

Handling files

Thierry Sans

Browser restrictions

- It is **impossible** to write a piece of code that reads an arbitrary file in (server-side) Javascript
- ➔ Only files selected by users through file input forms can be processed

```
<form . . . >  
    <input type="file" name="img" multiple>  
    <input type="submit">  
</form>
```

[optional] select
multiple files

Sending a file from the terminal

```
$ curl -X POST  
  -H "Content-Type: multipart/form-data"  
  -F "picture=@localpath/to/img.png"  
  -F "username=bart"  
http://...
```

Sending a file from the browser

- **Form action** (with page refresh)

```
<form action="/url"  
      method="POST"  
      enctype="multipart/form-data">
```

- **Ajax request** (without page refresh)

```
var file = document.get ...  
var formdata = new FormData();  
formdata.append("picture", file);  
xhr.send(formdata);
```

What is received on the server

File metadata

- filename
- mimetype (file type)
- size
- and others

File content

- Compressed binary or string

MIME types

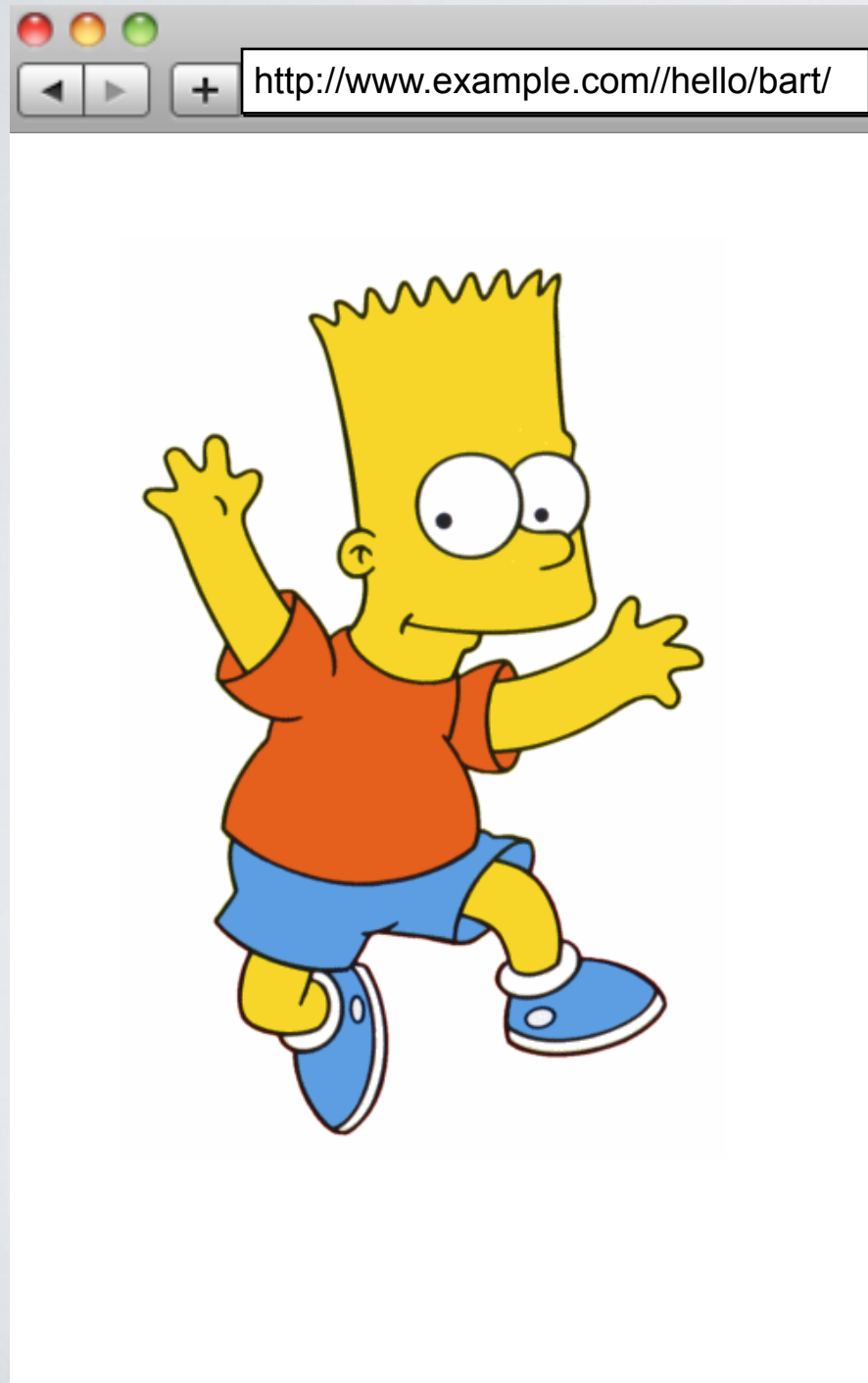
MIME (Multipurpose Internet Mail Extensions)
is also known as the **content type**

- ➔ Define the format of a document exchanged on internet (IETF standard) <http://www.iana.org/assignments/media-types/index.html>

Examples of MIME types

- text/html
- text/css
- text/javascript
- image/jpeg - image/gif - image/svg - image/png (and so on)
- application/pdf
- application/json

Example of how images are retrieved



GET hello/bart/



```
<html>
  <body>
    <img src=images/bart.jpg/>
  </body>
</html>
```

MIME : text/html



GET images/bart.jpg



MIME : image/jpg

Do/Don't with files

- Do **not** send a base64 encoded file content with JSON, use `multipart/form-data` instead (compression)
- Do **not** store uploaded files with the static content
- Do **not** serve uploaded files statically (security)
- Do store the mimetype and set the HTTP response header mimetype when files are sent back