Internet Draft
draft-masinter-form-data-00.txt
Expires in 6 months

Larry Masinter
Xerox Corporation

March 18, 1997

Returning Values from Forms: multipart/form-data

Status of this Memo

1. Abstract

This specification defines an Internet Media Type, multipart/form-data, which can be used by a wide variety of applications and transported by a wide variety of protocols as a way of returning a set of values as the result of a user filling out a form. Typical applications include form values generated by HTML forms and submitted by HTTP post or by electronic mail, but the format is independent of those contexts. The definition of multipart/form-data is derived from its original definition in RFC 1867.

2. Definition of multipart/form-data

The media-type multipart/form-data follows the rules of all multipart MIME data streams as outlined in RFC 1521. It is intended for use in returning the data that comes about from filling out a form. In a form (in HTML, although other applications may also use forms), there are a series of fields to be supplied by the user who fills out the form. Each field has a name. Within a given form, the names are unique.

multipart/form-data contains a series of parts. Each part is expected to contain a content-disposition header where the value is "form-data" and a name attribute specifies the field name within the form, e.g., 'content-disposition: form-data; name="xxxxx"', where xxxxx is the field name corresponding to that field. Field names originally in non-ASCII character sets may be encoded using the method outlined in RFC 1522.

As with all multipart MIME types, each part has an optional Content-Type which defaults to text/plain. If the contents of a file are returned via filling out a form, then the file input is identified as application/octet-stream or the appropriate media type, if known. If multiple files are to be returned as the result of a single form entry, they can be returned as multipart/mixed embedded within the multipart/form-data.

Each part may be encoded and the "content-transfer-encoding" header supplied if the value of that part does not conform to the default encoding.

Forms may request file inputs from the user. Those file inputs may also identify the file name. The file name may be described using the 'filename' parameter of the "content-disposition" header. This is not required, but is strongly recommended in any case where the original filename is known.

3 Use of multipart/form-data

As with other multipart types, a boundary is selected that does not occur in any of the data. (This selection is sometimes done probabilisticly.) Each field of the form is sent, in the order defined by the form, as a part of the multipart stream. Each part identifies the INPUT name within the original form. Each part

should be labelled with an appropriate content-type if the media type is known (e.g., inferred from the file extension or operating system typing information) or as application/octet-stream.

If the value of a form field is a set of files rather than a single file, that value can be transferred together using the multipart/mixed format.

While the HTTP protocol can transport arbitrary BINARY data, the default for mail transport (e.g., if the ACTION is a "mailto:" URL) is the 7BIT encoding. The value supplied for a part may need to be encoded and the "content-transfer-encoding" header supplied if the value does not conform to the default encoding. [See section 5 of RFC 1521 for more details.]

The original local file name may be supplied as well, either as a 'filename' parameter either of the 'content-disposition: form-data' header or in the case of multiple files in a 'content-disposition: file' header of the subpart. The client application should make best effort to supply the file name; if the file name of the client's operating system is not in US-ASCII, the file name might be approximated or encoded using the method of RFC 1522. This is a convenience for those cases where, for example, the uploaded files might contain references to each other, e.g., a TeX file and its .sty auxiliary style description.

On the server end, the ACTION might point to a HTTP URL that implements the forms action via CGI. In such a case, the CGI program would note that the content-type is multipart/form-data, parse the various fields (checking for validity, writing the file data to local files for subsequent processing, etc.).

4. Operability considerations

4.1 Compression, encryption

Some of the data in forms may be compressed or encrypted, using other MIME mechanisms. This is a function of the application that is generating the form-data.

4.2 Other data encodings rather than multipart

Various people have suggested using new mime top-level type "aggregate", e.g., aggregate/mixed or a content-transfer-encoding of "packet" to express indeterminate-length binary data, rather than relying on the multipart-style boundaries. While we are not opposed to doing so, this would require additional design and standardization work to get acceptance of "aggregate". On the other hand, the 'multipart' mechanisms are well established, simple to implement on both the sending client and receiving server, and as efficient as other methods of dealing with multiple combinations of binary data.

4.3 Remote files with third-party transfer

In some scenarios, the user operating the client software might want to specify a URL for remote data rather than a local file. In this case, is there a way to allow the browser to send to the client a pointer to the external data rather than the entire contents? This capability could be implemented, for example, by having the client send to the server data of type "message/external-body" with "access-type" set to, say, "uri", and the URL of the remote data in the body of the message.

4.4 Non-ASCII field names

Note that MIME headers are generally required to consist only of 7-bit data in the US-ASCII character set. Hence field names should be encoded according to the prescriptions of RFC 1522 if they contain characters outside of that set.

5. Security Considerations

The data format described in this document introduces no new security considerations outside of those introduced by the protocols that use it and of the component elements. It is important when interpreting content-disposition to not overwrite files in the recipients address space inadvertently.

6. Author's Addresses

Larry Masinter

Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304

Phone: (415) 812-4365
Fax: (415) 812-4333
EMail: masinter@parc.xerox.com

A. Media type registration for multipart/form-data

Media Type name:
multipart

Media subtype name:
form-data

Required parameters:
none

Optional parameters:
none

Encoding considerations:
No additional considerations other than as for other multipart types.

Published specification:
RFC 1867

Security Considerations

The multipart/form-data type introduces no new security considerations beyond what might occur with any of the enclosed parts.

References

[RFC 1521] MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies. N. Borenstein & N. Freed. September 1993.

[RFC 1522] MIME (Multipurpose Internet Mail Extensions) Part Two: Message Header Extensions for Non-ASCII Text. K. Moore. September 1993.

[RFC 1806] Communicating Presentation Information in Internet Messages: The Content-Disposition Header. R. Troost & S. Dorner, June 1995.

[RFC 1867] Form-based File UPload in HTML. E. Nebel, L. Masinter, November 1995.