RFC 223: Http Whiteboard Updates

Draft

65 Pages

Abstract

Updates to Http Whiteboard for Release 7.

0 Document Information

0.1 License

DISTRIBUTION AND FEEDBACK LICENSE, Version 2.0

The OSGi Alliance hereby grants you a limited copyright license to copy and display this document (the “Distribution”) in any medium without fee or royalty. This Distribution license is exclusively for the purpose of reviewing and providing feedback to the OSGi Alliance. You agree not to modify the Distribution in any way and further agree to not participate in any way in the making of derivative works thereof, other than as a necessary result of reviewing and providing feedback to the Distribution. You also agree to cause this notice, along with the accompanying consent, to be included on all copies (or portions thereof) of the Distribution. The OSGi Alliance also grants you a perpetual, non-exclusive, worldwide, fully paid-up, royalty free, limited license (without the right to sublicense) under any applicable copyrights, to create and/or distribute an implementation of the Distribution that: (i) fully implements the Distribution including all its required interfaces and functionality; (ii) does not modify, subset, superset or otherwise extend the OSGi Name Space, or include any public or protected packages, classes, Java interfaces, fields or methods within the OSGi Name Space other than those required and authorized by the Distribution. An implementation that does not satisfy limitations (i)-(ii) is not considered an implementation of the Distribution, does not receive the benefits of this license, and must not be described as an implementation of the Distribution. "OSGi Name Space" shall mean the public class or interface declarations whose names begin with "org.osgi" or any recognized successors or replacements thereof. The OSGi Alliance expressly reserves all rights not granted pursuant to these limited copyright licenses including termination of the license at will at any time.

EXCEPT FOR THE LIMITED COPYRIGHT LICENSES GRANTED ABOVE, THE OSGi ALLIANCE DOES NOT GRANT, EITHER EXPRESSLY OR IMPLIEDLY, A LICENSE TO ANY INTELLECTUAL PROPERTY IT, OR ANY THIRD PARTIES, OWN OR CONTROL. Title to the copyright in the Distribution will at all times remain with the OSGi Alliance. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted therein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

THE DISTRIBUTION IS PROVIDED "AS IS," AND THE OSGi ALLIANCE (INCLUDING ANY THIRD PARTIES THAT HAVE CONTRIBUTED TO THE DISTRIBUTION) MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DISTRIBUTION ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.
NEITHER THE OSGi ALLIANCE NOR ANY THIRD PARTY WILL BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THE DISTRIBUTION.

Implementation of certain elements of this Distribution may be subject to third party intellectual property rights, including without limitation, patent rights (such a third party may or may not be a member of the OSGi Alliance). The OSGi Alliance is not responsible and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

The Distribution is a draft. As a result, the final product may change substantially by the time of final publication, and you are cautioned against relying on the content of this Distribution. You are encouraged to update any implementation of the Distribution if and when such Distribution becomes a final specification.

The OSGi Alliance is willing to receive input, suggestions and other feedback (“Feedback”) on the Distribution. By providing such Feedback to the OSGi Alliance, you grant to the OSGi Alliance and all its Members a non-exclusive, non-transferable,

worldwide, perpetual, irrevocable, royalty-free copyright license to copy, publish, license, modify, sublicense or otherwise distribute and exploit your Feedback for any purpose. Likewise, if incorporation of your Feedback would cause an implementation of the Distribution, including as it may be modified, amended, or published at any point in the future ("Future Specification"), to necessarily infringe a patent or patent application that you own or control, you hereby commit to grant to all implementers of such Distribution or Future Specification an irrevocable, worldwide, sublicenseable, royalty free license under such patent or patent application to make, have made, use, sell, offer for sale, import and export products or services that implement such Distribution or Future Specification. You warrant that (a) to the best of your knowledge you have the right to provide this Feedback, and if you are providing Feedback on behalf of a company, you have the rights to provide Feedback on behalf of your company; (b) the Feedback is not confidential to you and does not violate the copyright or trade secret interests of another; and (c) to the best of your knowledge, use of the Feedback would not cause an implementation of the Distribution or a Future Specification to necessarily infringe any third-party patent or patent application known to you. You also acknowledge that the OSGi Alliance is not required to incorporate your Feedback into any version of the Distribution or a Future Specification.

I HEREBY ACKNOWLEDGE AND AGREE TO THE TERMS AND CONDITIONS DELINEATED ABOVE.

0.2 Trademarks

OSGi™ is a trademark, registered trademark, or service mark of the OSGi Alliance in the US and other countries. Java is a trademark, registered trademark, or service mark of Oracle Corporation in the US and other countries. All other trademarks, registered trademarks, or service marks used in this document are the property of their respective owners and are hereby recognized.

0.3 Feedback

This document can be downloaded from the OSGi Alliance design repository at https://github.com/osgi/design The public can provide feedback about this document by opening a bug at https://www.osgi.org/bugzilla/.

0.4 Table of Contents

0.5 Terminology and Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interpreted as described in 10.1.

`Source code is shown in this typeface.`

0.6 Revision History

The last named individual in this history is currently responsible for this document.

Revision	Date	Comments
Initial	2016-04-22	Initial draft Carsten Ziegeler, Adobe
0.1	2016-06-28	Updates from the Darmstadt F2F Carsten Ziegeler, Adobe
0.2	2016-07-06	Updates from the Darmstadt F2F Raymond Augé, Liferay

1 Introduction

This RFC collects a numbers of requested enhancements to Http Whiteboard Service that were suggested after Release 6 design work was completed.

2 Application Domain

The Http Whiteboard Specification was first released in 2015 as part of Release 6. From the Version 1.0 spec:

> The OSGi Http Whiteboard Specification provides a light and convenient way of using servlets, servlet filters, servlet listeners and web resources in an OSGi environment through the use of the [7] Whiteboard Pattern.

3 Problem Description

3.1 Whiteboard Services and Http Service (Bug 2872)

If a Http Whiteboard implementation is also implementing the Http Service, the whiteboard specification does not specify whether the Http contexts for the Http Service are represented as ServletContextHelper services. There is no way for a whiteboard service to be registered in a Http Context of the Http Service. For example adding a servlet filter for all servlets managed by the Http Service is not possible.

3.2 Reusable Logic across Servlet Contexts (Bug 2900)

Servlet filters are run after ServletContextHelper.handleSecurity, therefore code for logging all requests or handling common security problems (e.g handling Cross-Origin Resource Sharing) needs the be run as part of the ServletContextHelper.handleSecurity method. There is currently no way to share this common logic across different ServletContextHelpers.

3.3 Multipart Configuration Handling (Bug 2870)

There's no way to set multipart configurations for servlets. As such there's no way to use the Servlet 3.0 file upload API.

The Servlet 3.0 way of doing this would have been:

A) using the @MultipartConfig annotation:
https://docs.oracle.com/javaee/6/api/javax/servlet/annotation/MultipartConfig.html

b) using the web.xml and providing a sub element multipart-config.

```
<servlet>
   <servlet-name>StudentRegistrationUsn</servlet-name>
   <multipart-config>
      <max-file-size>10485760</max-file-size>
      <max-request-size>20971520</max-request-size>
      <file-size-threshold>5242880</file-size-threshold>
   </multipart-config>
</servlet>
```

3.4 Support Servlets without a pattern (Bug 2897)

Version 1.0 of the Http Whiteboard Specification requires that a registered servlet has pattern. However the servlet spec allows to register named servlets which can be targeted by “named dispatching” and these servlets might be registered without a pattern.

4 Requirements

HW-0010 - Provide a way to register servlets, filters, listeners, and resources through the whiteboard service with the Http Service.

HW-0020 - Provide a mechanism to share logic between ServletContextHelpers.

HW-0030 - Provide a mechanism to configure servlets for file upload.

HW-0040 - Allow to register servlets with just a name (and no patterns)

HW-0050 – Provide a mechanism to provide an alternative logger for logging through ServletContext.

5 Technical Solution

5.1 Whiteboard Services and Http Service

As the Http Whiteboard Specification does not specify if and how Http Contexts managed by the Http Service are registered as ServletContextHelper services, there is currently no way for a whiteboard service to target these. When a servlet or resource is registered with the Http Service, it is either registered with the default Http Context or with a provided one. These objects have no way to identify them for example via a name or a path.

A whiteboard service which should be registered with an Http Context from the Http Service can target this by filtering for ServletContextHelper services having the service registration property `osgi.http.whiteboard.context.httpservice`. The value for this property is not further specified.

The following example registers a servlet filter for all servlets managed by the Http Service:

```
@Component(service = Filter.class, scope=ServiceScope.PROTOTYPE,
   property={
     "osgi.http.whiteboard.filter.pattern=/",
     "osgi.http.whiteboard.context.select=" +
            "(osgi.http.whiteboard.context.httpservice=*)"})
public class MyFilter implements Filter
```

It is up to the implementation on how the ServletContextHelper services for the Http Context are handled. It is not required that these are actually registered with the service registry, the matching might be done internally by the implementation. As the above filter might match more than one ServletContextHelper, it should be registered with the prototype scope as outlined in the Http Whiteboard Specification.

In the same way~~, servlets, resources,~~ error pages and listeners can be associated with the Http Contexts managed by the Http Service. As the Http Service defines that the first servlet or resource for a path wins, this isn't compatible with the way the whiteboard implementation would handle that case. Instead of defining various special cases to handle this, servlets and resources can't be associated with an Http Context managed by the Http Service. If it is, this is handled as an error.

~~However for servlets and resources additional special rules apply: the Http Service defines that the servlets and resources share a single namespace and the first registration for a pattern wins. In contrast the Http Whiteboard specification defines that the servlet or resource with the highest ranking wins. To not break the contract of the Http Service, for the Http Contexts representing the space of the Http Service, the rule of the Http Service Specification is enforced: if there is already a servlet or resource for a pattern, this one is continued to be used. The whiteboard service is ignored. This should be logged as a warning with the log service if available. Likewise if a servlet or resource is unregistered from these contexts, other whiteboard services with the same pattern that have previously being ignored are not registered automatically. Any servlet or resource appearing later in time might be registered.~~

~~Question (CZ): Is the special casing worth it? The only other option is to not support servlets or resources – which is another special case. We have the use case for registering whiteboard servlets with the Http Service.~~

5.2 Request Preprocessing

A new service markre interface Preprocessor allows to register services using a whiteboard pattern. The interface extends the Filter interfaces. Services of this type are always run before request dispatching is performed. If there are several services of this type, they are run in order of there service ranking, the one with the highest ranking is used first. In the case of a service ranking tie, the servlet filter with the lowest service.id is processed first.

The preprocessor is handled in the same way as filters, e.g init and destroy are called etc. However as these preprocessors are run before dispatching and therefore the targetted servlet context is not known yet, FilterConfig.getServletContext returns the servlet context of the backing implementation. The same context is returned by the request object. The context path is the context path of this underlying servlet context.

~~The service interface is modeled after the Servlet Filter interface:~~

```
@ConsumerType
public interface Preprocessor extends Filter {

    public void doFilter(ServletRequest request, ServletResponse response,
                         PreprocessorChain chain)
    throws IOException, ServletException;
}
```

The passed in chain can be used to invoke the next preprocessor in the chain, or if the end of that chain is reached to start dispatching of the request. A preprocessor might decide to terminate the processing and directly generate a response.

5.3 Multipart Configuration Handling

Support for multipart configuration is enabled on a Servlet by setting the `osgi.http.whiteboard.servlet.multipart.enabled` equal to `true`.

Further refinement of the Servlet's multipart configuration can be made with the following properties:

- `osgi.http.whiteboard.servlet.multipart.maxFileSize` (Long)
  - the maximum size of a file being uploaded
- `osgi.http.whiteboard.servlet.multipart.location` (String)
  - the location where the files can be stored on disk
  - The value "is interpreted as an absolute path and defaults to the value of the javax.servlet.context.tempdir. If a relative path is specified, it will be relative to the tempdir location. The test for absolute path vs relative path MUST be done via java.io.File.isAbsolute."
  - Q1: I believe a `FilePermission(location, "read, write")` security check should be made during servlet registration resulting in a `FailedServletDTO`; as opposed to a lazy check during runtime. Thoughts?
  - Q2: If Q1, should the `FailedServletDTO` use a new, distinct reason code?
- `osgi.http.whiteboard.servlet.multipart.maxRequestSize` (Long)
  - the maximum request size
- `osgi.http.whiteboard.servlet.multipart.fileSizeThreshold` (Integer)
  - the size threshold after which the file will be written to disk

The ServletDTO and FailedServletDTO should reflect these settings.

5.4 Support Servlets without a pattern

The requirements for a whiteboard servlet in section 140.4 is changed from requiring a configured pattern using the property `osgi.http.whiteboard.servlet.pattern`. The servlet must have at least one valid value for one of these properties:

- `osgi.http.whiteboard.servlet.pattern`
- `osgi.http.whiteboard.servlet.name`
- `osgi.http.whiteboard.servlet.errorPage`

5.5 ~~Capabilities~~

~~The Http Service implementation bundle must provide the osgi.implementation capability with name osgi.httpservice. This capability can be used by provisioning tools and during resolution to ensure that a Http Service is present. The capability must also declare a uses constraint for the org.osgi.service.http and servlet api packages and provide the version of this specification:~~

```
Provide-Capability: osgi.implementation;
                    osgi.implementation="osgi.httpservice";
                    uses:="org.osgi.service.http,javax.servlet,javax.servlet.http";
                    version:Version="1.3"
```

~~This capability must follow the rules defined for the osgi.implementation Namespace.~~

~~The bundle providing the Http Service must provide a capability in the osgi.service namespace representing this service. This capability must also declare a uses constraint for the org.osgi.service.http and the servlet api packages:~~

```
Provide-Capability: osgi.service;
                    objectClass:List<String>="org.osgi.service.http.HttpService";
                    uses:="org.osgi.service.http,javax.servlet,javax.servlet.http"
```

~~This capability must follow the rules defined for the osgi.service Namespace.~~

5.6 ~~Support for ServletContext logging~~

~~If a web component calls one of the various log() methods on the `ServletContext`, the logging is done through the implementation of the Http Whiteboard which might defer to the logging of the application server in bridged mode. In some cases its useful to log these log statements within the context of the web application.~~

~~A new method `boolean log(String, Throwable)` is added to the `ServletContextHelper`. The default implementation does nothing and returns `false`.~~

~~The implementation of the ServletContext#log methods - provided by the Http Whiteboard implementation – first calls the above log method of the ServletContextHelper. If it returns false, it continues with logging through the currently available mechanism. If it returns true, this means the ServletContextHelper took care of logging, and the Http Whiteboard implementation does not log that statement anymore.~~

6 Data Transfer Objects

RFC 185 defines Data Transfer Objects as a generic means for management solutions to interact with runtime entities in an OSGi Framework. DTOs provides a common, easily serializable representation of the technology.

For all new functionality added to the OSGi Framework the question should be asked: would this feature benefit from a DTO? The expectation is that in most cases it would.

The DTOs for the design in this RFC should be described here and if there are no DTOs being defined an explanation should be given explaining why this is not applicable in this case.

This section is optional and could also be provided in a separate RFC.

7 Javadoc

OSGi Javadoc

06/07/16 2:03 PM

Package Summary		Page
org.osgi.service.http.context	Http Whiteboard Context Package Version 1.0.	12
org.osgi.service.http.runtime	Http Runtime Package Version 1.0.	18
org.osgi.service.http.runtime.dto	Http Runtime DTO Package Version 1.0.	21
org.osgi.service.http.whiteboard	Http Whiteboard Package Version 1.0.	54

Package org.osgi.service.http.context

Http Whiteboard Context Package Version 1.0.

See:

Description

Class Summary		Page
ServletContextHelper	Helper service for a servlet context used by a Http Whiteboard implementation to serve HTTP requests.	13

Package org.osgi.service.http.context Description

Http Whiteboard Context Package Version 1.0.

Bundles wishing to use this package must list the package in the Import-Package header of the bundle's manifest. This package has two types of users: the consumers that use the API in this package and the providers that implement the API in this package.

Example import for consumers using the API in this package:

```
Import-Package: org.osgi.service.http.context; version="[1.0,2.0)"
```

Example import for providers implementing the API in this package:

```
Import-Package: org.osgi.service.http.context; version="[1.0,1.1)"
```

Class ServletContextHelper

org.osgi.service.http.context

```
java.lang.Object
  └ org.osgi.service.http.context.ServletContextHelper
```

```
abstract public class ServletContextHelper
extends Object
```

Helper service for a servlet context used by a Http Whiteboard implementation to serve HTTP requests.

This service defines methods that the Http Whiteboard implementation may call to get information for a request when dealing with whiteboard services.

Each `ServletContextHelper` is registered with a `"osgi.http.whiteboard.context.name"` service property containing a name to reference by servlets, servlet filters, resources, and listeners. If there is more than one `ServletContextHelper` registered with the same context name, the one with the highest service ranking is active, the others are inactive.

A context is registered with the `"osgi.http.whiteboard.context.path"` service property to define a path under which all services registered with this context are reachable. If there is more than one `ServletContextHelper` registered with the same path, each duplicate context path is searched by service ranking order according to `org.osgi.framework.ServiceReference.compareTo(Object)` until a matching servlet or resource is found.

Servlets, servlet filters, resources, and listeners services may be associated with a `ServletContextHelper` service with the `"osgi.http.whiteboard.context.select"` service property. If the referenced `ServletContextHelper` service does not exist or is currently not active, the whiteboard services for that `ServletContextHelper` are not active either.

If no `ServletContextHelper` service is associated, that is no `"osgi.http.whiteboard.context.select"` service property is configured for a whiteboard service, a default `ServletContextHelper` is used.

Those whiteboard services that are associated with the same `ServletContextHelper` object will share the same `ServletContext` object.

The behavior of the methods on the default `ServletContextHelper` is defined as follows:

- `getMimeType` - Always returns `null`.
- `handleSecurity` - Always returns `true`.
- `getResource` - Assumes the named resource is in the bundle of the whiteboard service, addressed from the root. This method calls the whiteboard service bundle's `Bundle.getEntry` method, and returns the appropriate URL to access the resource. On a Java runtime environment that supports permissions, the Http Whiteboard implementation needs to be granted `org.osgi.framework.AdminPermission[*,RESOURCE]`.
- `getResourcePaths` - Assumes that the resources are in the bundle of the whiteboard service. This method calls `Bundle.findEntries` method, and returns the found entries. On a Java runtime environment that supports permissions, the Http Whiteboard implementation needs to be granted `org.osgi.framework.AdminPermission[*,RESOURCE]`.
- `getRealPath` - Always returns `null`.

See Also:

`HttpWhiteboardConstants.HTTP_WHITEBOARD_CONTEXT_NAME`, `HttpWhiteboardConstants.HTTP_WHITEBOARD_CONTEXT_PATH`

ThreadSafe

Field Summary	Page

static String	**AUTHENTICATION_TYPE** HttpServletRequest attribute specifying the scheme used in authentication.	14
static String	**AUTHORIZATION** HttpServletRequest attribute specifying the Authorization object obtained from the org.osgi.service.useradmin.UserAdmin service.	14
static String	**REMOTE_USER** HttpServletRequest attribute specifying the name of the authenticated user.	14

Constructor Summary	Page
ServletContextHelper() Construct a new context helper.	15
ServletContextHelper(Bundle bundle) Construct a new context helper associated with the specified bundle.	15

Method Summary	Page
String **getMimeType**(String name) Maps a name to a MIME type.	16
String **getRealPath**(String path) Gets the real path corresponding to the given virtual path.	17
URL **getResource**(String name) Maps a resource name to a URL.	16
Set<String> **getResourcePaths**(String path) Returns a directory-like listing of all the paths to resources within the web application whose longest sub-path matches the supplied path argument.	16
boolean **handleSecurity**(HttpServletRequest request, HttpServletResponse response) Handles security for the specified request.	15

Field Detail

REMOTE_USER

```
public static final String REMOTE_USER = "org.osgi.service.http.authentication.remote.user"
```

HttpServletRequest attribute specifying the name of the authenticated user. The value of the attribute can be retrieved by HttpServletRequest.getRemoteUser.

AUTHENTICATION_TYPE

```
public static final String AUTHENTICATION_TYPE = "org.osgi.service.http.authentication.type"
```

HttpServletRequest attribute specifying the scheme used in authentication. The value of the attribute can be retrieved by HttpServletRequest.getAuthType.

AUTHORIZATION

```
public static final String AUTHORIZATION = "org.osgi.service.useradmin.authorization"
```

HttpServletRequest attribute specifying the Authorization object obtained from the org.osgi.service.useradmin.UserAdmin service. The value of the attribute can be retrieved by HttpServletRequest.getAttribute(ServletContextHelper.AUTHORIZATION) .

Constructor Detail

ServletContextHelper

```
public ServletContextHelper()
```

Construct a new context helper.

If needed, the subclass will have to handle the association with a specific bundle.

~~ServletContextHelper~~

```
public ServletContextHelper(Bundle bundle)
```

Construct a new context helper associated with the specified bundle.

Parameters:

`bundle` - The bundle to be associated with this context helper.

Method Detail

handleSecurity

```
public boolean handleSecurity(HttpServletRequest request,
                              HttpServletResponse response)
                       throws IOException
```

Handles security for the specified request.

The Http Whiteboard implementation calls this method prior to servicing the specified request. This method controls whether the request is processed in the normal manner or an error is returned.

If the request requires authentication and the `Authorization` header in the request is missing or not acceptable, then this method should set the `WWW-Authenticate` header in the response object, set the status in the response object to Unauthorized(401) and return `false`. See also RFC 2617: HTTP Authentication: Basic and Digest Access Authentication.

If the request requires a secure connection and the `getScheme` method in the request does not return 'https' or some other acceptable secure protocol, then this method should set the status in the response object to Forbidden(403) and return `false`.

When this method returns `false`, the Http Whiteboard implementation will send the response back to the client, thereby completing the request. When this method returns `true`, the Http Whiteboard implementation will proceed with servicing the request.

If the specified request has been authenticated, this method must set the `AUTHENTICATION_TYPE` request attribute to the type of authentication used, and the `REMOTE_USER` request attribute to the remote user (request attributes are set using the `setAttribute` method on the request). If this method does not perform any authentication, it must not set these attributes.

If the authenticated user is also authorized to access certain resources, this method must set the `AUTHORIZATION` request attribute to the `Authorization` object obtained from the `org.osgi.service.useradmin.UserAdmin` service.

The servlet responsible for servicing the specified request determines the authentication type and remote user by calling the `getAuthType` and `getRemoteUser` methods, respectively, on the request.

Parameters:

`request` - The HTTP request.
`response` - The HTTP response.

Returns:

`true` if the request should be serviced, `false` if the request should not be serviced and Http Whiteboard implementation will send the response back to the client.

Throws:

`IOException` - May be thrown by this method. If this occurs, the Http Whiteboard implementation will terminate the request and close the socket.

---

getResource

```
public URL getResource(String name)
```

Maps a resource name to a URL.

Called by the Http Whiteboard implementation to map the specified resource name to a URL. For servlets, the Http Whiteboard implementation will call this method to support the `ServletContext` methods `getResource` and `getResourceAsStream`. For resources, the Http Whiteboard implementation will call this method to locate the named resource.

The context can control from where resources come. For example, the resource can be mapped to a file in the bundle's persistent storage area via `BundleContext.getDataFile(name).toURI().toURL()` or to a resource in the context's bundle via `getClass().getResource(name)`

Parameters:

`name` - The name of the requested resource.

Returns:

A URL that a Http Whiteboard implementation can use to read the resource or `null` if the resource does not exist.

---

getMimeType

```
public String getMimeType(String name)
```

Maps a name to a MIME type.

Called by the Http Whiteboard implementation to determine the MIME type for the specified name. For whiteboard services, the Http Whiteboard implementation will call this method to support the `ServletContext` method `getMimeType`. For resource servlets, the Http Whiteboard implementation will call this method to determine the MIME type for the `Content-Type` header in the response.

Parameters:

`name` - The name for which to determine the MIME type.

Returns:

The MIME type (e.g. text/html) of the specified name or `null` to indicate that the Http Whiteboard implementation should determine the MIME type itself.

---

getResourcePaths

```
public Set<String> getResourcePaths(String path)
```

Returns a directory-like listing of all the paths to resources within the web application whose longest sub-path matches the supplied path argument.

Called by the Http Whiteboard implementation to support the `ServletContext` method `getResourcePaths` for whiteboard services.

Parameters:

`path` - The partial path used to match the resources, which must start with a /.

Returns:

A Set containing the directory listing, or `null` if there are no resources in the web application whose path begins with the supplied path.

---

getRealPath

```
public String getRealPath(String path)
```

Gets the real path corresponding to the given virtual path.

Called by the Http Whiteboard implementation to support the `ServletContext` method `getRealPath` for whiteboard services.

Parameters:

`path` - The virtual path to be translated to a real path.

Returns:

The real path, or `null` if the translation cannot be performed.

Package org.osgi.service.http.runtime

Http Runtime Package Version 1.0.

See:

Description

Interface Summary		Page
HttpServiceRuntime	The HttpServiceRuntime service represents the runtime information of an Http Whiteboard implementation.	19

Class Summary		Page
HttpServiceRuntimeConstants	Defines standard names for Http Runtime Service constants.	20

Package org.osgi.service.http.runtime Description

Http Runtime Package Version 1.0.

Bundles wishing to use this package must list the package in the Import-Package header of the bundle's manifest. This package has two types of users: the consumers that use the API in this package and the providers that implement the API in this package.

Example import for consumers using the API in this package:

```
Import-Package: org.osgi.service.http.runtime; version="[1.0,2.0)"
```

Example import for providers implementing the API in this package:

```
Import-Package: org.osgi.service.http.runtime; version="[1.0,1.1)"
```

Interface HttpServiceRuntime

org.osgi.service.http.runtime

public interface **HttpServiceRuntime**

The HttpServiceRuntime service represents the runtime information of an Http Whiteboard implementation.

It provides access to DTOs representing the current state of the service.

The HttpServiceRuntime service must be registered with the HttpServiceRuntimeConstants.HTTP_SERVICE_ENDPOINT service property.

ThreadSafe

Method Summary		Page
RequestInfoDTO	**calculateRequestInfoDTO**(String path) Return a request info DTO containing the services involved with processing a request for the specified path.	19
RuntimeDTO	**getRuntimeDTO**() Return the runtime DTO representing the current state.	19

Method Detail

getRuntimeDTO

RuntimeDTO **getRuntimeDTO**()

Return the runtime DTO representing the current state.

Returns:
The runtime DTO.

calculateRequestInfoDTO

RequestInfoDTO **calculateRequestInfoDTO**(String path)

Return a request info DTO containing the services involved with processing a request for the specified path.

Parameters:
path - The request path, relative to the root of the Http Whiteboard implementation.
Returns:
The request info DTO for the specified path.

Class HttpServiceRuntimeConstants

org.osgi.service.http.runtime

```
java.lang.Object
  └ org.osgi.service.http.runtime.HttpServiceRuntimeConstants
```

```
final public class HttpServiceRuntimeConstants
extends Object
```

Defines standard names for Http Runtime Service constants.

Field Summary		Page
static String	**HTTP_SERVICE_ENDPOINT** Http Runtime Service service property specifying the endpoints upon which the Http Whiteboard implementation is listening.	20
static String	**HTTP_SERVICE_ID** Http Runtime Service service property to associate the Http Runtime Service with one or more HttpService services.	20

Field Detail

HTTP_SERVICE_ENDPOINT

```
public static final String HTTP_SERVICE_ENDPOINT = "osgi.http.endpoint"
```

Http Runtime Service service property specifying the endpoints upon which the Http Whiteboard implementation is listening.

An endpoint value is a URL or a relative path, to which the Http Whiteboard implementation is listening. For example, `http://192.168.1.10:8080/` or `/myapp/`. A relative path may be used if the scheme and authority parts of the URL are not known, e.g. in a bridged Http Whiteboard implementation. If the Http Whiteboard implementation is serving the root context and neither scheme nor authority is known, the value of the property is "/". Both, a URL and a relative path, must end with a slash.

An Http Whiteboard implementation can be listening on multiple endpoints.

The value of this service property must be of type `String`, `String[]`, or `Collection<String>`.

HTTP_SERVICE_ID

```
public static final String HTTP_SERVICE_ID = "osgi.http.service.id"
```

Http Runtime Service service property to associate the Http Runtime Service with one or more HttpService services.

If this Http Whiteboard implementation also implements the Http Service Specification, this service property is set to a collection of `service.id` for the `HttpService` services registered by this implementation.

The value of this service property must be of type `Collection<Long>`.

Package org.osgi.service.http.runtime.dto

Http Runtime DTO Package Version 1.0.

See:

Description

Class Summary		Page
BaseServletDTO	Represents common information about a `javax.servlet.Servlet` service.	23
DTOConstants	Defines standard constants for the DTOs.	25
ErrorPageDTO	Represents a `javax.servlet.Servlet` for handling errors and currently being used by a servlet context.	27
FailedErrorPageDTO	Represents a `javax.servlet.Servlet` service registered as an error page but currently not being used by a servlet context due to a problem.	28
FailedFilterDTO	Represents a servlet `Filter` service which is currently not being used by a servlet context due to a problem.	30
FailedListenerDTO	Represents a listener service which is currently not being used by a servlet context due to a problem.	32
FailedResourceDTO	Represents a resource definition which is currently not being used by a servlet context due to a problem.	33
FailedServletContextDTO	Represents a servlet context that is currently not used due to some problem.	34
FailedServletDTO	Represents a `javax.servlet.Servlet` service which is currently not being used by a servlet context due to a problem.	36
FilterDTO	Represents a servlet `javax.servlet.Filter` service currently being used for by a servlet context.	38
ListenerDTO	Represents a listener currently being used by a servlet context.	41
RequestInfoDTO	Represents the services used to process a specific request.	43
ResourceDTO	Represents a resource definition currently being used by a servlet context.	45
RuntimeDTO	Represents the state of a Http Service Runtime.	47
ServletContextDTO	Represents a `javax.servlet.ServletContext` created for servlets, resources, servlet Filters, and listeners associated with that servlet context.	49
ServletDTO	Represents a `javax.servlet.Servlet` currently being used by a servlet context.	52

Package org.osgi.service.http.runtime.dto Description

Http Runtime DTO Package Version 1.0.

Bundles wishing to use this package must list the package in the Import-Package header of the bundle's manifest. This package has two types of users: the consumers that use the API in this package and the providers that implement the API in this package.

Example import for consumers using the API in this package:

```
Import-Package: org.osgi.service.http.runtime.dto; version="[1.0,2.0)"
```

Example import for providers implementing the API in this package:

```
Import-Package: org.osgi.service.http.runtime.dto; version="[1.0,1.1)"
```

Class BaseServletDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.BaseServletDTO
```

Direct Known Subclasses:
ErrorPageDTO, ServletDTO

```
abstract public class BaseServletDTO
extends DTO
```

Represents common information about a `javax.servlet.Servlet` service.

NotThreadSafe

Field Summary	Page
boolean **asyncSupported** Specifies whether the servlet supports asynchronous processing.	24
Map<String, String> **initParams** The servlet initialization parameters as provided during registration of the servlet.	24
String **name** The name of the servlet.	23
long **serviceId** Service property identifying the servlet.	24
long **servletContextId** The service id of the servlet context for the servlet represented by this DTO.	24
String **servletInfo** The information string from the servlet.	23

Constructor Summary	Page
BaseServletDTO()	24

Field Detail

name

```
public String name
```

The name of the servlet. This value is never `null`.

servletInfo

```
public String servletInfo
```

The information string from the servlet.

This is the value returned by the `Servlet.getServletInfo()` method.

---

asyncSupported

```
public boolean asyncSupported
```

Specifies whether the servlet supports asynchronous processing.

initParams

```
public Map<String,String> initParams
```

The servlet initialization parameters as provided during registration of the servlet. Additional parameters like the Http Service Runtime attributes are not included. If the service has no initialization parameters, the map is empty.

servletContextId

```
public long servletContextId
```

The service id of the servlet context for the servlet represented by this DTO.

serviceId

```
public long serviceId
```

Service property identifying the servlet. In the case of a servlet registered in the service registry and picked up by a Http Whiteboard Implementation, this value is not negative and corresponds to the service id in the registry. If the servlet has not been registered in the service registry, the value is negative and a unique negative value is generated by the Http Service Runtime in this case.

Constructor Detail

BaseServletDTO

```
public BaseServletDTO()
```

Class DTOConstants

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ org.osgi.service.http.runtime.dto.DTOConstants
```

```
final public class DTOConstants
extends Object
```

Defines standard constants for the DTOs.

Field Summary		Page
static int	**FAILURE_REASON_EXCEPTION_ON_INIT** An exception occurred during initializing of the service.	26
static int	**FAILURE_REASON_NO_SERVLET_CONTEXT_MATCHING** No matching `ServletContextHelper`.	25
static int	**FAILURE_REASON_SERVICE_IN_USE** The service is not registered as a prototype scoped service and is already in use with a servlet context and therefore can't be used with another servlet context.	26
static int	**FAILURE_REASON_SERVICE_NOT_GETTABLE** The service is registered in the service registry but getting the service fails as it returns `null`.	26
static int	**FAILURE_REASON_SERVLET_CONTEXT_FAILURE** Matching `ServletContextHelper`, but the context is not used due to a problem with the context.	25
static int	**FAILURE_REASON_SHADOWED_BY_OTHER_SERVICE** Service is shadowed by another service.	26
static int	**FAILURE_REASON_UNKNOWN** Failure reason is unknown.	25
static int	**FAILURE_REASON_VALIDATION_FAILED** The service is registered in the service registry but the service properties are invalid.	26

Field Detail

FAILURE_REASON_UNKNOWN

```
public static final int FAILURE_REASON_UNKNOWN = 0
```

Failure reason is unknown.

FAILURE_REASON_NO_SERVLET_CONTEXT_MATCHING

```
public static final int FAILURE_REASON_NO_SERVLET_CONTEXT_MATCHING = 1
```

No matching `ServletContextHelper`.

FAILURE_REASON_SERVLET_CONTEXT_FAILURE

```
public static final int FAILURE_REASON_SERVLET_CONTEXT_FAILURE = 2
```

Matching `ServletContextHelper`, but the context is not used due to a problem with the context.

FAILURE_REASON_SHADOWED_BY_OTHER_SERVICE

```
public static final int FAILURE_REASON_SHADOWED_BY_OTHER_SERVICE = 3
```

Service is shadowed by another service.

For example, a service with the same service properties but a higher service ranking.

FAILURE_REASON_EXCEPTION_ON_INIT

```
public static final int FAILURE_REASON_EXCEPTION_ON_INIT = 4
```

An exception occurred during initializing of the service.

This reason can only happen for servlets and servlet filters.

FAILURE_REASON_SERVICE_NOT_GETTABLE

```
public static final int FAILURE_REASON_SERVICE_NOT_GETTABLE = 5
```

The service is registered in the service registry but getting the service fails as it returns `null`.

FAILURE_REASON_VALIDATION_FAILED

```
public static final int FAILURE_REASON_VALIDATION_FAILED = 6
```

The service is registered in the service registry but the service properties are invalid.

FAILURE_REASON_SERVICE_IN_USE

```
public static final int FAILURE_REASON_SERVICE_IN_USE = 7
```

The service is not registered as a prototype scoped service and is already in use with a servlet context and therefore can't be used with another servlet context.

Class ErrorPageDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.BaseServletDTO
          └ org.osgi.service.http.runtime.dto.ErrorPageDTO
```

Direct Known Subclasses:
FailedErrorPageDTO

```
public class ErrorPageDTO
extends BaseServletDTO
```

Represents a `javax.servlet.Servlet` for handling errors and currently being used by a servlet context.

NotThreadSafe

Field Summary	Page
long[] **errorCodes** The error codes the error page is used for.	27
String[] **exceptions** The exceptions the error page is used for.	27

Fields inherited from class org.osgi.service.http.runtime.dto.BaseServletDTO
asyncSupported, initParams, name, serviceId, servletContextId, servletInfo

Constructor Summary	Page
ErrorPageDTO()	27

Field Detail

exceptions

```
public String[] exceptions
```

The exceptions the error page is used for. This array might be empty.

errorCodes

```
public long[] errorCodes
```

The error codes the error page is used for. This array might be empty.

Constructor Detail

ErrorPageDTO

```
public ErrorPageDTO()
```

Class FailedErrorPageDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.BaseServletDTO
          └ org.osgi.service.http.runtime.dto.ErrorPageDTO
              └ org.osgi.service.http.runtime.dto.FailedErrorPageDTO
```

```
public class FailedErrorPageDTO
extends ErrorPageDTO
```

Represents a `javax.servlet.Servlet` service registered as an error page but currently not being used by a servlet context due to a problem.

As the servlet represented by this DTO is not used due to a failure, the field `BaseServletDTO.servletContextId` always returns `0` and does not point to an existing `ServletContextHelper`.

NotThreadSafe

Field Summary		Page
int	**failureReason** The reason why the servlet represented by this DTO is not used.	28

Fields inherited from class org.osgi.service.http.runtime.dto.ErrorPageDTO
errorCodes, exceptions

Fields inherited from class org.osgi.service.http.runtime.dto.BaseServletDTO
asyncSupported, initParams, name, serviceId, servletContextId, servletInfo

Constructor Summary	Page
FailedErrorPageDTO()	29

Field Detail

failureReason

```
public int failureReason
```

The reason why the servlet represented by this DTO is not used.

See Also:

DTOConstants.FAILURE_REASON_UNKNOWN,
DTOConstants.FAILURE_REASON_EXCEPTION_ON_INIT,
DTOConstants.FAILURE_REASON_NO_SERVLET_CONTEXT_MATCHING,
DTOConstants.FAILURE_REASON_SERVICE_NOT_GETTABLE,
DTOConstants.FAILURE_REASON_SERVLET_CONTEXT_FAILURE,
DTOConstants.FAILURE_REASON_SHADOWED_BY_OTHER_SERVICE

Constructor Detail

FailedErrorPageDTO

```
public FailedErrorPageDTO()
```

Class FailedFilterDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.FilterDTO
          └ org.osgi.service.http.runtime.dto.FailedFilterDTO
```

```
public class FailedFilterDTO
extends FilterDTO
```

Represents a servlet `Filter` service which is currently not being used by a servlet context due to a problem.

As the service represented by this DTO is not used due to a failure, the field `FilterDTO.servletContextId` always returns `0` and does not point to an existing servlet context.

NotThreadSafe

Field Summary	Page
int **failureReason** The reason why the servlet filter represented by this DTO is not used.	30

Fields inherited from class org.osgi.service.http.runtime.dto.FilterDTO
asyncSupported, dispatcher, initParams, name, patterns, regexs, serviceId, servletContextId, servletNames

Constructor Summary	Page
FailedFilterDTO()	31

Field Detail

failureReason

```
public int failureReason
```

The reason why the servlet filter represented by this DTO is not used.

See Also:

DTOConstants.FAILURE_REASON_UNKNOWN,
DTOConstants.FAILURE_REASON_EXCEPTION_ON_INIT,
DTOConstants.FAILURE_REASON_NO_SERVLET_CONTEXT_MATCHING,
DTOConstants.FAILURE_REASON_SERVICE_NOT_GETTABLE,
DTOConstants.FAILURE_REASON_SERVLET_CONTEXT_FAILURE,
DTOConstants.FAILURE_REASON_SHADOWED_BY_OTHER_SERVICE

Constructor Detail

FailedFilterDTO

```
public FailedFilterDTO()
```

Class FailedListenerDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.ListenerDTO
          └ org.osgi.service.http.runtime.dto.FailedListenerDTO
```

```
public class FailedListenerDTO
extends ListenerDTO
```

Represents a listener service which is currently not being used by a servlet context due to a problem.

As the listener represented by this DTO is not used due to a failure, the field BaseServletDTO.servletContextId always returns 0 and does not point to an existing servlet context.

NotThreadSafe

Field Summary		Page
int	**failureReason** The reason why the listener represented by this DTO is not used.	32

Fields inherited from class org.osgi.service.http.runtime.dto.ListenerDTO
serviceId, servletContextId, types

Constructor Summary	Page
FailedListenerDTO()	32

Field Detail

failureReason

```
public int failureReason
```

The reason why the listener represented by this DTO is not used.

See Also:

DTOConstants.FAILURE_REASON_UNKNOWN,
DTOConstants.FAILURE_REASON_EXCEPTION_ON_INIT,
DTOConstants.FAILURE_REASON_NO_SERVLET_CONTEXT_MATCHING,
DTOConstants.FAILURE_REASON_SERVICE_NOT_GETTABLE,
DTOConstants.FAILURE_REASON_SERVLET_CONTEXT_FAILURE,
DTOConstants.FAILURE_REASON_SHADOWED_BY_OTHER_SERVICE

Constructor Detail

FailedListenerDTO

```
public FailedListenerDTO()
```

Class FailedResourceDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.ResourceDTO
          └ org.osgi.service.http.runtime.dto.FailedResourceDTO
```

```
public class FailedResourceDTO
extends ResourceDTO
```

Represents a resource definition which is currently not being used by a servlet context due to a problem.

As the resource represented by this DTO is not used due to a failure, the field `ResourceDTO.servletContextId` always returns `0` and does not point to an existing servlet context.

NotThreadSafe

Field Summary		Page
int	**failureReason** The reason why the resource represented by this DTO is not used.	33

Fields inherited from class org.osgi.service.http.runtime.dto.ResourceDTO
patterns, prefix, serviceId, servletContextId

Constructor Summary	Page
FailedResourceDTO()	33

Field Detail

failureReason

```
public int failureReason
```

The reason why the resource represented by this DTO is not used.

See Also:

DTOConstants.FAILURE_REASON_UNKNOWN,
DTOConstants.FAILURE_REASON_EXCEPTION_ON_INIT,
DTOConstants.FAILURE_REASON_NO_SERVLET_CONTEXT_MATCHING,
DTOConstants.FAILURE_REASON_SERVICE_NOT_GETTABLE,
DTOConstants.FAILURE_REASON_SERVLET_CONTEXT_FAILURE,
DTOConstants.FAILURE_REASON_SHADOWED_BY_OTHER_SERVICE

Constructor Detail

FailedResourceDTO

```
public FailedResourceDTO()
```

Class FailedServletContextDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.ServletContextDTO
          └ org.osgi.service.http.runtime.dto.FailedServletContextDTO
```

```
public class FailedServletContextDTO
extends ServletContextDTO
```

Represents a servlet context that is currently not used due to some problem. The following fields return an empty array for a `FailedServletContextDTO`:

- `ServletContextDTO.servletDTOs`
- `ServletContextDTO.resourceDTOs`
- `ServletContextDTO.filterDTOs`
- `ServletContextDTO.errorPageDTOs`
- `ServletContextDTO.listenerDTOs`

The method `ServletContextDTO.attributes` returns an empty map for a `FailedServletContextDTO`.

NotThreadSafe

Field Summary	Page
int **failureReason** The reason why the servlet context represented by this DTO is not used.	34

Fields inherited from class org.osgi.service.http.runtime.dto.ServletContextDTO
attributes, contextPath, errorPageDTOs, filterDTOs, initParams, listenerDTOs, name, resourceDTOs, serviceId, servletDTOs

Constructor Summary	Page
FailedServletContextDTO()	35

Field Detail

failureReason

```
public int failureReason
```

The reason why the servlet context represented by this DTO is not used.

See Also:

DTOConstants.FAILURE_REASON_UNKNOWN,
DTOConstants.FAILURE_REASON_EXCEPTION_ON_INIT,
DTOConstants.FAILURE_REASON_NO_SERVLET_CONTEXT_MATCHING,
DTOConstants.FAILURE_REASON_SERVICE_NOT_GETTABLE,
DTOConstants.FAILURE_REASON_SERVLET_CONTEXT_FAILURE,
DTOConstants.FAILURE_REASON_SHADOWED_BY_OTHER_SERVICE

Constructor Detail

FailedServletContextDTO

```
public FailedServletContextDTO()
```

Class FailedServletDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.BaseServletDTO
          └ org.osgi.service.http.runtime.dto.ServletDTO
              └ org.osgi.service.http.runtime.dto.FailedServletDTO
```

```
public class FailedServletDTO
extends ServletDTO
```

Represents a `javax.servlet.Servlet` service which is currently not being used by a servlet context due to a problem.

As the servlet represented by this DTO is not used due to a failure, the field `BaseServletDTO.servletContextId` always returns `0` and does not point to an existing servlet context.

NotThreadSafe

Field Summary	Page
int **failureReason** The reason why the servlet represented by this DTO is not used.	36

Fields inherited from class org.osgi.service.http.runtime.dto.ServletDTO
multipartEnabled, multipartFileSizeThreshold, multipartLocation, multipartMaxFileSize, multipartMaxRequestSize, patterns

Fields inherited from class org.osgi.service.http.runtime.dto.BaseServletDTO
asyncSupported, initParams, name, serviceId, servletContextId, servletInfo

Constructor Summary	Page
FailedServletDTO()	37

Field Detail

failureReason

```
public int failureReason
```

The reason why the servlet represented by this DTO is not used.

See Also:

DTOConstants.FAILURE_REASON_UNKNOWN,
DTOConstants.FAILURE_REASON_EXCEPTION_ON_INIT,
DTOConstants.FAILURE_REASON_NO_SERVLET_CONTEXT_MATCHING,
DTOConstants.FAILURE_REASON_SERVICE_NOT_GETTABLE,
DTOConstants.FAILURE_REASON_SERVLET_CONTEXT_FAILURE,
DTOConstants.FAILURE_REASON_SHADOWED_BY_OTHER_SERVICE

Constructor Detail

FailedServletDTO

```
public FailedServletDTO()
```

Class FilterDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.FilterDTO
```

Direct Known Subclasses:
FailedFilterDTO

```
public class FilterDTO
extends DTO
```

Represents a servlet `javax.servlet.Filter` service currently being used for by a servlet context.

NotThreadSafe

Field Summary		Page
boolean	**asyncSupported** Specifies whether the servlet filter supports asynchronous processing.	39
String[]	**dispatcher** The dispatcher associations for the servlet filter.	39
Map<String, String>	**initParams** The servlet filter initialization parameters as provided during registration of the servlet filter.	39
String	**name** The name of the servlet filter.	38
String[]	**patterns** The request mappings for the servlet filter.	39
String[]	**regexs** The request mappings for the servlet filter.	39
long	**serviceId** Service property identifying the servlet filter.	39
long	**servletContextId** The service id of the servlet context for the servlet filter represented by this DTO.	39
String[]	**servletNames** The servlet names for the servlet filter.	39

Constructor Summary	Page
FilterDTO()	40

Field Detail

name

```
public String name
```

The name of the servlet filter. This field is never `null`.

patterns

```
public String[] patterns
```

The request mappings for the servlet filter.

The specified patterns are used to determine whether a request is mapped to the servlet filter. This array might be empty.

servletNames

```
public String[] servletNames
```

The servlet names for the servlet filter.

The specified names are used to determine the servlets whose requests are mapped to the servlet filter. This array might be empty.

regexs

```
public String[] regexs
```

The request mappings for the servlet filter.

The specified regular expressions are used to determine whether a request is mapped to the servlet filter. This array might be empty.

asyncSupported

```
public boolean asyncSupported
```

Specifies whether the servlet filter supports asynchronous processing.

dispatcher

```
public String[] dispatcher
```

The dispatcher associations for the servlet filter.

The specified names are used to determine in what occasions the servlet filter is called. This array is never `null`.

initParams

```
public Map<String,String> initParams
```

The servlet filter initialization parameters as provided during registration of the servlet filter. Additional parameters like the Http Service Runtime attributes are not included. If the servlet filter has not initialization parameters, this map is empty.

serviceId

```
public long serviceId
```

Service property identifying the servlet filter. In the case of a servlet filter registered in the service registry and picked up by a Http Whiteboard Implementation, this value is not negative and corresponds to the service id in the registry. If the servlet filter has not been registered in the service registry, the value is negative and a unique negative value is generated by the Http Service Runtime in this case.

servletContextId

```
public long servletContextId
```

The service id of the servlet context for the servlet filter represented by this DTO.

Constructor Detail

FilterDTO

```
public FilterDTO()
```

Class ListenerDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.ListenerDTO
```

Direct Known Subclasses:
FailedListenerDTO

```
public class ListenerDTO
extends DTO
```

Represents a listener currently being used by a servlet context.

NotThreadSafe

Field Summary		Page
long	**serviceId** Service property identifying the listener.	41
long	**servletContextId** The service id of the servlet context for the listener represented by this DTO.	41
String[]	**types** The fully qualified type names the listener.	41

Constructor Summary	Page
ListenerDTO()	42

Field Detail

types

```
public String[] types
```

The fully qualified type names the listener. This array is never empty.

serviceId

```
public long serviceId
```

Service property identifying the listener. In the case of a Listener registered in the service registry and picked up by a Http Whiteboard Implementation, this value is not negative and corresponds to the service id in the registry. If the listener has not been registered in the service registry, the value is negative and a unique negative value is generated by the Http Service Runtime in this case.

servletContextId

```
public long servletContextId
```

The service id of the servlet context for the listener represented by this DTO.

Constructor Detail

ListenerDTO

```
public ListenerDTO()
```

Class RequestInfoDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.RequestInfoDTO
```

```
public class RequestInfoDTO
extends DTO
```

Represents the services used to process a specific request.

NotThreadSafe

Field Summary		Page
FilterDTO[]	**filterDTOs** The servlet filters processing this request.	43
String	**path** The path of the request relative to the root.	43
ResourceDTO	**resourceDTO** The resource processing this request.	44
long	**servletContextId** The service id of the servlet context processing the request represented by this DTO.	43
ServletDTO	**servletDTO** The servlet processing this request.	44

Constructor Summary	Page
RequestInfoDTO()	44

Field Detail

path

```
public String path
```

The path of the request relative to the root.

servletContextId

```
public long servletContextId
```

The service id of the servlet context processing the request represented by this DTO.

filterDTOs

```
public FilterDTO[] filterDTOs
```

The servlet filters processing this request. If no servlet filters are called for processing this request, an empty array is returned.

---

servletDTO

```
public ServletDTO servletDTO
```

The servlet processing this request. If the request is processed by a servlet, this field points to the DTO of the servlet. If the request is processed by another type of component like a resource, this field is `null`.

resourceDTO

```
public ResourceDTO resourceDTO
```

The resource processing this request. If the request is processed by a resource, this field points to the DTO of the resource. If the request is processed by another type of component like a servlet, this field is `null`.

Constructor Detail

RequestInfoDTO

```
public RequestInfoDTO()
```

Class ResourceDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.ResourceDTO
```

Direct Known Subclasses:
FailedResourceDTO

```
public class ResourceDTO
extends DTO
```

Represents a resource definition currently being used by a servlet context.

NotThreadSafe

Field Summary		Page
String[]	**patterns** The request mappings for the resource.	45
String	**prefix** The prefix of the resource.	45
long	**serviceId** Service property identifying the resource.	45
long	**servletContextId** The service id of the servlet context for the resource represented by this DTO.	46

Constructor Summary	Page
ResourceDTO()	46

Field Detail

patterns

```
public String[] patterns
```

The request mappings for the resource.

The specified patterns are used to determine whether a request is mapped to the resource. This value is never `null`.

prefix

```
public String prefix
```

The prefix of the resource.

serviceId

```
public long serviceId
```

Service property identifying the resource. In the case of a resource registered in the service registry and picked up by a Http Whiteboard Implementation, this value is not negative and corresponds to the service id in the registry. If the resource has not been registered in the service registry, the value is negative and a unique negative value is generated by the Http Service Runtime in this case.

servletContextId

```
public long servletContextId
```

The service id of the servlet context for the resource represented by this DTO.

Constructor Detail

ResourceDTO

```
public ResourceDTO()
```

Class RuntimeDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.RuntimeDTO
```

```
public class RuntimeDTO
extends DTO
```

Represents the state of a Http Service Runtime.

NotThreadSafe

Field Summary		Page
FailedErrorPageDTO[]	**failedErrorPageDTOs** Returns the representations of the error page `javax.servlet.Servlet` services associated with this runtime but currently not used due to some problem.	48
FailedFilterDTO[]	**failedFilterDTOs** Returns the representations of the servlet `javax.servlet.Filter` services associated with this runtime but currently not used due to some problem.	48
FailedListenerDTO[]	**failedListenerDTOs** Returns the representations of the listeners associated with this runtime but currently not used due to some problem.	48
FailedResourceDTO[]	**failedResourceDTOs** Returns the representations of the resources associated with this runtime but currently not used due to some problem.	48
FailedServletContextDTO[]	**failedServletContextDTOs** Returns the representations of the `javax.servlet.ServletContext` objects currently not used by the Http service runtime due to some problem.	48
FailedServletDTO[]	**failedServletDTOs** Returns the representations of the `javax.servlet.Servlet` services associated with this runtime but currently not used due to some problem.	48
ServiceReferenceDTO	**serviceDTO** The DTO for the corresponding `org.osgi.service.http.runtime.HttpServiceRuntime`.	47
ServletContextDTO[]	**servletContextDTOs** Returns the representations of the `javax.servlet.ServletContext` objects used by the Http Service Runtime.	48

Constructor Summary	Page
RuntimeDTO()	48

Field Detail

serviceDTO

```
public ServiceReferenceDTO serviceDTO
```

The DTO for the corresponding `org.osgi.service.http.runtime.HttpServiceRuntime`. This value is never `null`.

servletContextDTOs

```
public ServletContextDTO[] servletContextDTOs
```

Returns the representations of the `javax.servlet.ServletContext` objects used by the Http Service Runtime. The returned array may be empty if the Http Service Runtime is currently not using any `javax.servlet.ServletContext` objects.

failedServletContextDTOs

```
public FailedServletContextDTO[] failedServletContextDTOs
```

Returns the representations of the `javax.servlet.ServletContext` objects currently not used by the Http service runtime due to some problem. The returned array may be empty.

failedServletDTOs

```
public FailedServletDTO[] failedServletDTOs
```

Returns the representations of the `javax.servlet.Servlet` services associated with this runtime but currently not used due to some problem. The returned array may be empty.

failedResourceDTOs

```
public FailedResourceDTO[] failedResourceDTOs
```

Returns the representations of the resources associated with this runtime but currently not used due to some problem. The returned array may be empty.

failedFilterDTOs

```
public FailedFilterDTO[] failedFilterDTOs
```

Returns the representations of the servlet `javax.servlet.Filter` services associated with this runtime but currently not used due to some problem. The returned array may be empty.

failedErrorPageDTOs

```
public FailedErrorPageDTO[] failedErrorPageDTOs
```

Returns the representations of the error page `javax.servlet.Servlet` services associated with this runtime but currently not used due to some problem. The returned array may be empty.

failedListenerDTOs

```
public FailedListenerDTO[] failedListenerDTOs
```

Returns the representations of the listeners associated with this runtime but currently not used due to some problem. The returned array may be empty.

Constructor Detail

RuntimeDTO

```
public RuntimeDTO()
```

Class ServletContextDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.ServletContextDTO
```

Direct Known Subclasses:
FailedServletContextDTO

```
public class ServletContextDTO
extends DTO
```

Represents a `javax.servlet.ServletContext` created for servlets, resources, servlet Filters, and listeners associated with that servlet context. The Servlet Context is usually backed by a `ServletContextHelper` service.

NotThreadSafe

Field Summary		Page
Map<String, Object>	**attributes** The servlet context attributes.	50
String	**contextPath** The servlet context path.	50
ErrorPageDTO[]	**errorPageDTOs** Returns the representations of the error page `Servlet` services associated with this context.	51
FilterDTO[]	**filterDTOs** Returns the representations of the servlet `Filter` services associated with this context.	50
Map<String, String>	**initParams** The servlet context initialization parameters.	50
ListenerDTO[]	**listenerDTOs** Returns the representations of the listener services associated with this context.	51
String	**name** The name of the servlet context.	50
ResourceDTO[]	**resourceDTOs** Returns the representations of the resource services associated with this context.	50
long	**serviceId** Service property identifying the servlet context.	50
ServletDTO[]	**servletDTOs** Returns the representations of the `Servlet` services associated with this context.	50

Constructor Summary	Page
ServletContextDTO()	51

Field Detail

name

public String **name**

The name of the servlet context. The name of the corresponding ServletContextHelper.

This is the value returned by the `ServletContext.getServletContextName()` method.

contextPath

public String **contextPath**

The servlet context path. This is the value returned by the `ServletContext.getContextPath()` method.

initParams

public Map<String,String> **initParams**

The servlet context initialization parameters. This is the set of parameters provided when registering this context. Additional parameters like the Http Service Runtime attributes are not included. If the context has no initialization parameters, this map is empty.

attributes

public Map<String,Object> **attributes**

The servlet context attributes.

The value type must be a numerical type, `Boolean`, `String`, `DTO` or an array of any of the former. Therefore this method will only return the attributes of the servlet context conforming to this constraint. Other attributes are omitted. If there are no attributes conforming to the constraint, an empty map is returned.

serviceId

public long **serviceId**

Service property identifying the servlet context. In the case of a servlet context backed by a `ServletContextHelper` registered in the service registry and picked up by a Http Whiteboard Implementation, this value is not negative and corresponds to the service id in the registry. If the servlet context is not backed by a service registered in the service registry, the value is negative and a unique negative value is generated by the Http Service Runtime in this case.

servletDTOs

public ServletDTO[] **servletDTOs**

Returns the representations of the `Servlet` services associated with this context. The representations of the `Servlet` services associated with this context. The returned array may be empty if this context is currently not associated with any `Servlet` services.

resourceDTOs

public ResourceDTO[] **resourceDTOs**

Returns the representations of the resource services associated with this context. The representations of the resource services associated with this context. The returned array may be empty if this context is currently not associated with any resource services.

filterDTOs

public FilterDTO[] **filterDTOs**

Returns the representations of the servlet `Filter` services associated with this context. The representations of the servlet `Filter` services associated with this context. The returned array may be empty if this context is currently not associated with any servlet `Filter` services.

errorPageDTOs

`public ErrorPageDTO[] errorPageDTOs`

Returns the representations of the error page `Servlet` services associated with this context. The representations of the error page `Servlet` services associated with this context. The returned array may be empty if this context is currently not associated with any error pages.

listenerDTOs

`public ListenerDTO[] listenerDTOs`

Returns the representations of the listener services associated with this context. The representations of the listener services associated with this context. The returned array may be empty if this context is currently not associated with any listener services.

Constructor Detail

ServletContextDTO

`public ServletContextDTO()`

Class ServletDTO

org.osgi.service.http.runtime.dto

```
java.lang.Object
  └ DTO
      └ org.osgi.service.http.runtime.dto.BaseServletDTO
          └ org.osgi.service.http.runtime.dto.ServletDTO
```

Direct Known Subclasses:
FailedServletDTO

```
public class ServletDTO
extends BaseServletDTO
```

Represents a `javax.servlet.Servlet` currently being used by a servlet context.

NotThreadSafe

Field Summary	Page
boolean **multipartEnabled** Specifies whether multipart support is enabled.	53
int **multipartFileSizeThreshold** Specifies the size threshold after which the file will be written to disk.	53
String **multipartLocation** Specifies the location where the files can be stored on disk.	53
long **multipartMaxFileSize** Specifies the maximum size of a file being uploaded.	53
long **multipartMaxRequestSize** Specifies the maximum request size.	53
String[] **patterns** The request mappings for the servlet.	52

Fields inherited from class org.osgi.service.http.runtime.dto.BaseServletDTO
asyncSupported, initParams, name, serviceId, servletContextId, servletInfo

Constructor Summary	Page
ServletDTO()	53

Field Detail

patterns

```
public String[] patterns
```

The request mappings for the servlet.

The specified patterns are used to determine whether a request is mapped to the servlet. This array is never empty.

multipartEnabled

```
public boolean multipartEnabled
```

Specifies whether multipart support is enabled.

multipartFileSizeThreshold

```
public int multipartFileSizeThreshold
```

Specifies the size threshold after which the file will be written to disk.

multipartLocation

```
public String multipartLocation
```

Specifies the location where the files can be stored on disk.

multipartMaxFileSize

```
public long multipartMaxFileSize
```

Specifies the maximum size of a file being uploaded.

multipartMaxRequestSize

```
public long multipartMaxRequestSize
```

Specifies the maximum request size.

Constructor Detail

ServletDTO

```
public ServletDTO()
```

Package org.osgi.service.http.whiteboard

Http Whiteboard Package Version 1.0.

See:

Description

Class Summary		Page
HttpWhiteboardConstants	Defines standard constants for the Http Whiteboard services.	55

Package org.osgi.service.http.whiteboard Description

Http Whiteboard Package Version 1.0.

Bundles wishing to use this package must list the package in the Import-Package header of the bundle's manifest. This package has two types of users: the consumers that use the API in this package and the providers that implement the API in this package.

Example import for consumers using the API in this package:

```
Import-Package: org.osgi.service.http.whiteboard; version="[1.0,2.0)"
```

Example import for providers implementing the API in this package:

```
Import-Package: org.osgi.service.http.whiteboard; version="[1.0,1.1)"
```

Class HttpWhiteboardConstants

org.osgi.service.http.whiteboard

```
java.lang.Object
  └ org.osgi.service.http.whiteboard.HttpWhiteboardConstants
```

```
final public class HttpWhiteboardConstants
extends Object
```

Defines standard constants for the Http Whiteboard services.

Field Summary		Page
static String	**DISPATCHER_ASYNC** Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied in the asynchronous context.	63
static String	**DISPATCHER_ERROR** Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied when an error page is called.	63
static String	**DISPATCHER_FORWARD** Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied to forward calls to the dispatcher.	62
static String	**DISPATCHER_INCLUDE** Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied to include calls to the dispatcher.	62
static String	**DISPATCHER_REQUEST** Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied to client requests.	62
static String	**HTTP_WHITEBOARD_CONTEXT_INIT_PARAM_PREFIX** Service property prefix referencing a ServletContextHelper service.	57
static String	**HTTP_WHITEBOARD_CONTEXT_NAME** Service property specifying the name of an ServletContextHelper service.	56
static String	**HTTP_WHITEBOARD_CONTEXT_PATH** Service property specifying the path of an ServletContextHelper service.	57
static String	**HTTP_WHITEBOARD_CONTEXT_SELECT** Service property referencing a ServletContextHelper service.	57
static String	**HTTP_WHITEBOARD_DEFAULT_CONTEXT_NAME** The name of the default ServletContextHelper.	57
static String	**HTTP_WHITEBOARD_FILTER_ASYNC_SUPPORTED** Service property specifying whether a servlet Filter service supports asynchronous processing.	61
static String	**HTTP_WHITEBOARD_FILTER_DISPATCHER** Service property specifying the dispatcher handling of a servlet Filter.	61
static String	**HTTP_WHITEBOARD_FILTER_INIT_PARAM_PREFIX** Service property prefix referencing a service.	62
static String	**HTTP_WHITEBOARD_FILTER_NAME** Service property specifying the servlet filter name of a Filter service.	60
static String	**HTTP_WHITEBOARD_FILTER_PATTERN** Service property specifying the request mappings for a Filter service.	60

static String	**HTTP_WHITEBOARD_FILTER_REGEX** Service property specifying the request mappings for a servlet `Filter` service.	61
static String	**HTTP_WHITEBOARD_FILTER_SERVLET** Service property specifying the servlet names for a servlet `Filter` service.	61
static String	**HTTP_WHITEBOARD_LISTENER** Service property to mark a Listener service as a Whiteboard service.	62
static String	**HTTP_WHITEBOARD_RESOURCE_PATTERN** Service property specifying the request mappings for resources.	63
static String	**HTTP_WHITEBOARD_RESOURCE_PREFIX** Service property specifying the resource entry prefix for a resource service.	63
static String	**HTTP_WHITEBOARD_SERVLET_ASYNC_SUPPORTED** Service property specifying whether a `Servlet` service supports asynchronous processing.	58
static String	**HTTP_WHITEBOARD_SERVLET_ERROR_PAGE** Service property specifying whether a `Servlet` service acts as an error page.	58
static String	**HTTP_WHITEBOARD_SERVLET_INIT_PARAM_PREFIX** Service property prefix referencing a service.	59
static String	**HTTP_WHITEBOARD_SERVLET_MULTIPART_ENABLED** Service property specifying whether a `Servlet` service has enabled multipart request processing.	59
static String	**HTTP_WHITEBOARD_SERVLET_MULTIPART_FILESIZETHRESHOLD** Service property specifying the size threshold after which the file will be written to disk.	59
static String	**HTTP_WHITEBOARD_SERVLET_MULTIPART_LOCATION** Service property specifying the location where the files can be stored on disk.	59
static String	**HTTP_WHITEBOARD_SERVLET_MULTIPART_MAXFILESIZE** Service property specifying the maximum size of a file being uploaded.	60
static String	**HTTP_WHITEBOARD_SERVLET_MULTIPART_MAXREQUESTSIZE** Service property specifying the maximum request size.	60
static String	**HTTP_WHITEBOARD_SERVLET_NAME** Service property specifying the servlet name of a `Servlet` service.	58
static String	**HTTP_WHITEBOARD_SERVLET_PATTERN** Service property specifying the request mappings for a `Servlet` service.	58
static String	**HTTP_WHITEBOARD_TARGET** Service property specifying the target filter to select the Http Whiteboard implementation to process the service.	64

Field Detail

HTTP_WHITEBOARD_CONTEXT_NAME

```
public static final String HTTP_WHITEBOARD_CONTEXT_NAME = "osgi.http.whiteboard.context.name"
```

Service property specifying the name of an `ServletContextHelper` service.

For `ServletContextHelper` services, this service property must be specified. Context services without this service property are ignored.

Servlet, listener, servlet filter, and resource services might refer to a specific `ServletContextHelper` service referencing the name with the `HTTP_WHITEBOARD_CONTEXT_SELECT` property.

For `ServletContextHelper` services, the value of this service property must be of type `String`. The value must follow the "symbolic-name" specification from Section 1.3.2 of the OSGi Core Specification.

See Also:

HTTP_WHITEBOARD_CONTEXT_PATH, HTTP_WHITEBOARD_CONTEXT_SELECT, HTTP_WHITEBOARD_DEFAULT_CONTEXT_NAME

HTTP_WHITEBOARD_DEFAULT_CONTEXT_NAME

```
public static final String HTTP_WHITEBOARD_DEFAULT_CONTEXT_NAME = "default"
```

The name of the default ServletContextHelper. If a service is registered with this property, it is overriding the default context with a custom provided context.

See Also:

HTTP_WHITEBOARD_CONTEXT_NAME

HTTP_WHITEBOARD_CONTEXT_PATH

```
public static final String HTTP_WHITEBOARD_CONTEXT_PATH = "osgi.http.whiteboard.context.path"
```

Service property specifying the path of an ServletContextHelper service.

For ServletContextHelper services this service property is required. Context services without this service property are ignored.

This property defines a context path under which all whiteboard services associated with this context are registered. Having different contexts with different paths allows to separate the URL space.

For ServletContextHelper services, the value of this service property must be of type `String`. The value is either a slash for the root or it must start with a slash but not end with a slash. Valid characters are defined in rfc3986#section-3.3. Contexts with an invalid path are ignored.

See Also:

HTTP_WHITEBOARD_CONTEXT_NAME, HTTP_WHITEBOARD_CONTEXT_SELECT

HTTP_WHITEBOARD_CONTEXT_INIT_PARAM_PREFIX

```
public static final String HTTP_WHITEBOARD_CONTEXT_INIT_PARAM_PREFIX = "context.init."
```

Service property prefix referencing a ServletContextHelper service.

For ServletContextHelper services this prefix can be used for service properties to mark them as initialization parameters which can be retrieved from the associated servlet context. The prefix is removed from the service property name to build the initialization parameter name.

For ServletContextHelper services, the value of each initialization parameter service property must be of type `String`.

HTTP_WHITEBOARD_CONTEXT_SELECT

```
public static final String HTTP_WHITEBOARD_CONTEXT_SELECT =
"osgi.http.whiteboard.context.select"
```

Service property referencing a ServletContextHelper service.

For servlet, listener, servlet filter, or resource services, this service property refers to the associated `ServletContextHelper` service. The value of this property is a filter expression which is matched against the service registration properties of the `ServletContextHelper` service. If this service property is not specified, the default context is used. If there is no context service matching, the servlet, listener, servlet filter, or resource service is ignored.

For example, if a whiteboard service wants to select a servlet context helper with the name "Admin" the expression would be "(osgi.http.whiteboard.context.name=Admin)". Selecting all contexts could be done with "(osgi.http.whiteboard.context.name=*)".

For servlet, listener, servlet filter, or resource services, the value of this service property must be of type `String`.

See Also:

HTTP_WHITEBOARD_CONTEXT_NAME, HTTP_WHITEBOARD_CONTEXT_PATH

HTTP_WHITEBOARD_SERVLET_NAME

```
public static final String HTTP_WHITEBOARD_SERVLET_NAME = "osgi.http.whiteboard.servlet.name"
```

Service property specifying the servlet name of a `Servlet` service.

This name is used as the value for the `ServletConfig.getServletName()` method. If this service property is not specified, the fully qualified name of the service object's class is used as the servlet name. Filter services may refer to servlets by this name in their HTTP_WHITEBOARD_FILTER_SERVLET service property to apply the filter to the servlet.

Servlet names should be unique among all servlet services associated with a single ServletContextHelper.

The value of this service property must be of type `String`.

HTTP_WHITEBOARD_SERVLET_PATTERN

```
public static final String HTTP_WHITEBOARD_SERVLET_PATTERN =
"osgi.http.whiteboard.servlet.pattern"
```

Service property specifying the request mappings for a `Servlet` service.

The specified patterns are used to determine whether a request should be mapped to the servlet. Servlet services without this service property or HTTP_WHITEBOARD_SERVLET_ERROR_PAGE are ignored.

The value of this service property must be of type `String`, `String[]`, or `Collection<String>`.

See Also:

"Java Servlet Specification Version 3.0, Section 12.2 Specification of Mappings"

HTTP_WHITEBOARD_SERVLET_ERROR_PAGE

```
public static final String HTTP_WHITEBOARD_SERVLET_ERROR_PAGE =
"osgi.http.whiteboard.servlet.errorPage"
```

Service property specifying whether a `Servlet` service acts as an error page.

The service property values may be the name of a fully qualified exception class, a three digit HTTP status code, the value "4xx" for all error codes in the 400 range, or the value "5xx" for all error codes in the 500 range. Any value that is not a three digit number, or one of the two special values is considered to be the name of a fully qualified exception class.

The value of this service property must be of type `String`, `String[]`, or `Collection<String>`.

HTTP_WHITEBOARD_SERVLET_ASYNC_SUPPORTED

```
public static final String HTTP_WHITEBOARD_SERVLET_ASYNC_SUPPORTED =
"osgi.http.whiteboard.servlet.asyncSupported"
```

Service property specifying whether a `Servlet` service supports asynchronous processing.

By default servlet services do not support asynchronous processing.

The value of this service property must be of type `Boolean`.

See Also:
"Java Servlet Specification Version 3.0, Section 2.3.3.3 Asynchronous Processing"

HTTP_WHITEBOARD_SERVLET_INIT_PARAM_PREFIX

```
public static final String HTTP_WHITEBOARD_SERVLET_INIT_PARAM_PREFIX = "servlet.init."
```

Service property prefix referencing a service.

For services this prefix can be used for service properties to mark them as initialization parameters which can be retrieved from the associated servlet config. The prefix is removed from the service property name to build the initialization parameter name.

For services, the value of each initialization parameter service property must be of type `String`.

HTTP_WHITEBOARD_SERVLET_MULTIPART_ENABLED

```
public static final String HTTP_WHITEBOARD_SERVLET_MULTIPART_ENABLED =
"osgi.http.whiteboard.servlet.multipart.enabled"
```

Service property specifying whether a `Servlet` service has enabled multipart request processing.

By default servlet services do not have multipart request processing enabled.

The value of this service property must be of type `Boolean`.

See Also:
"Java Servlet Specification Version 3.0, Section 8.1.5 @MultipartConfig"

HTTP_WHITEBOARD_SERVLET_MULTIPART_FILESIZETHRESHOLD

```
public static final String HTTP_WHITEBOARD_SERVLET_MULTIPART_FILESIZETHRESHOLD =
"osgi.http.whiteboard.servlet.multipart.fileSizeThreshold"
```

Service property specifying the size threshold after which the file will be written to disk.

When not set the default threshold is determined by the implementation.

The value of this service property must be of type `Integer`.

See Also:
"Java Servlet Specification Version 3.0, Section 14.4 Deployment Descriptor Diagram"

HTTP_WHITEBOARD_SERVLET_MULTIPART_LOCATION

```
public static final String HTTP_WHITEBOARD_SERVLET_MULTIPART_LOCATION =
"osgi.http.whiteboard.servlet.multipart.location"
```

Service property specifying the location where the files can be stored on disk.

When not set the default location is defined by the value of the system property "java.io.tmpdir".

The value of this service property must be of type `String`.

See Also:
"Java Servlet Specification Version 3.0, Section 14.4 Deployment Descriptor Diagram"

HTTP_WHITEBOARD_SERVLET_MULTIPART_MAXFILESIZE

```
public static final String HTTP_WHITEBOARD_SERVLET_MULTIPART_MAXFILESIZE =
"osgi.http.whiteboard.servlet.multipart.maxFileSize"
```

Service property specifying the maximum size of a file being uploaded.

When not set the default maximum size is -1 (no maximum size).

The value of this service property must be of type `Long`.

See Also:
"Java Servlet Specification Version 3.0, Section 14.4 Deployment Descriptor Diagram"

HTTP_WHITEBOARD_SERVLET_MULTIPART_MAXREQUESTSIZE

```
public static final String HTTP_WHITEBOARD_SERVLET_MULTIPART_MAXREQUESTSIZE =
"osgi.http.whiteboard.servlet.multipart.maxRequestSize"
```

Service property specifying the maximum request size.

When not set the default maximum request size is -1 (no maximum size).

The value of this service property must be of type `Long`.

See Also:
"Java Servlet Specification Version 3.0, Section 14.4 Deployment Descriptor Diagram"

HTTP_WHITEBOARD_FILTER_NAME

```
public static final String HTTP_WHITEBOARD_FILTER_NAME = "osgi.http.whiteboard.filter.name"
```

Service property specifying the servlet filter name of a `Filter` service.

This name is used as the value for the `FilterConfig.getFilterName()` method. If this service property is not specified, the fully qualified name of the service object's class is used as the servlet filter name.

Servlet filter names should be unique among all servlet filter services associated with a single `ServletContextHelper`.

The value of this service property must be of type `String`.

HTTP_WHITEBOARD_FILTER_PATTERN

```
public static final String HTTP_WHITEBOARD_FILTER_PATTERN =
"osgi.http.whiteboard.filter.pattern"
```

Service property specifying the request mappings for a `Filter` service.

The specified patterns are used to determine whether a request should be mapped to the servlet filter. Filter services without this service property or the `HTTP_WHITEBOARD_FILTER_SERVLET` or the `HTTP_WHITEBOARD_FILTER_REGEX` service property are ignored.

The value of this service property must be of type `String`, `String[]`, or `Collection<String>`.

See Also:
"Java Servlet Specification Version 3.0, Section 12.2 Specification of Mappings"

HTTP_WHITEBOARD_FILTER_SERVLET

```
public static final String HTTP_WHITEBOARD_FILTER_SERVLET =
"osgi.http.whiteboard.filter.servlet"
```

Service property specifying the servlet names for a servlet `Filter` service.

The specified names are used to determine the servlets whose requests should be mapped to the servlet filter. Servlet filter services without this service property or the HTTP_WHITEBOARD_FILTER_PATTERN or the HTTP_WHITEBOARD_FILTER_REGEX service property are ignored.

The value of this service property must be of type `String`, `String[]`, or `Collection<String>`.

HTTP_WHITEBOARD_FILTER_REGEX

```
public static final String HTTP_WHITEBOARD_FILTER_REGEX = "osgi.http.whiteboard.filter.regex"
```

Service property specifying the request mappings for a servlet `Filter` service.

The specified regular expressions are used to determine whether a request should be mapped to the servlet filter. The regular expressions must follow the syntax defined in `java.util.regex.Pattern`. Servlet filter services without this service property or the HTTP_WHITEBOARD_FILTER_SERVLET or the HTTP_WHITEBOARD_FILTER_PATTERN service property are ignored.

The value of this service property must be of type `String`, `String[]`, or `Collection<String>`.

See Also:
"java.util.regex.Pattern"

HTTP_WHITEBOARD_FILTER_ASYNC_SUPPORTED

```
public static final String HTTP_WHITEBOARD_FILTER_ASYNC_SUPPORTED =
"osgi.http.whiteboard.filter.asyncSupported"
```

Service property specifying whether a servlet `Filter` service supports asynchronous processing.

By default servlet filters services do not support asynchronous processing.

The value of this service property must be of type `Boolean`.

See Also:
"Java Servlet Specification Version 3.0, Section 2.3.3.3 Asynchronous Processing"

HTTP_WHITEBOARD_FILTER_DISPATCHER

```
public static final String HTTP_WHITEBOARD_FILTER_DISPATCHER =
"osgi.http.whiteboard.filter.dispatcher"
```

Service property specifying the dispatcher handling of a servlet `Filter`.

By default servlet filter services are associated with client requests only (see value DISPATCHER_REQUEST).

The value of this service property must be of type `String`, `String[]`, or `Collection<String>`. Allowed values are DISPATCHER_ASYNC, DISPATCHER_ERROR, DISPATCHER_FORWARD, DISPATCHER_INCLUDE, DISPATCHER_REQUEST.

See Also:
"Java Servlet Specification Version 3.0, Section 6.2.5 Filters and the RequestDispatcher"

HTTP_WHITEBOARD_FILTER_INIT_PARAM_PREFIX

```
public static final String HTTP_WHITEBOARD_FILTER_INIT_PARAM_PREFIX = "filter.init."
```

Service property prefix referencing a service.

For services this prefix can be used for service properties to mark them as initialization parameters which can be retrieved from the associated filter config. The prefix is removed from the service property name to build the initialization parameter name.

For services, the value of each initialization parameter service property must be of type `String`.

~~HTTP_WHITEBOARD_LISTENER~~

```
public static final String HTTP_WHITEBOARD_LISTENER = "osgi.http.whiteboard.listener"
```

Service property to mark a Listener service as a Whiteboard service. Listener services with this property set to the string value "true" will be treated as Whiteboard services opting in to being handled by the Http Whiteboard implementation. If the value "false" is specified, the service is opting out and this case is treated exactly the same as if this property is missing. If an invalid value is specified this is treated as a failure.

The value of this service property must be of type `String`. Valid values are "true" and "false" ignoring case.

~~DISPATCHER_REQUEST~~

```
public static final String DISPATCHER_REQUEST = "REQUEST"
```

Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied to client requests.

See Also:
"Java Servlet Specification Version 3.0, Section 6.2.5 Filters and the RequestDispatcher"

DISPATCHER_INCLUDE

```
public static final String DISPATCHER_INCLUDE = "INCLUDE"
```

Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied to include calls to the dispatcher.

See Also:
"Java Servlet Specification Version 3.0, Section 6.2.5 Filters and the RequestDispatcher"

DISPATCHER_FORWARD

```
public static final String DISPATCHER_FORWARD = "FORWARD"
```

Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied to forward calls to the dispatcher.

See Also:
"Java Servlet Specification Version 3.0, Section 6.2.5 Filters and the RequestDispatcher"

DISPATCHER_ASYNC

```
public static final String DISPATCHER_ASYNC = "ASYNC"
```

Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied in the asynchronous context.

See Also:
"Java Servlet Specification Version 3.0, Section 6.2.5 Filters and the RequestDispatcher"

DISPATCHER_ERROR

```
public static final String DISPATCHER_ERROR = "ERROR"
```

Possible value for the HTTP_WHITEBOARD_FILTER_DISPATCHER property indicating the servlet filter is applied when an error page is called.

See Also:
"Java Servlet Specification Version 3.0, Section 6.2.5 Filters and the RequestDispatcher"

HTTP_WHITEBOARD_RESOURCE_PATTERN

```
public static final String HTTP_WHITEBOARD_RESOURCE_PATTERN =
"osgi.http.whiteboard.resource.pattern"
```

Service property specifying the request mappings for resources.

The specified patterns are used to determine whether a request should be mapped to resources. Resource services without this service property are ignored.

The value of this service property must be of type `String`, `String[]`, or `Collection<String>`.

See Also:
"Java Servlet Specification Version 3.0, Section 12.2 Specification of Mappings",
HTTP_WHITEBOARD_RESOURCE_PREFIX

HTTP_WHITEBOARD_RESOURCE_PREFIX

```
public static final String HTTP_WHITEBOARD_RESOURCE_PREFIX =
"osgi.http.whiteboard.resource.prefix"
```

Service property specifying the resource entry prefix for a resource service.

If a resource service is registered with this property, requests are served with bundle resources.

This prefix is used to map a requested resource to the bundle's entries. The value must not end with slash ("/") with the exception that a name of the form "/" is used to denote the root of the bundle. See the specification text for details on how HTTP requests are mapped.

The value of this service property must be of type `String`.

See Also:
HTTP_WHITEBOARD_RESOURCE_PATTERN

HTTP_WHITEBOARD_TARGET

```
public static final String HTTP_WHITEBOARD_TARGET = "osgi.http.whiteboard.target"
```

Service property specifying the target filter to select the Http Whiteboard implementation to process the service.

An Http Whiteboard implementation can define any number of service properties which can be referenced by the target filter. The service properties should always include the `osgi.http.endpoint` service property if the endpoint information is known.

If this service property is not specified, then all Http Whiteboard implementations can process the service.

The value of this service property must be of type `String` and be a valid `filter string`.

8 Considered Alternatives

9 Security Considerations

~~*Description of all known vulnerabilities this may either introduce or address as well as scenarios of how the weaknesses could be circumvented.*~~

A `FilePermission` check may be required in order to verify a servlet has permissions to read and write to the directory specified by the `osgi.http.whiteboard.servlet.multipart.location` property. As this value defaults to being relative to the location of `javax.servlet.context.tempdir` (which is not specified in the http whiteboard) the check must be made even if the property is not set. This is due to the fact the servlet may not have permission to read or write to this default location.

10 Document Support

10.1 References

[1]. Bradner, S., Key words for use in RFCs to Indicate Requirement Levels, RFC2119, March 1997.

[2]. Software Requirements & Specifications. Michael Jackson. ISBN 0-201-87712-0

10.2 Author's Address

Name	Carsten Ziegeler
Company	Adobe Systems Incorporated

10.3 Acronyms and Abbreviations

10.4 End of Document