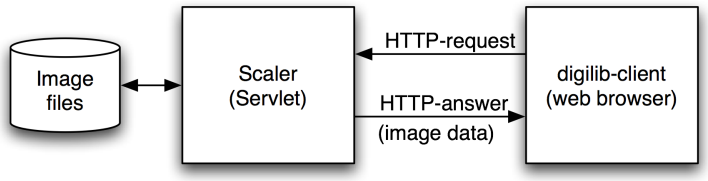
A short introduction to digilib

digilib -- how does it work?

The image server digilib is a state-less web-based client-server application for interactive viewing and manipulation of images.

Frontend and Scaler server

digilib consists mainly of two parts, the image server component proper, called “Scaler” and a client-side part that runs in the users web browser:



1: digilib client and Scaler Servlet (simple)

The users browser sends an HTTP request for a certain (zoomed, scaled, rotated) image to the Scaler server and the server returns the image data as HTTP response.

To complete the schematics of figure 1 we must also take into account that the client-side part consisting of HTML and Javascript code has also been requested and loaded from a frontend-web server into the users browser:

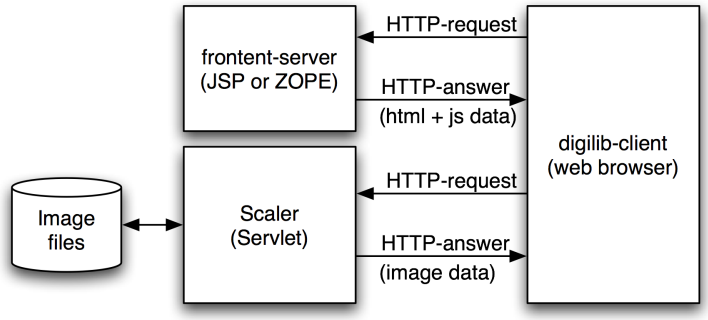


Figure 2: digilib client with frontend and Scaler server

To date there are several frontend implementations for digilib like the “classic” (grey buttons, implemented in JSP[1]) version that comes with the default digilib distribution or the “Zogilib” frontend version in ZOPE[2]. The Frontend-server and the Scaler-server do not have to run on the same machine and often there are several frontends that use the same Scaler server.

Request formats

In theory the formats of the HTTP requests for the frontend and the Scaler server can be completely different. The HTML and Javscript code of the frontend just has to generate the correct request for the Scaler image. Most of the current frontend implementations share some or all of the parameters with the Scaler API.

1 See public CVS http://cvs.berlios.de/cgi-bin/viewcvs.cgi/digilib/client/digitallibrary/
2 See public CVS http://itgroup.mpiwg-berlin.mpg.de/cgi-bin/cvsweb.cgi/zogiLib/

Scaler

The Scaler API is documented on the digilib.berlios.de pages[3]. Here is the current version (as of September 2006):

The Scaler servlet takes parameters in the http request format: `Scaler/request_path/?param1=value1&param2=value2&...` Unknown parameters will be silently ignored.

Recognised parameters:

- **`request_path`** path to file or directory.
- **`fn`** path to file or directory below `/docuserver/images` (or other configured image directory). This path will be added to the `request_path` behind the servlet name. Either parameter can be empty.
- **`pn`** page number. Index into the (alphabetically sorted) directory given by the path. Starts with 1. Ignored if the path points to a file. Default: 1.
- **`dw`** destination image width (pixels). If omitted the image is scaled to fit `dh`.
- **`dh`** destination image height (pixels). If omitted the image is scaled to fit `dw`.
- **`wx`** relative x offset of the image area to be sent (0 <= `wx` <= 1). Default: 0.
- **`wy`** relative y offset of the image area to be sent (0 <= `wy` <= 1). Default: 0.
- **`ww`** relative width of the image area to be sent (0 <= `ww` <= 1). Default: 1.
- **`wh`** relative height of the image area to be sent (0 <= `wh` <= 1). Default: 1.
- **`ws`** additional scaling factor. The resulting image will have the size [`ws*dw,ws*dh`]. Default: 1.
- **`mo`** flags for the mode of operation separated by ",".
  - **`fit`**: always scale the image to fit [`dw,dh`] (default).
  - **`clip`**: send the file in its original resolution, cropped to fit [`dw,dh`].
  - **`osize`**: scale to original size based on the image resolution (from the image metadata) and display resolution (from parameter ddpi). Fails if either resolution is unknown.
  - **`file`**: send the file as-is (may be very large and all sorts of image types!). If configuration doesn't allow sending files (sendfile-allowed=false) revert to clip.
  - **`rawfile`**: send the file as-is with a mime-type of application/octet-stream so it can be downloaded with the browser.
  - **`errtxt`**: send error response as HTML.
  - **`errimg`**: response as image (default).
  - **`q0-q2`**: quality of interpolation in scaling (q0: worst, default).
  - **`lores`**: try to use scaled image (default)
  - **`hires`**: always use unscaled image.
  - If the image is zoomed (`ww`, `wh` < 1) the use of the scaled image files depends on the requested resolution.
  - **`vmir`**: mirror image vertically.
  - **`hmir`**: mirror image horizontally.

3 http://developer.berlios.de/docman/display_doc.php?docid=106&group_id=251

    - **jpg**: the resulting image is always sent as JPEG (otherwise TIFF and PNG images are sent as PNG).
- **cont**: change contrast of the image. Negative values reduce contrast, positive values enhance contrast. Pixel value is multiplied by 2^cont. Default: 0
- **brgt**: change brightness of image. Negative value reduces brightness, positive value enhances brightness. The value brgt is added to the pixel value. Default: 0
- **rot**: rotate image. Rotation angle is given in degrees. Default: 0
- **rgbm**: modify colour by multiplication. The contrast of the red green and blue components of the image can be reduced or enhanced similar to the cont parameter. The factors for red, green and blue are separated by slashes (for example 0.86/0/-0.5). Default: 0/0/0
- **rgba**: modify colour by addition. The brightness of the red green and blue components of the image can be reduced or enhanced similar to the brgt parameter. The factors for red, green and blue are separated by slashes (for example 100/0/25). Default: 0/0/0
- **ddpi**: resolution of client display for osize mode. Either ddpi or ddpix and ddpiy must be set to use osize mode.
- **ddpix**: resolution of client display in x direction for osize mode.
- **ddpiy**: resolution of client display in y direction for osize mode.
- The image to be loaded can be specified by the request_path (deprecated) or the fn (preferred) parameter and the optional index pn
    - if fn points to a directory then the file with the index pn (in alphabetical order according to ASCII) will be loaded
    - if fn points to a file (with or without extension) then this file will be loaded

The image will be scaled equally in horizontal and vertical direction such that the resulting image does not exceed the rectangle [dw,dh]. If only either height or width is given the image is scaled to match only the given parameter. The size of the resulting image in the other parameter is determined by the aspect ratio of the image.

An example for a Scaler URL is: http://nausikaa2.mpiwg-berlin.mpg.de/digitallibrary/servlet/Scaler?fn=experimental/rombilder&wh=0.1712&ww=0.1282&wy=0.1681&wx=0.6895&dw=862&dh=904 such a URL would be used as src attribute to an img element in the frontend HTML.