
- ▶ Stateless communication

request / response

headers / body

formats / actions



HTTP

- ▶ Network protocol driving the web
- ▶ Current version: 1.1
- ▶ RFC 2616 (June 1999)

<http://tools.ietf.org/html/rfc2616>

- ▶ Inventor: Sir Tim Berners-Lee
- ▶ Client / server based
- ▶ Stateless communication
- ▶ Defines
 - ▶ Request / response
 - ▶ Headers / body
 - ▶ Formats / actions



WebDAV

- ▶ HTTP Extensions for Distributed Authoring
 - ▶ RFC 2518 (February 1999)
 - ▶ <http://tools.ietf.org/html/rfc2518>

▶ HTTP Extensions for Web Distributed Authoring and Versioning

(RFC 4918)

(February 2007)

▶ <http://tools.ietf.org/html/rfc4918>

▶ WebDAV

▶ WebDAV to HTTP

▶ WebDAV for distributed editing



WebDAV

- ▶ HTTP Extensions for Distributed Authoring
 - ▶ RFC 2518 (February 1999)
 - ▶ <http://tools.ietf.org/html/rfc2518>
- ▶ HTTP Extensions for Web Distributed Authoring and Versioning
 - ▶ RFC 4918 (June 2007)
 - ▶ <http://tools.ietf.org/html/rfc4918>

WebDAV

- ▶ HTTP Extensions for Distributed Authoring
 - ▶ RFC 2518 (February 1999)
 - ▶ <http://tools.ietf.org/html/rfc2518>
- ▶ HTTP Extensions for Web Distributed Authoring and Versioning
 - ▶ RFC 4918 (June 2007)
 - ▶ <http://tools.ietf.org/html/rfc4918>
- ▶ IETF Standard

WebDAV is based on HTTP
and enables distributed editing

WebDAV

- ▶ HTTP Extensions for Distributed Authoring
 - ▶ RFC 2518 (February 1999)
 - ▶ <http://tools.ietf.org/html/rfc2518>
- ▶ HTTP Extensions for Web Distributed Authoring and Versioning
 - ▶ RFC 4918 (June 2007)
 - ▶ <http://tools.ietf.org/html/rfc4918>
- ▶ IETF Standard
- ▶ Extension to HTTP
- ▶ Allows distributed editing

WebDAV

- ▶ HTTP Extensions for Distributed Authoring
 - ▶ RFC 2518 (February 1999)
 - ▶ <http://tools.ietf.org/html/rfc2518>
- ▶ HTTP Extensions for Web Distributed Authoring and Versioning
 - ▶ RFC 4918 (June 2007)
 - ▶ <http://tools.ietf.org/html/rfc4918>
- ▶ IETF Standard
- ▶ Extension to HTTP
- ▶ Allows distributed editing

WebDAV allows your users to edit web content easily.

Request methods

▶ HTTP

- ▶ GET
- ▶ POST
- ▶ HEAD

▶ WebDAV

- ▶ MKCOL
- ▶ COPY
- ▶ MOVE
- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ UNLOCK

Request methods

▶ HTTP

- ▶ GET
- ▶ POST
- ▶ HEAD
- ▶ PUT
- ▶ DELETE

▶ WebDAV

- ▶ MKCOL
- ▶ COPY
- ▶ MOVE
- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ UNLOCK

Request methods

▶ HTTP

- ▶ GET
- ▶ POST
- ▶ HEAD
- ▶ PUT
- ▶ DELETE
- ▶ OPTIONS
- ▶ TRACE
- ▶ CONNECT

▶ WebDAV

- ▶ MKCOL
- ▶ COPY
- ▶ MOVE
- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ UNLOCK

Request methods

▶ HTTP

- ▶ GET
- ▶ POST
- ▶ HEAD
- ▶ PUT
- ▶ DELETE
- ▶ OPTIONS
- ▶ TRACE
- ▶ CONNECT

▶ WebDAV

- ▶ MKCOL
- ▶ COPY
- ▶ MOVE
- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ UNLOCK

Request methods

▶ HTTP

- ▶ GET
- ▶ POST
- ▶ HEAD
- ▶ PUT
- ▶ DELETE
- ▶ OPTIONS
- ▶ TRACE
- ▶ CONNECT

▶ WebDAV

- ▶ MKCOL
- ▶ COPY
- ▶ MOVE

- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ UNLOCK

Request methods

▶ HTTP

- ▶ GET
- ▶ POST
- ▶ HEAD
- ▶ PUT
- ▶ DELETE
- ▶ OPTIONS
- ▶ TRACE
- ▶ CONNECT

▶ WebDAV

- ▶ MKCOL
- ▶ COPY
- ▶ MOVE
- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ UNLOCK

Request methods

▶ HTTP

- ▶ GET
- ▶ POST
- ▶ HEAD
- ▶ PUT
- ▶ DELETE
- ▶ OPTIONS
- ▶ TRACE
- ▶ CONNECT

▶ WebDAV

- ▶ MKCOL
- ▶ COPY
- ▶ MOVE
- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ UNLOCK

Request methods

▶ HTTP

- ▶ GET
- ▶ POST
- ▶ HEAD
- ▶ PUT
- ▶ DELETE
- ▶ OPTIONS
- ▶ TRACE
- ▶ CONNECT

▶ WebDAV

- ▶ MKCOL
- ▶ COPY
- ▶ MOVE
- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ UNLOCK

Significant for WebDAV

Request headers

▶ HTTP

- ▶ Accept[-*]
- ▶ Authorization
- ▶ If-[None-]Match
- ▶ If-[Un]Modified-Since
- ▶ User-Agent
- ▶ ...

▶ WebDAV

- ▶ Depth
- ▶ Destination
- ▶ If-None-Match
- ▶ Overwrite
- ▶ Timeout

Request headers

▶ HTTP

- ▶ Accept[-*]
- ▶ Authorization
- ▶ If-[None-]Match
- ▶ If-[Un]Modified-Since
- ▶ User-Agent
- ▶ ...

▶ WebDAV

- ▶ Depth
- ▶ Destination
- ▶ If
- ▶ Overwrite
- ▶ Timeout

Request headers

▶ HTTP

- ▶ Accept[-*]
- ▶ Authorization
- ▶ If-[None-]Match
- ▶ If-[Un]Modified-Since
- ▶ User-Agent
- ▶ ...

▶ WebDAV

- ▶ Depth
- ▶ Destination
- ▶ If
- ▶ Overwrite
- ▶ Timeout

Significant for WebDAV

Response headers

▶ HTTP

- ▶ Accept-Ranges
- ▶ Content-Length
- ▶ Content-Type
- ▶ ETag
- ▶ Location
- ▶ Retry-After
- ▶ Server
- ▶ WWW-Authenticate
- ▶ ...

▶ WebDAV

- ▶ DAV
- ▶ Lock-Token
- ▶ Timeout

Response headers

▶ HTTP

- ▶ Accept-Ranges
- ▶ Content-Length
- ▶ Content-Type
- ▶ ETag
- ▶ Location
- ▶ Retry-After
- ▶ Server
- ▶ WWW-Authenticate
- ▶ ...

▶ WebDAV

- ▶ DAV
- ▶ Lock-Token
- ▶ Timeout

Response headers

▶ HTTP

- ▶ Accept-Ranges
- ▶ Content-Length
- ▶ Content-Type
- ▶ ETag
- ▶ Location
- ▶ Retry-After
- ▶ Server
- ▶ WWW-Authenticate
- ▶ ...

▶ WebDAV

- ▶ DAV
- ▶ Lock-Token
- ▶ Timeout

Significant for WebDAV

Request / response bodies

▶ HTTP

- ▶ Request body mostly not significant
- ▶ Only PUT method needs body (to be stored)
- ▶ Response body usually content to deliver (unspecified)
- ▶ Error responses may contain arbitrary content

Request / response bodies

- ▶ WebDAV
 - ▶ Bodies are significant
 - ▶ Many methods require XML bodies

Request

- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ Potentially others (multi-status)

Response

- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ Potentially others (multi-status)

Request / response bodies

- ▶ WebDAV
 - ▶ Bodies are significant
 - ▶ Many methods require XML bodies

Request

- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ COPY (optional)
- ▶ MOVE (optional)
- ▶ LOCK

Response

- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ Potentially others (multi-status)

Request / response bodies

▶ WebDAV

- ▶ Bodies are significant
- ▶ Many methods require XML bodies

Request

- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ COPY (optional)
- ▶ MOVE (optional)
- ▶ LOCK

Response

- ▶ PROPFIND
- ▶ PROPPATCH
- ▶ LOCK
- ▶ Potentially others
(multi-status)

Properties

- ▶ Concept introduced by WebDAV
- ▶ Store meta information about content
- ▶ Usually not directly visible to the user

Types

▶ Date

▶ Name

▶ Size

▶ Discovery

▶ Dead properties

▶ Arbitrary data

▶ Custom namespace

▶ XML favored

Properties

- ▶ Concept introduced by WebDAV
- ▶ Store meta information about content
- ▶ Usually not directly visible to the user

- ▶ Live properties

- ▶ creationdate
- ▶ displayname
- ▶ get*
- ▶ lockdiscovery
- ▶ ...

- ▶ Dead properties

- ▶ Arbitrary data
- ▶ Custom namespace
- ▶ XML favored

Properties

- ▶ Concept introduced by WebDAV
- ▶ Store meta information about content
- ▶ Usually not directly visible to the user

- ▶ Live properties
 - ▶ creationdate
 - ▶ displayname
 - ▶ get*
 - ▶ lockdiscovery
 - ▶ ...
- ▶ Dead properties
 - ▶ Arbitrary data
 - ▶ Custom namespace
 - ▶ XML favored

Outline

Introduction

HTTP & WebDAV

Development challenges

Zeta Webdav component



Development challenges

- ▶ Server development in general
- ▶ WebDAV RFCs are a BBOM
 - ▶ Unstructured
 - ▶ Ambiguous
 - ▶ Design fails
- ▶ Misbehaving clients
 - ▶ Ignore the specification
 - ▶ Different interpretations of RFCs
 - ▶ Proprietary BS
- ▶ Exchangeable back ends



Outline

Development challenges

- RFC problems

- Client problems

- Back end flexibility



COPY / MOVE methods

► **Errors on COPY**

“[...] if an error occurs while copying an internal collection, the server **MUST NOT** copy any resources identified by members of this collection (i.e., the server must skip this subtree) [...]” [?]

“[...] if an error occurs while copying an internal collection, the server **SHOULD** perform an equivalent of a copy (COPY), followed by a copy of the original collection for maintenance processing, followed by a delete of the original collection [...]” [?]

“[...]” [?]

“[...] if an error occurs while moving an internal collection, the server **SHOULD** finish as much of the original move as possible [...]” [?]

COPY / MOVE methods

► Errors on COPY

“[...] if an error occurs while copying an internal collection, the server MUST NOT copy any resources identified by members of this collection (i.e., the server must skip this subtree) [...]” [?]

► MOVE

“[...] is the logical equivalent of a copy (COPY), followed by consistency maintenance processing, followed by a delete of the source,[...]” [?]

Errors

“[...] after detecting the error, the move operation SHOULD finish as much of the original move as possible [...]” [?]

COPY / MOVE methods

▶ **Errors on COPY**

“[...] if an error occurs while copying an internal collection, the server **MUST NOT** copy any resources identified by members of this collection (i.e., the server must skip this subtree) [...]” [?]

▶ **MOVE**

“[...] is the logical equivalent of a copy (COPY), followed by consistency maintenance processing, followed by a delete of the source, [...]” [?]

▶ **MOVE errors**

“[...] after detecting the error, the move operation **SHOULD** try to finish as much of the original move as possible [...]” [?]

The If header

- ▶ Makes operations conditional.
- ▶ Apply operation only if
 - ▶ all conditions are met
 - ▶ no condition is met

```
1 No-tag-list = List
2 Tagged-list = Resource 1*List
3 Resource   = Coded-URL
4 List       = "("
5             1*(["Not"](State-token | "[" entity-tag "]"))
6             ")"
7 State-token = Coded-URL
8 Coded-URL   = "<" absoluteURI ">"
```

The If header

- ▶ Can contain
 - ▶ lock tokens
 - ▶ entity tags

```
1 If: (<locktoken:a-write-lock-token>  
2     [" I am an ETag" ]) ([" I am another ETag" ])
```

The If header

- ▶ Conditions can apply to
 - ▶ single resources
 - ▶ sets of resources
 - ▶ all affected resources

```
1 <http://example.com/resource1>  
2   (<locktoken:a-write-lock-token> [W/"A_weak_ETag" ])  
3   (Not [ "strong_ETag" ])
```

- ▶ Required own parser implementation
- ▶ Parser is about 150 LOC

Locking

- ▶ 2 different types of locks
 - ▶ Exclusive
 - ▶ Shared

Lock conditions must be validated before anything is done
Must refresh on every lock use (successful or not)

How to associate principles with lock tokens

Locking Resources

(RFC 4918)

Do not behave like real resources

Do not vanish when the lock is released

Lock collection can be created on them

Locking

- ▶ 2 different types of locks
 - ▶ Exclusive
 - ▶ Shared
- ▶ Lock conditions must be validated before anything else

▶ Must refresh on every lock use (successful or not)

▶ Must know how to associate principles with lock tokens

▶ Locks are Resources

▶ Defined in RFC 4918

▶ Must not behave like real resources

▶ Must not vanish when the lock is released

▶ Lock collection can be created on them

Locking

- ▶ 2 different types of locks
 - ▶ Exclusive
 - ▶ Shared
- ▶ Lock conditions must be validated before anything else
- ▶ Timeout refresh on every lock use (successful or not)
(Fixed in RFC 4918)

- ▶ Locks are not how to associate principles with lock tokens
- ▶ Locks are not Resources
 - ▶ Locks are not defined in RFC 4918
 - ▶ Locks do not behave like real resources
 - ▶ Locks do not vanish when the lock is released
 - ▶ Locks do not have a collection that can be created on them

Locking

- ▶ 2 different types of locks
 - ▶ Exclusive
 - ▶ Shared
- ▶ Lock conditions must be validated before anything else
- ▶ Timeout refresh on every lock use (successful or not)
(Fixed in RFC 4918)
- ▶ **Not specified how to associate principles with lock tokens**

Lock Tokens

(RFC 4918)

do not behave like real resources

do not vanish when the lock is released

lock collection can be created on them

Locking

- ▶ 2 different types of locks
 - ▶ Exclusive
 - ▶ Shared
- ▶ Lock conditions must be validated before anything else
- ▶ Timeout refresh on every lock use (successful or not)
(Fixed in RFC 4918)
- ▶ Not specified how to associate principles with lock tokens
- ▶ **Lock-Null-Resources**
(Partly fixed in RFC 4918)
 - ▶ Do not behave like real resources
 - ▶ Must vanish when the lock is released
 - ▶ A collection can be created on them

Outline

Development challenges

RFC problems

Client problems

Back end flexibility



Konqueror / Nautilus

- ▶ Konqueror (KDE)
 - ▶ Does not decode URLs properly
 - ▶ Requires Apache like error messages for 404

... with charset="..." info in MIME types

Konqueror / Nautilus

- ▶ Konqueror (KDE)
 - ▶ Does not decode URLs properly
 - ▶ Requires Apache like error messages for 404
- ▶ Nautilus (Gnome)
 - ▶ Cannot cope with `charset="..."` info in MIME types

Windows / InternetExplorer

- ▶ At least 3 different WebDAV user agents in Windows
 - ▶ Loaded depending on how you initialize connection
 - ▶ Transparently switched occasionally

- ▶ Always send `MS-Auth-Via` header on every response
- ▶ Always send `DAV:` namespaces set on live properties
- ▶ Always send special namespace shortcut to be used (sic!)
- ▶ Always send different shortcuts for DAV: namespace

Windows / InternetExplorer

- ▶ At least 3 different WebDAV user agents in Windows
 - ▶ Loaded depending on how you initialize connection
 - ▶ Transparently switched occasionally
- ▶ Requires custom header **MS-Author-Via** on every response

• `msc:*` namespaces set on live properties

• `msc:*` is special namespace shortcut to be used (sic!)

• `msc:*` uses different shortcuts for DAV: namespace

Windows / InternetExplorer

- ▶ At least 3 different WebDAV user agents in Windows
 - ▶ Loaded depending on how you initialize connection
 - ▶ Transparently switched occasionally
- ▶ Requires custom header `MS-Author-Via` on every response
- ▶ Requires custom namespaces set on live properties
 - ▶ Requires special namespace shortcut to be used (sic!)
 - ▶ Requires different shortcuts for DAV: namespace

Windows / InternetExplorer II

- ▶ Cannot cope with non-significant white spaces in XML bodies

- ▶ Missing carriage return at the end of every XML body
- ▶ Missing carriage returns leads invalid PROPFIND requests

Windows / InternetExplorer II

- ▶ Cannot cope with non-significant white spaces in XML bodies
- ▶ Requires newline at the end of every XML body

ends invalid PROPFIND requests

Windows / InternetExplorer II

- ▶ Cannot cope with non-significant white spaces in XML bodies
- ▶ Requires newline at the end of every XML body
- ▶ Occasionally sends invalid PROPFIND requests

Outline

Development challenges

RFC problems

Client problems

Back end flexibility



Back end flexibility

- ▶ Exchangeable back end
 - ▶ File system
 - ▶ Memory (testing)
 - ▶ SQL Database?
 - ▶ Subversion?

no client issues
enhancements back end independent
implementation

Back end flexibility

- ▶ Exchangeable back end
 - ▶ File system
 - ▶ Memory (testing)
 - ▶ SQL Database?
 - ▶ Subversion?
- ▶ Independent of client issues

enhancements back end independent
implementation

Back end flexibility

- ▶ Exchangeable back end
 - ▶ File system
 - ▶ Memory (testing)
 - ▶ SQL Database?
 - ▶ Subversion?
- ▶ Independent of client issues
- ▶ Protocol enhancements back end independent

Implementation

Back end flexibility

- ▶ Exchangeable back end
 - ▶ File system
 - ▶ Memory (testing)
 - ▶ SQL Database?
 - ▶ Subversion?
- ▶ Independent of client issues
- ▶ Protocol enhancements back end independent
- ▶ Easy implementation

Outline

Introduction

HTTP & WebDAV

Development challenges

Zeta Webdav component



Outline

Zeta Webdav component

Background

Usage

Customization

End note



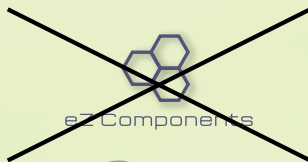
eZ Components



eZ Components



Apache Zeta Components



History: eZ Components

- ▶ Developed since 2004 by eZ Systems AS
 - ▶ Makers of enterprise OSS CMS eZ Publish
 - ▶ <http://ez.no>
- ▶ Initial goal
 - ▶ Refactor and cleanup core libs of eZ Publish
 - ▶ Pull out standalone, general purpose library
- ▶ Objectives
 - ▶ High quality code
 - ▶ Well thought out API
 - ▶ Backwards compatibility
 - ▶ Extensive documentation

History: The break

March 2010

- ▶ All eZ Components developers left eZ Systems
- ▶ Still, we wanted to . . .
 - ▶ put voluntary efforts into the project
 - ▶ push it further
 - ▶ ensure it is not broken
- ▶ Discussion with eZ Systems
 - ▶ Spin off the project to become independent
 - ▶ Look for or create a foundation
- ▶ Result: Propose code to Apache Software Foundation

Apache Software Foundation

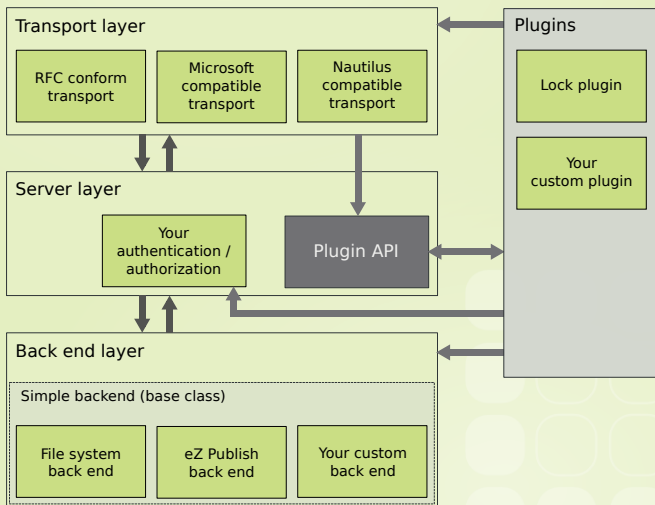


- ▶ <http://apache.org>
- ▶ Well known OSS foundation
- ▶ Home of OSS projects like
 - ▶ Apache HTTP server
 - ▶ Subversion
 - ▶ CouchDB
- ▶ <http://zeta-components.org>

The Webdav component

- ▶ General purpose WebDAV server
- ▶ Easy integration and customization
- ▶ Work around client issues

Zeta Webdav architecture



Outline

Zeta Webdav component

Background

Usage

Customization

End note



Setup a simple WebDAV server

```
1 $server = ezcWebdavServer::getInstance();
2 $backend = new ezcWebdavFileBackend(
3     dirname( __FILE__ ) . '/backend'
4 );
5
6 $server->handle( $backend );
```

Setup a simple WebDAV server

```
1 $server = ezcWebdavServer::getInstance();
2 $backend = new ezcWebdavFileBackend(
3     dirname( __FILE__ ) . '/backend'
4 );
5
6 $server->handle( $backend );
```

Setup a simple WebDAV server

```
1 $server = ezcWebdavServer::getInstance();
2 $backend = new ezcWebdavFileBackend(
3     dirname( __FILE__ ) . '/backend'
4 );
5
6 $server->handle( $backend );
```

Setup a simple WebDAV server

```
1 $server = ezcWebdavServer::getInstance();
2 $backend = new ezcWebdavFileBackend(
3     dirname( __FILE__ ) . '/backend'
4 );
5
6 $server->handle( $backend );
```

WebDAV server with locking

```
1  require_once 'custom_lock_auth.php';
2
3  $server = ezcWebdavServer::getInstance();
4
5  $server->auth = new myCustomLockAuth(
6      // Some configuration directory here
7      dirname( __FILE__ ) . '/tokens.php'
8  );
9
10 $server->pluginRegistry->registerPlugin(
11     new ezcWebdavLockPluginConfiguration()
12 );
13
14 $backend = new ezcWebdavFileBackend(
15     // Your WebDAV directory here
16     dirname( __FILE__ ) . '/backend'
17 );
18
19 $server->handle( $backend );
```

WebDAV server with locking

```
1 require_once 'custom_lock_auth.php';
2
3 $server = ezcWebdavServer::getInstance();
4
5 $server->auth = new myCustomLockAuth(
6     // Some configuration directory here
7     dirname( __FILE__ ) . '/tokens.php'
8 );
9
10 $server->pluginRegistry->registerPlugin(
11     new ezcWebdavLockPluginConfiguration()
12 );
13
14 $backend = new ezcWebdavFileBackend(
15     // Your WebDAV directory here
16     dirname( __FILE__ ) . '/backend'
17 );
18
19 $server->handle( $backend );
```

WebDAV server with locking

```
1 require_once 'custom_lock_auth.php';
2
3 $server = ezcWebdavServer::getInstance();
4
5 $server->auth = new myCustomLockAuth(
6     // Some configuration directory here
7     dirname( __FILE__ ) . '/tokens.php'
8 );
9
10 $server->pluginRegistry->registerPlugin(
11     new ezcWebdavLockPluginConfiguration()
12 );
13
14 $backend = new ezcWebdavFileBackend(
15     // Your WebDAV directory here
16     dirname( __FILE__ ) . '/backend'
17 );
18
19 $server->handle( $backend );
```


WebDAV server with locking

```
1  require_once 'custom_lock_auth.php';
2
3  $server = ezcWebdavServer::getInstance();
4
5  $server->auth = new myCustomLockAuth(
6      // Some configuration directory here
7      dirname( __FILE__ ) . '/tokens.php'
8  );
9
10 $server->pluginRegistry->registerPlugin (
11     new ezcWebdavLockPluginConfiguration()
12 );
13
14 $backend = new ezcWebdavFileBackend(
15     // Your WebDAV directory here
16     dirname( __FILE__ ) . '/backend'
17 );
18
19 $server->handle( $backend );
```

Authentication

```
1 interface ezcWebdavAnonymousAuthenticator
2 {
3     public function authenticateAnonymous( ezcWebdavAnonymousAuth $data );
4 }

1 interface ezcWebdavBasicAuthenticator extends ezcWebdavAnonymousAuthenticator
2 {
3     public function authenticateBasic( ezcWebdavBasicAuth $data );
4 }

1 interface ezcWebdavDigestAuthenticator extends ezcWebdavBasicAuthenticator
2 {
3     public function authenticateDigest( ezcWebdavDigestAuth $data );
4 }
```

Authentication

```
1 interface ezcWebdavAnonymousAuthenticator
2 {
3     public function authenticateAnonymous( ezcWebdavAnonymousAuth $data );
4 }
```

```
1 interface ezcWebdavBasicAuthenticator extends ezcWebdavAnonymousAuthenticator
2 {
3     public function authenticateBasic( ezcWebdavBasicAuth $data );
4 }
```

```
1 interface ezcWebdavDigestAuthenticator extends ezcWebdavBasicAuthenticator
2 {
3     public function authenticateDigest( ezcWebdavDigestAuth $data );
4 }
```

Authentication

```
1 interface ezcWebdavAnonymousAuthenticator
2 {
3     public function authenticateAnonymous( ezcWebdavAnonymousAuth $data );
4 }
```

```
1 interface ezcWebdavBasicAuthenticator extends ezcWebdavAnonymousAuthenticator
2 {
3     public function authenticateBasic( ezcWebdavBasicAuth $data );
4 }
```

```
1 interface ezcWebdavDigestAuthenticator extends ezcWebdavBasicAuthenticator
2 {
3     public function authenticateDigest( ezcWebdavDigestAuth $data );
4 }
```

Authentication

```
1 interface ezcWebdavAnonymousAuthenticator
2 {
3     public function authenticateAnonymous( ezcWebdavAnonymousAuth $data );
4 }
```

```
1 interface ezcWebdavBasicAuthenticator extends ezcWebdavAnonymousAuthenticator
2 {
3     public function authenticateBasic( ezcWebdavBasicAuth $data );
4 }
```

```
1 interface ezcWebdavDigestAuthenticator extends ezcWebdavBasicAuthenticator
2 {
3     public function authenticateDigest( ezcWebdavDigestAuth $data );
4 }
```

Authorization

```
1 interface ezcWebdavAuthorizer
2 {
3     const ACCESS_READ = 1;
4     const ACCESS_WRITE = 2;
5
6     public function authorize( $user, $path, $access = self::ACCESS_READ );
7 }
```

```
1 interface ezcWebdavLockAuthorizer extends ezcWebdavAuthorizer
2 {
3     public function assignLock( $user, $lockToken );
4     public function ownsLock( $user, $lockToken );
5     public function releaseLock( $user, $lockToken );
6
7 }
8
```

Authorization

```
1 interface ezcWebdavAuthorizer
2 {
3     const ACCESS_READ = 1;
4     const ACCESS_WRITE = 2;
5
6     public function authorize( $user, $path, $access = self::ACCESS_READ );
7 }
```

```
1 interface ezcWebdavLockAuthorizer extends ezcWebdavAuthorizer
2 {
3     public function assignLock( $user, $lockToken );
4     public function ownsLock( $user, $lockToken );
5     public function releaseLock( $user, $lockToken );
6
7 }
8
```

Authorization

```
1 interface ezcWebdavAuthorizer
2 {
3     const ACCESS_READ = 1;
4     const ACCESS_WRITE = 2;
5
6     public function authorize( $user, $path, $access = self::ACCESS_READ );
7 }
```

```
1 interface ezcWebdavLockAuthorizer extends ezcWebdavAuthorizer
2 {
3     public function assignLock( $user, $lockToken );
4     public function ownsLock( $user, $lockToken );
5     public function releaseLock( $user, $lockToken );
6
7 }
8 }
```


Authorization

```
1 interface ezcWebdavAuthorizer
2 {
3     const ACCESS_READ = 1;
4     const ACCESS_WRITE = 2;
5
6     public function authorize( $user, $path, $access = self::ACCESS_READ );
7 }
```

```
1 interface ezcWebdavLockAuthorizer extends ezcWebdavAuthorizer
2 {
3     public function assignLock( $user, $lockToken );
4     public function ownsLock( $user, $lockToken );
5     public function releaseLock( $user, $lockToken );
6 }
7
8 }
```

Authorization

```
1 interface ezcWebdavAuthorizer
2 {
3     const ACCESS_READ = 1;
4     const ACCESS_WRITE = 2;
5
6     public function authorize( $user, $path, $access = self::ACCESS_READ );
7 }
```

```
1 interface ezcWebdavLockAuthorizer extends ezcWebdavAuthorizer
2 {
3     public function assignLock( $user, $lockToken );
4     public function ownsLock( $user, $lockToken );
5     public function releaseLock( $user, $lockToken );
6 }
7
8 }
```

Authorization

```
1 interface ezcWebdavAuthorizer
2 {
3     const ACCESS_READ = 1;
4     const ACCESS_WRITE = 2;
5
6     public function authorize( $user, $path, $access = self::ACCESS_READ );
7 }
```

```
1 interface ezcWebdavLockAuthorizer extends ezcWebdavAuthorizer
2 {
3     public function assignLock( $user, $lockToken );
4     public function ownsLock( $user, $lockToken );
5     public function releaseLock( $user, $lockToken );
6 }
7
8 }
```

Outline

Zeta Webdav component

Background

Usage

Customization

End note



Path handling

```
1 $server = ezcWebdavServer::getInstance();
2
3 $pathFactory = new ezcWebdavBasicPathFactory(
4     'http://example.com/webdav/index.php'
5 );
6
7 foreach ( $server->configurations as $conf )
8 {
9     $conf->pathFactory = $pathFactory;
10 }
11
12 $backend = new ezcWebdavFileBackend(
13     dirname( __FILE__ ) . '/backend'
14 );
15
16 $server->handle( $backend );
```

Path handling

```
1 $server = ezcWebdavServer::getInstance();
2
3 $pathFactory = new ezcWebdavBasicPathFactory(
4     'http://example.com/webdav/index.php'
5 );
6
7 foreach ( $server->configurations as $conf )
8 {
9     $conf->pathFactory = $pathFactory;
10 }
11
12 $backend = new ezcWebdavFileBackend(
13     dirname( __FILE__ ) . '/backend'
14 );
15
16 $server->handle( $backend );
```

Path handling

```
1 $server = ezcWebdavServer::getInstance();
2
3 $pathFactory = new ezcWebdavBasicPathFactory(
4     'http://example.com/webdav/index.php'
5 );
6
7 foreach ( $server->configurations as $conf )
8 {
9     $conf->pathFactory = $pathFactory;
10 }
11
12 $backend = new ezcWebdavFileBackend(
13     dirname( __FILE__ ) . '/backend'
14 );
15
16 $server->handle( $backend );
```

The back end

- ▶ `ezcWebdavFileBackend`
- ▶ `ezcWebdavMemoryBackend`

▶ `ezcWebdavBackend` (your own)

▶ `ezcWebdavSimpleBackend`

▶ `ezcWebdavBackendGet`

▶ `ezcWebdavBackendPut`

▶ `ezcWebdavBackendChange`

▶ `ezcWebdavBackendMakeCollection`

▶ `ezcWebdavLockBackend`

The back end

- ▶ `ezcWebdavFileBackend`
- ▶ `ezcWebdavMemoryBackend`
- ▶ Write your own

```
ezcWebdavSimpleBackend  
ezcWebdavBackend  
ezcWebdavBackendPut  
ezcWebdavBackendChange  
ezcWebdavBackendMakeCollection  
ezcWebdavLockBackend
```

The back end

- ▶ `ezcWebdavFileBackend`
- ▶ `ezcWebdavMemoryBackend`
- ▶ Write your own
 - ▶ `ezcWebdavSimpleBackend`

```
ezcWebdavBackend  
ezcWebdavBackendPut  
ezcWebdavBackendChange  
ezcWebdavBackendMakeCollection  
ezcWebdavLockBackend
```

The back end

- ▶ `ezcWebdavFileBackend`
- ▶ `ezcWebdavMemoryBackend`
- ▶ Write your own
 - ▶ `ezcWebdavSimpleBackend`
 - ▶ `ezcWebdavBackend`
 - ▶ `ezcWebdavBackendPut`
 - ▶ `ezcWebdavBackendChange`
 - ▶ `ezcWebdavBackendMakeCollection`
 - ▶ `ezcWebdavLockBackend`

ezcWebdavSimpleBackend

- ▶ Proper method handling
- ▶ Authorization
- ▶ Handling of If[-*] headers
- ▶ about 1600 LOC
- ▶ No deeper knowledge on WebDAV required

New clients

- ▶ Adjust transport layer
- ▶ Needs some patience
 - ▶ Use Wireshark!
- ▶ Extend one or more of
 - ▶ `ezcWebdavTransport`
 - ▶ `ezcWebdavHeaderHandler`
 - ▶ `ezcWebdavPropertyHandler`



New clients

- ▶ Adjust transport layer
- ▶ Needs some patience
 - ▶ Use Wireshark!
- ▶ Extend one or more of
 - ▶ `ezcWebdavTransport`
 - ▶ `ezcWebdavHeaderHandler`
 - ▶ `ezcWebdavPropertyHandler`

```
1 $newClientConf = new ezcWebdavServerConfiguration(  
2     '(My.*Webdav\s+Client)i',  
3     'myCustomTransportTransport'  
4     // ...  
5 );  
6  
7 $server->configurations->insertBefore( $newClientConf, 0 );
```

New clients

- ▶ Adjust transport layer
- ▶ Needs some patience
 - ▶ Use Wireshark!
- ▶ Extend one or more of
 - ▶ `ezcWebdavTransport`
 - ▶ `ezcWebdavHeaderHandler`
 - ▶ `ezcWebdavPropertyHandler`

```
1 $newClientConf = new ezcWebdavServerConfiguration(  
2     '(My.*Webdav\s+Client)i',  
3     'myCustomTransportTransport'  
4     // ...  
5 );  
6  
7 $server->configurations->insertBefore( $newClientConf, 0 );
```

Plugins

- ▶ Hook into requests / responses
- ▶ Handle unknown requests
- ▶ Issue unknown responses
- ▶ Send requests to the back end



Outline

Zeta Webdav component

Background

Usage

Customization

End note



Join the team!

- ▶ Come and contribute!
- ▶ Highly technical discussions
- ▶ Open minded people
- ▶ Apache Software Foundation
- ▶ <http://zeta-components.org>



Q / A

Are there any questions left?



Thanks for listening

Please rate this talk at
<http://joind.in/2808>
and / or give me some feedback right now!

Thanks for listening

Please rate this talk at
<http://joind.in/2808>
and / or give me some feedback right now!

This is very important for ...

- ▶ Speakers
- ▶ Organizers
- ▶ You!



Thanks for listening

Please rate this talk at
<http://joind.in/2808>
and / or give me some feedback right now!

Stay in touch

- ▶ Tobias Schlitt
- ▶ toby@qafoo.com
- ▶ @tobySen / @qafoo

Rent a PHP quality expert:

<http://qafoo.com>