

# EiffelRSS

---

*FETCH Developer Guide*

Michael Käser <kaeserm@student.ethz.ch>

Martin Luder <luderm@student.ethz.ch>

Thomas Weibel <weibelt@student.ethz.ch>

### **Abstract**

FETCH is a class which has features that can fetch data from a source address to a local STRING using various services.

FETCH provides a simple interface for the DATA\_RESOURCE class in EiffelNet.

## Contents

<b>1</b>	<b>Overview</b>	<b>1</b>
<b>2</b>	<b>Usage</b>	<b>2</b>
<b>3</b>	<b>Features of FETCH</b>	<b>3</b>
3.1	Initialization . . . . .	3
3.1.1	make . . . . .	3
3.1.2	make_source . . . . .	3
3.2	Access . . . . .	3
3.2.1	data . . . . .	3
3.2.2	source_address . . . . .	3
3.2.3	error . . . . .	3
3.3	Basic Operations . . . . .	4
3.3.1	set_address . . . . .	4
3.3.2	fetch . . . . .	4

## List of Figures

1	BON diagram of cluster <code>FETCH</code> . . . . .	1
2	BON diagram of classes <code>FETCH</code> and <code>FETCH_HTTP_PROTOCOL</code> . . .	1

## 1 Overview

FETCH is a class which has features that can fetch data from a source address to a local STRING using various services.

FETCH provides a simple interface for the DATA\_RESOURCE class in EiffelNet.

A valid source address has the following format: `service://address`.

Supported services are: file, http, ftp.

See figure 1 for an overview of the cluster.

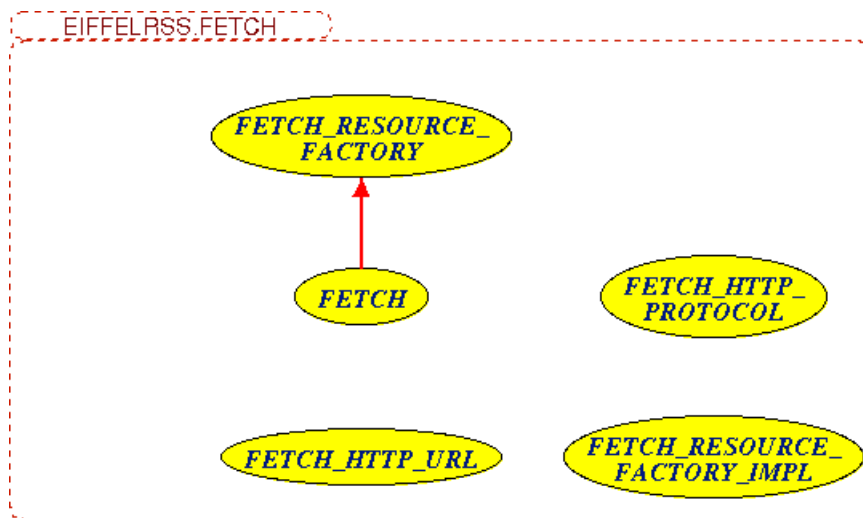


Figure 1: BON diagram of cluster FETCH

Figure 2 shows the classes `FETCH` and `FETCH_HTTP_PROTOCOL`.

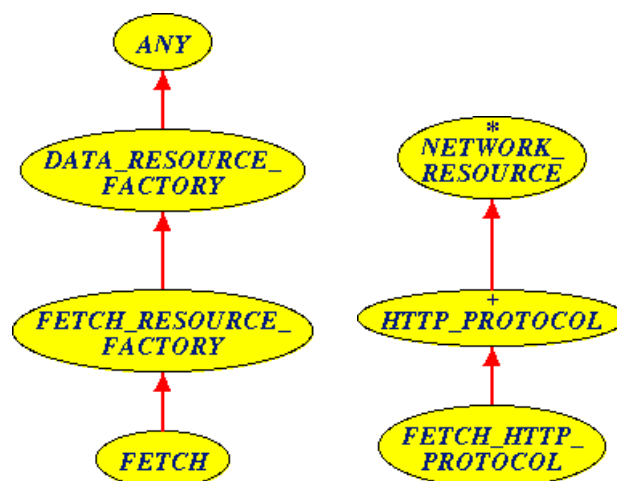


Figure 2: BON diagram of classes `FETCH` and `FETCH_HTTP_PROTOCOL`

## 2 Usage

```
class
  USAGE_EXAMPLE

create
  make

feature — Initialization

  make is
    — Creation procedure.
    local
      fetch: FETCH
      address: STRING
    do
      create fetch.make

      io.put_string ("Please enter an address to fetch: \
→%N")
      io.read_line

      address := io.last_string.twin

      fetch.set_address (address)
      fetch.fetch

      io.put_new_line

      if (fetch.error = fetch.Invalid_address) then
        io.put_string ("Error: Invalid address")
      elseif (fetch.error = fetch.Transfer_failed) then
        io.put_string ("Error: Transfer failed")
      else
        io.put_string (fetch.data)
      end

      io.put_new_line
    end

end — class USAGE_EXAMPLE
```

## 3 Features of FETCH

### 3.1 Initialization

#### 3.1.1 make

```
make
— Create the object without a source address
```

#### 3.1.2 make\_source

```
make_source (an_address: STRING) is
— Create the object with a predefined source
```

### 3.2 Access

#### 3.2.1 data

```
data: STRING
— The data fetched
```

This can be void if there was an error.

#### 3.2.2 source\_address

```
source_address: STRING
— The source address
```

This can be void.

#### 3.2.3 error

```
error: INTEGER
— An error number
```

error can be one of the following constants: `None`, `Invalid_address`, `Transfer_failed`.

`None` means that there was no error and data is available. `Invalid_address` means that the given source address was either empty or not valid. If the error is `Transfer_failed`, there was a problem when trying to load the data, i.e. there was no connection to the internet.

### 3.3 Basic Operations

#### 3.3.1 set\_address

```
set_address (an_address: STRING)  
— Sets the address to an_address
```

This sets the source address to `an_address`. After calling this feature, error can be `Invalid_address`.

#### 3.3.2 fetch

```
fetch  
— Try to fetch the data from source_address
```

If error is not `Invalid_address`, `fetch` tries to open it and load the data. If there is a problem while loading the data, error is set to `Transfer_failed`.