

Table of contents

1. [Application](#)²
2. [Client](#)³
3. [Component](#)⁴
4. [Connector](#)⁵
5. [Context](#)⁶
6. [Directory](#)⁷
7. [Engine](#)⁸
8. [Filter](#)⁹
9. [Finder](#)¹⁰
10. [Guard](#)¹¹
11. [Redirector](#)¹²
12. [Representation](#)¹³
13. [Request](#)¹⁴
14. [Resource](#)¹⁵
15. [Response](#)¹⁶
16. [Restlet](#)¹⁷
17. [Route](#)¹⁸
18. [Router](#)¹⁹
19. [Server](#)²⁰
20. [Transformer](#)²¹
21. [Uniform](#)²²
22. [Virtual Host](#)²³

Application

Restlet that can be attached to one or more VirtualHosts. Applications are guaranteed to receive calls with their base reference set relatively to the VirtualHost that served them. This class is both a descriptor able to create the root Restlet and the actual Restlet that can be attached to one or more VirtualHost instances.

Client

Connector acting as a generic client. It internally uses one of the available connectors registered with the current Restlet implementation.

Component

Restlet managing a set of Clients, Servers and other Restlets.

"A component is an abstract unit of software instructions and internal state that provides a transformation of data via its interface." Roy T. Fielding

Component managing a set of VirtualHosts and Applications. Applications are expected to be directly attached to VirtualHosts. Components are also exposing a number of services in order to control several operational features in a portable way, like access log and status setting.

Connector

Restlet enabling communication between Components.

"A connector is an abstract mechanism that mediates communication, coordination, or cooperation among components. Connectors enable communication between components by transferring data elements from one interface to another without changing the data." Roy T. Fielding

"Encapsulate the activities of accessing resources and transferring resource representations. The connectors present an abstract interface for component communication, enhancing simplicity by providing a clean separation of concerns and hiding the underlying implementation of resources and communication mechanisms." Roy T. Fielding

Context

Contextual data and services provided to a Restlet. The context is the means by which a Restlet may access the software environment within the framework. It is typically provided by the immediate parent Restlet (Component and Application are the most common cases). The services provided are access to a logger, access to configuration parameters and to a request dispatcher.

Directory

Handler mapping a directory of local resources. Those resources have representations accessed by the file system, the WAR context or the class loaders. An automatic content negotiation mechanism (similar to the one in Apache HTTP server) is used to select the best representation of a resource based on the available variants and on the client capabilities and preferences.

Engine

A Restlet Engine is an implementation of the Restlet API. The reference implementation, provided by Noelios Technologies, is therefore called the Noelios Restlet Engine (NRE).

Filter

Restlet filtering calls before passing them to an attached Restlet. The purpose is to do some pre-processing or post-processing on the calls going through it before or after they are actually handled by an attached Restlet. Also note that you can attach and detach targets while handling incoming calls as the filter is ensured to be thread-safe.

Finder

Restlet that can find the target resource that will concretely handle the request. Based on a given resource class, it is also able to instantiate the resource with the call's context, request and response without requiring the usage of a subclass. Once the target resource has been found, the call is automatically dispatched to the appropriate `handle*()` method (where the '*' character corresponds to the method name) if the corresponding `allow*()` method returns true.

For example, if you want to support a MOVE method for a WebDAV server, you just have to add a `handleMove()` method in your subclass of Resource and it will be automatically be used by the Finder instance at runtime.

If no matching `handle*()` method is found, then a `Status.CLIENT_ERROR_METHOD_NOT_ALLOWED` is returned.

Guard

Filter guarding the access to an attached Restlet.

Redirector

Rewrites URIs then redirects the call or the client to a new destination.

Representation

Current or intended state of a resource. For performance purpose, it is essential that a minimal overhead occurs upon initialization. The main overhead must only occur during invocation of content processing methods (`write`, `getStream`, `getChannel` and `toString`). Current or intended state of a resource.

"REST components perform actions on a resource by using a representation to capture the current or intended state of that resource and transferring that representation between components. A representation is a sequence of bytes, plus representation metadata to describe those bytes. Other commonly used but less precise names for a representation include: document, file, and HTTP message entity, instance, or variant."

Roy T. Fielding

Request

Generic request sent by client connectors. It is then received by server connectors and processed by Restlets. This request can also be processed by a chain of Restlets, on the client or server sides. Requests are uniform across all types of connectors, protocols and components.

Resource

Intended conceptual target of a hypertext reference. "Any information that can be named can be a resource: a document or image, a temporal service (e.g. "today's weather in Los Angeles"), a collection of other resources, a non-virtual object (e.g. a person), and so on. In other words, any concept that might be the target of an author's hypertext reference must fit within the definition of a resource."

"The only thing that is required to be static for a resource is the semantics of the mapping, since the semantics is what distinguishes one resource from another." Roy T. Fielding

Another definition adapted from the URI standard (RFC 3986): a resource is the conceptual mapping to a representation (also known as entity) or set of representations, not necessarily the representation which corresponds to that mapping at any particular instance in time. Thus, a resource can remain constant even when its content (the representations to which it currently corresponds) changes over time, provided that the conceptual mapping is not changed in the process. In addition, a resource is always identified by a URI.

Response

Generic response sent by server connectors. It is then received by client connectors. Responses are uniform across all types of connectors, protocols and components.

Restlet

Dispatcher that provides a context and life cycle support.

Route

Filter scoring the affinity of calls with the attached Restlet. The score is used by an associated Router in order to determine the most appropriate Restlet for a given call.

Router

Restlet routing calls to one of the attached routes. Each route can compute an affinity score for each call depending on various criteria. The `attach()` method allow the creation of routes based on URI patterns matching the beginning of a the resource reference's remaining part.

In addition, several routing modes are supported, implementing various algorithms:

- Best match (default)
- First match
- Last match
- Random match
- Round robin
- Custom

Note that for routes using URI patterns will update the resource reference's base reference during the routing if they are selected. If you are using hierarchical paths, remember to directly attach the child routers to their parent router instead of the top level Restlet component. Also, remember to manually handle the path separator characters in your path patterns otherwise the delegation will not work as expected.

Finally, you can modify the routes list while handling incoming calls as the delegation code is ensured to be thread-safe.

Server

Connector acting as a generic server. It internally uses one of the available connectors registered with the current Restlet implementation.

Transformer

Filter that can transform XML representations by applying an XSLT transform sheet.

Uniform

Base class exposing a uniform REST interface.

"The central feature that distinguishes the REST architectural style from other network-based styles is its emphasis on a uniform interface between components. By applying the software engineering principle of generality to the component interface, the overall system architecture is simplified and the visibility of interactions is improved. Implementations are decoupled from the services they provide, which encourages independent evolvability." Roy T. Fielding

It has many subclasses that focus on a specific ways to handle calls like filtering, routing or finding a target resource. The context property is typically provided by a parent component as a way to give access to features such as logging and client connectors.

Virtual Host

Router of calls from Server connectors to Restlets. The attached Restlets are typically Applications.