Bookmarklet Builder for Offline Data Retrieval

By
Sheetal Naidu
Advisor: Dr. Chris Pollett

Agenda

- Introduction
- Design
- Technologies Used
- Implementation
- Performance Tests
- Observations
- Conclusions

Introduction

- Bookmarklet Builder for Offline Data Retrieval is a system that lets you create a bookmarklet cache of a website which can then be viewed offline.
- A Bookmarklet is a Javascript program wrapped around a string of HTML code performing some action once it is loaded in a browser.
- To begin today we will look at the idea behind Bookmarklet Builder.

Bookmarklet Builder

- Bookmarklet Builder creates a bookmarklet which is a data:URI of a website or a set of web pages
- What is data:URI? - A data URI is a URL scheme which provides a way of including small data objects as immediate data in a web page rather than specifying the object as an external resource
- Its general syntax is

data:[<mediatype>][;base64],<data>

- URI - Uniform Resource Identifier (URI) is a compact string of characters for identifying an abstract or physical resource.
- URL - URL is a URI scheme which identifies a resource mainly by the way it is accessed. That is, its network “location”.

Prevalence of data:URI

- Existing Uses of data:URI
  - Data: URI of Images are included in HTML or XML pages instead of linking to their external resources
  - Mainly to reduce the number of HTTP requests thus making the page/s load faster
- Existing data:URI conversions
  - Online tools that convert text, images and at most, single pages to data: URI
- Existing Support for data:URI
  - Most browsers including IE version 8 onwards

Design

- Modules
  - UI
  - Crawler
  - PHP program
- Output is a data:URI

Technologies Used

- Javascript
  - An object-oriented scripting language which we mainly used to provide client-side functionality
- PHP
  - A general purpose scripting language originally designed for web development and interpreted by web browsers
- Nutch
- Document Object Model (DOM)

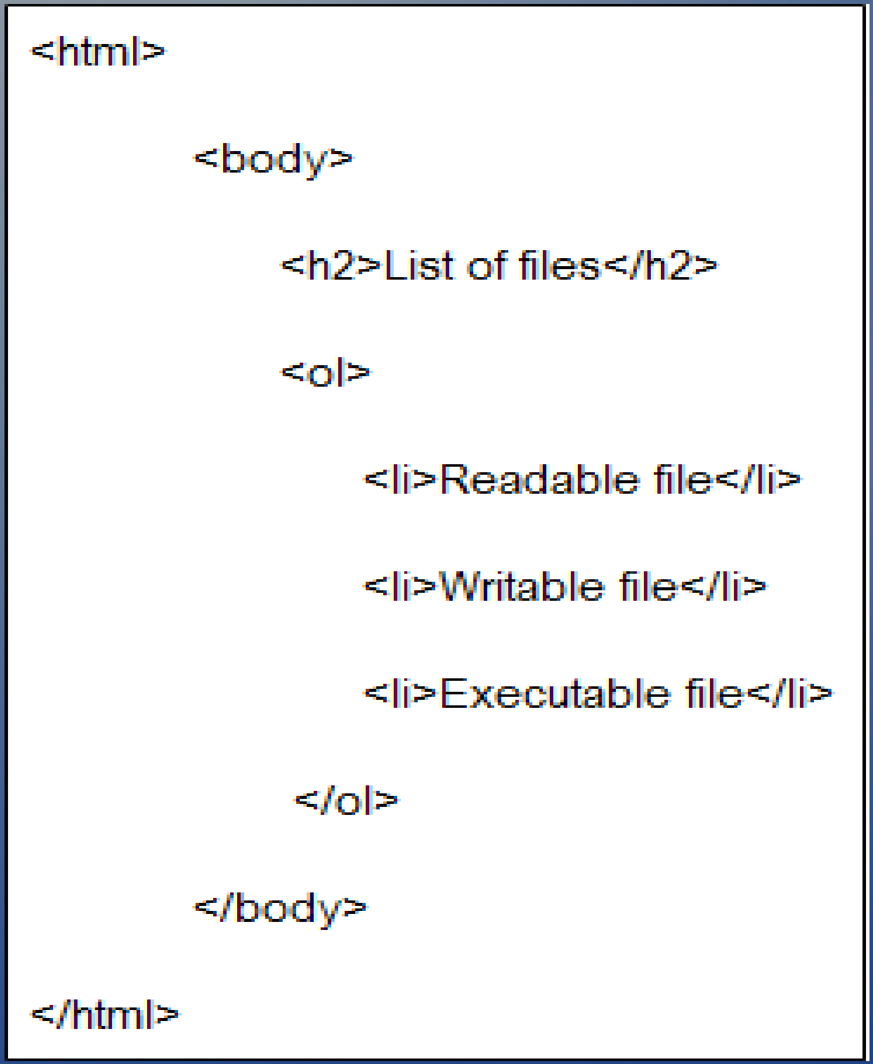
Crawler

- Nutch
  - Nutch is an open source Java search engine
  - We used only the crawling functionality provided by Nutch
- Open source, hence free
- Easy to install and use. And good documentation is available
- Input to the crawler is a URL and Depth
- Crawls the site and generates output of a list of pages
- This list is used for further processing

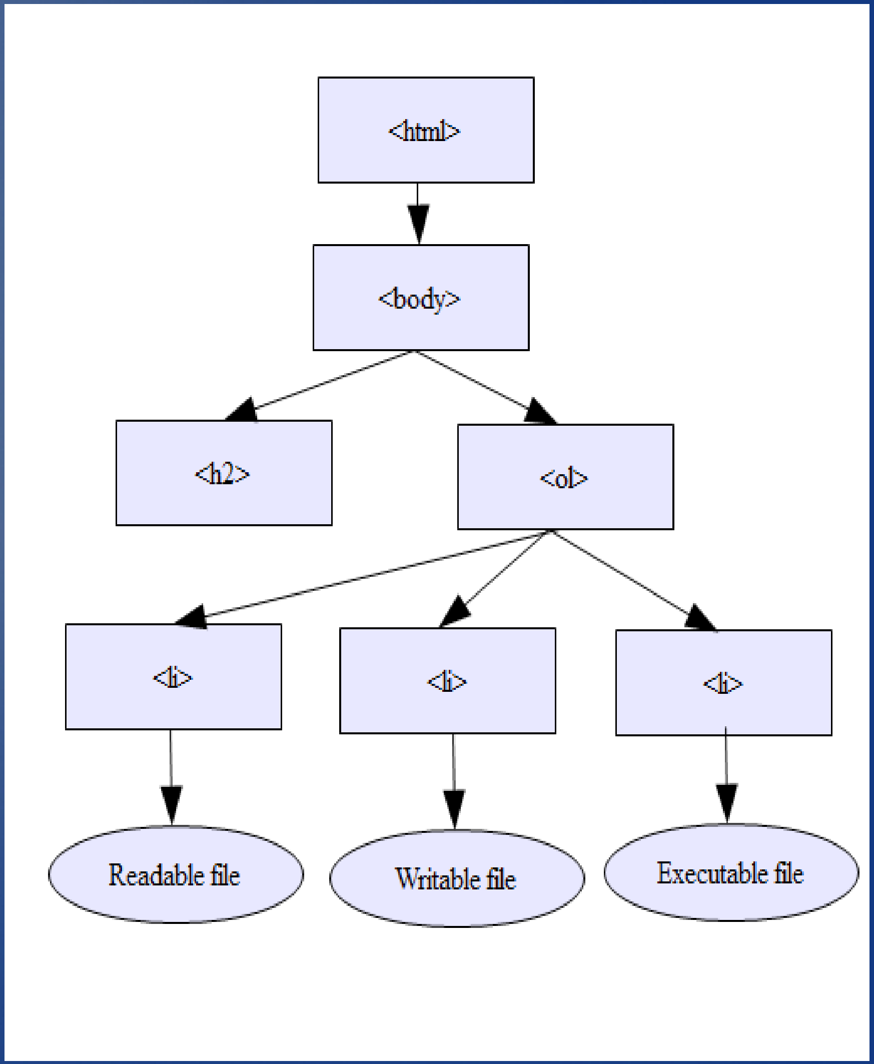
DOM

- DOM provides a language independent platform to access the properties and elements of a web page.
- It is an Application Programming Interface to represent and manipulate the content of HTML and XML documents.
- Example of a DOM structure

Figures of sample code and its corresponding DOM structure

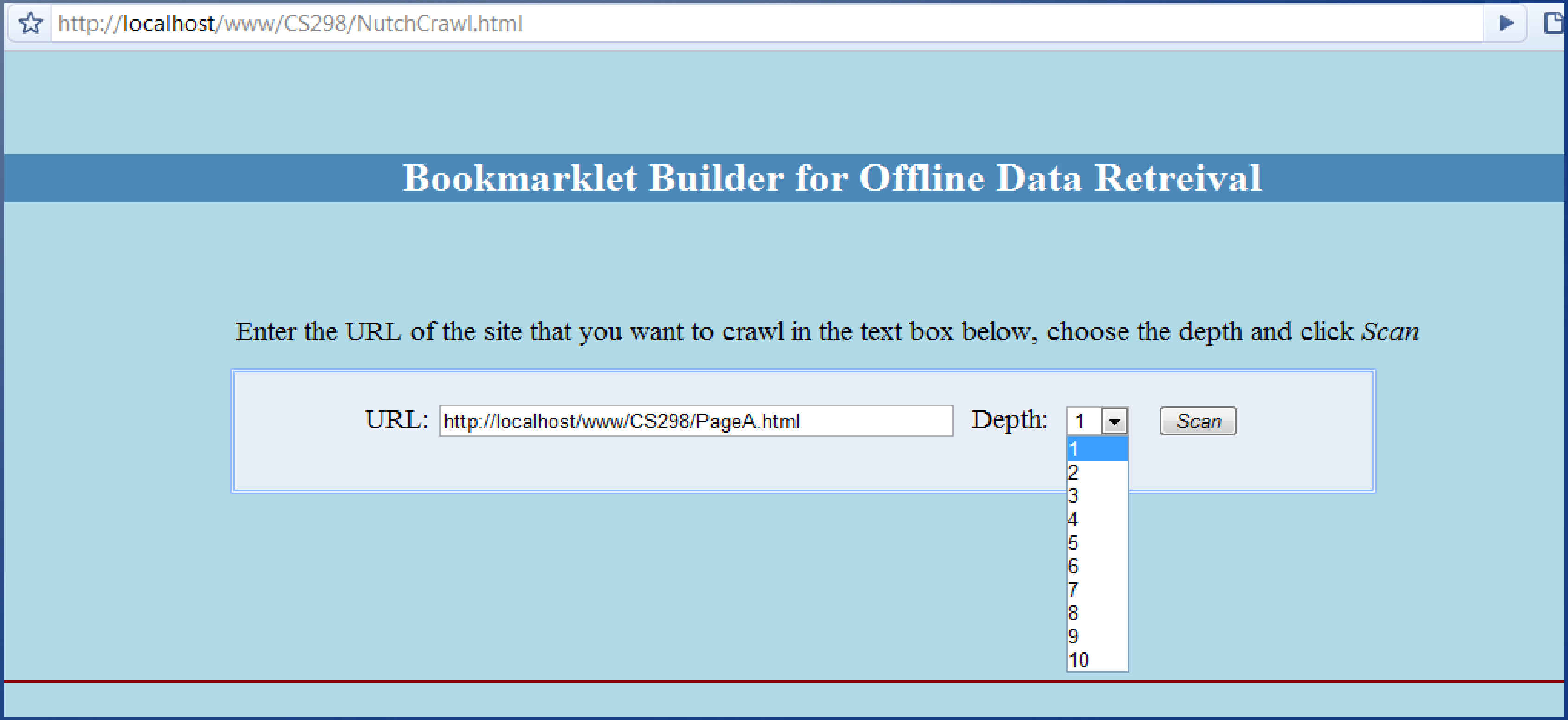


```
<html>
        <body>
            <h2>List of files</h2>
            <ol>
                <li>Readable file</li>
                <li>Writable file</li>
                <li>Executable file</li>
             </ol>
        </body>
</html>
```



Implementation – Web UI

- Web based design
- Input to the system
  - URL of a website
  - Depth



http://localhost/www/CS298/NutchCrawl.html

Bookmarklet Builder for Offline Data Retreival

Enter the URL of the site that you want to crawl in the text box below, choose the depth and click *Scan*

URL: http://localhost/www/CS298/PageA.html Depth: 1

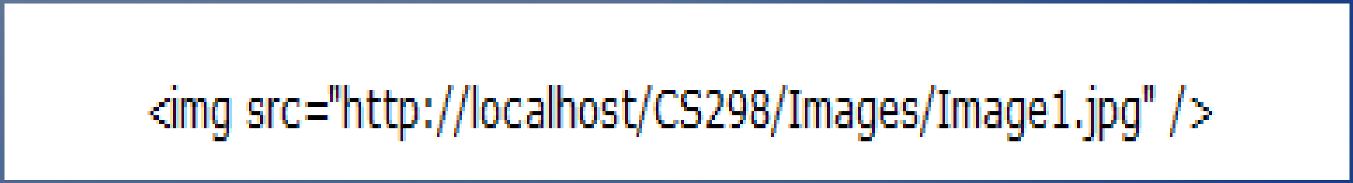
Implementation - Nutch

- Crawl command
  - bin/nutch crawl url_file -dir crawl_data -depth 1 -topN 10
- Readdb command
  - bin/nutch readdb crawl_data/crawldb -dump output_dir
- Sample output of readdb
  -

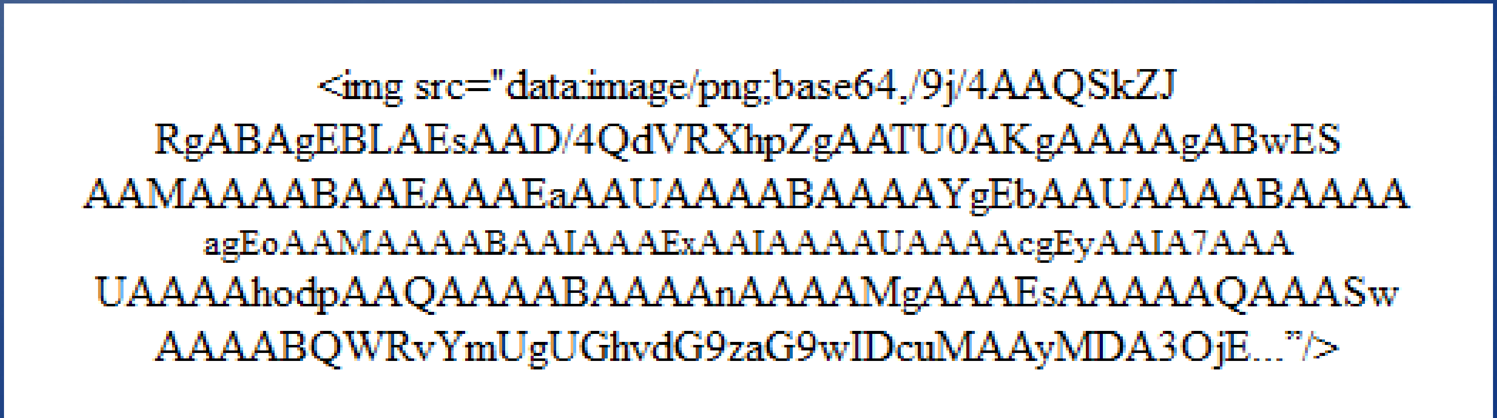
```
http://localhost/CS297/PageA.html Version: 4
Status: 2 (DB_fetched)
Fetch time: Fri Dec 07 16:28:34 PST 2007
Modified time: Wed Dec 31 16:00:00 PST 1969
Retries since fetch: 0
Retry interval: 30.0 days
Score: 1.6666667
Signature: e48ea88ce7aaa83d3115c598205ea05e
Metadata: null
```

Implementation – PHP Program

- Fetch each page – Contents of a page are stored as a string of data
- Converting Images



```
<img src="http://localhost/CS298/Images/Image1.jpg" />
```



```
<img src="data:image/png;base64,/9j/4AAQSkZJ
RgABAgEBLAEsAAD/4QdVRXhpZgAATU0AKgAAAAgABwES
AAMAAAABAAEAAAEaAAUAAAABAAAAYgEbAAUAAAABAAAA
agEoAAMAAAABAAIAAAExAAIAAAAUAAAAcgEyAAIA7AAA
UAAAAhodpAAQAAAABAAAAnAAAAMgAAAEsAAAAAQAAASw
AAAABQWRvYmUgUGhvdG9zaG9wIDcuMAAyMDA3OjE..."/>
```

Implementation – PHP Program cont'd.

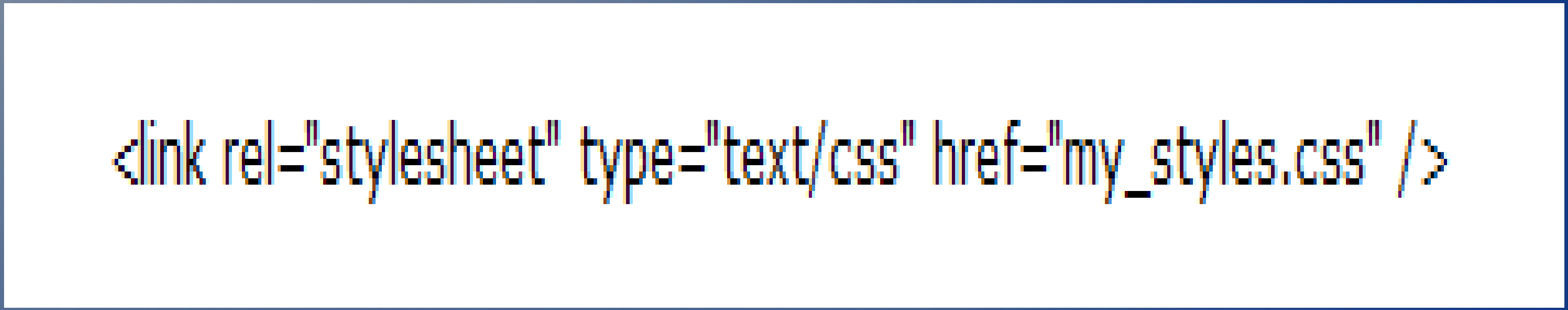
- Converting Links

```
<a href=”http://www.yahoo.com”>
```

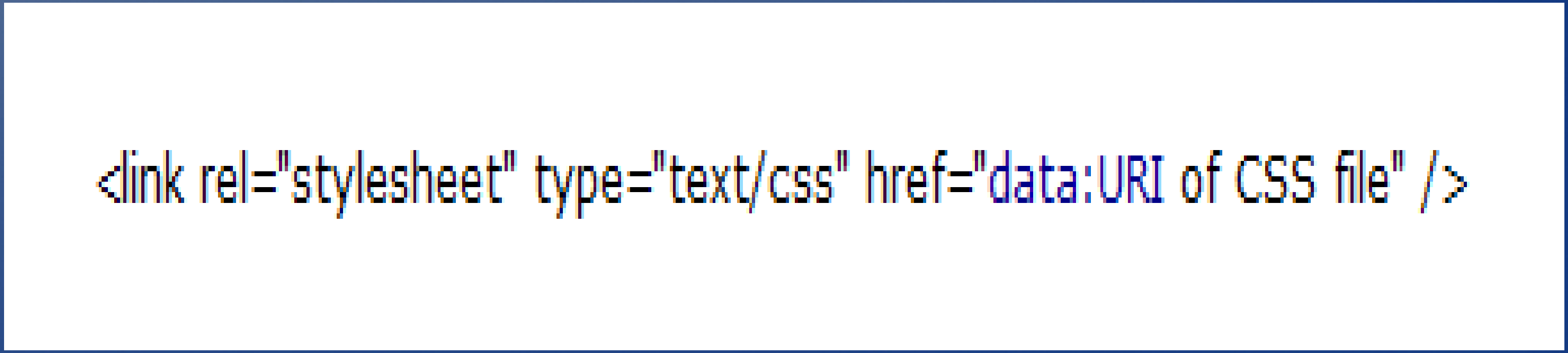
```
<a href=”javascript:parent.change_object_content('url_of_page)”>
```

Implementation – PHP Program cont'd.

- Converting CSS files



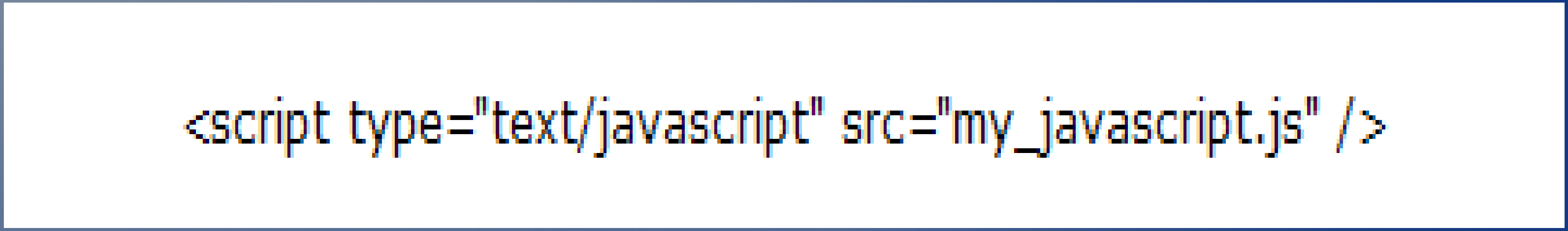
```
<link rel="stylesheet" type="text/css" href="my_styles.css" />
```



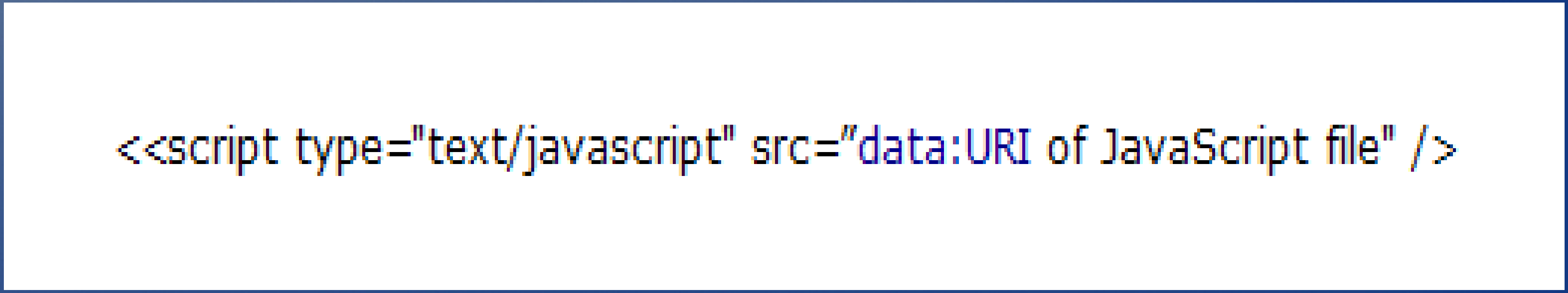
```
<link rel="stylesheet" type="text/css" href="data:URI of CSS file" />
```

Implementation – PHP Program cont'd.

- Converting Javascript files



```
<script type="text/javascript" src="my_javascript.js" />
```



```
<<script type="text/javascript" src="data:URI of JavaScript file" />
```

Performance Tests

- Different types of inputs were supplied to the system
  - Text only pages
    - Average size – 35 KB
  - Pages with Images
    - Average size - 290KB
  - Site with Varying Depth

Test Results for Average Web Page

No. of Pages	Time to Crawl	Time to Convert to URI	Total Time
5	48	7	55
6	52	27	79
8	51	22	73
10	48	40	88
12	50	85	135
14	48	61	109

- All times are in seconds; Depth = 2
- The above observations were made in Firefox
- The last row has a smaller “Time to Convert to URI” value where as the no. of pages has increased. This is because the pages added were 30% smaller in size than the other pages.

Results for Text-only pages

No. of Pages	Time to Crawl	Time to convert to URI	Total Time
4	46	0.1	46.1
6	49	0.2	49.2
8	49	0.3	49.3
10	50	0.9	50.9

- All times are in seconds; Depth =2
- These observations were made in Firefox

Performance Tests with Varying Depth

Depth	No. of Pages	Time to Crawl	Time to Convert to URI	Total Time
2	5	48	7	55
3	5	59	15	74
4	6	69	16	85
5	7	82	22	104
6	8	89	33	122
7	9	105	35	140

- All times are in seconds and these observations were made in Firefox.

data:URI sizes

No. of Pages	URI Length (no. of characters)
5	1491318
8	3561366
10	4921554
13	6961830
15	8322798

- These results were observed in Firefox and Opera web browsers*

Observations

- Recursive conversion to data: URI
  - Our system converts data into the data: URI form three times and browsers are able to display the information properly
- More testing is necessary to find if there is a maximum number for such recursive conversion
- Length of data:URI - the maximum length seen in our tests was 8322338 characters in Firefox and Opera

Observations cont'd.

- Firefox displays URI lengths of up to 4921554
- Opera displays URI lengths of greater than 5601824 characters
- For at least up to 8322338 characters, the content is displayed properly even if the URI itself is not displayed in the browser
- Firefox and Chrome behave differently from Opera in the way the Back button works

Conclusions

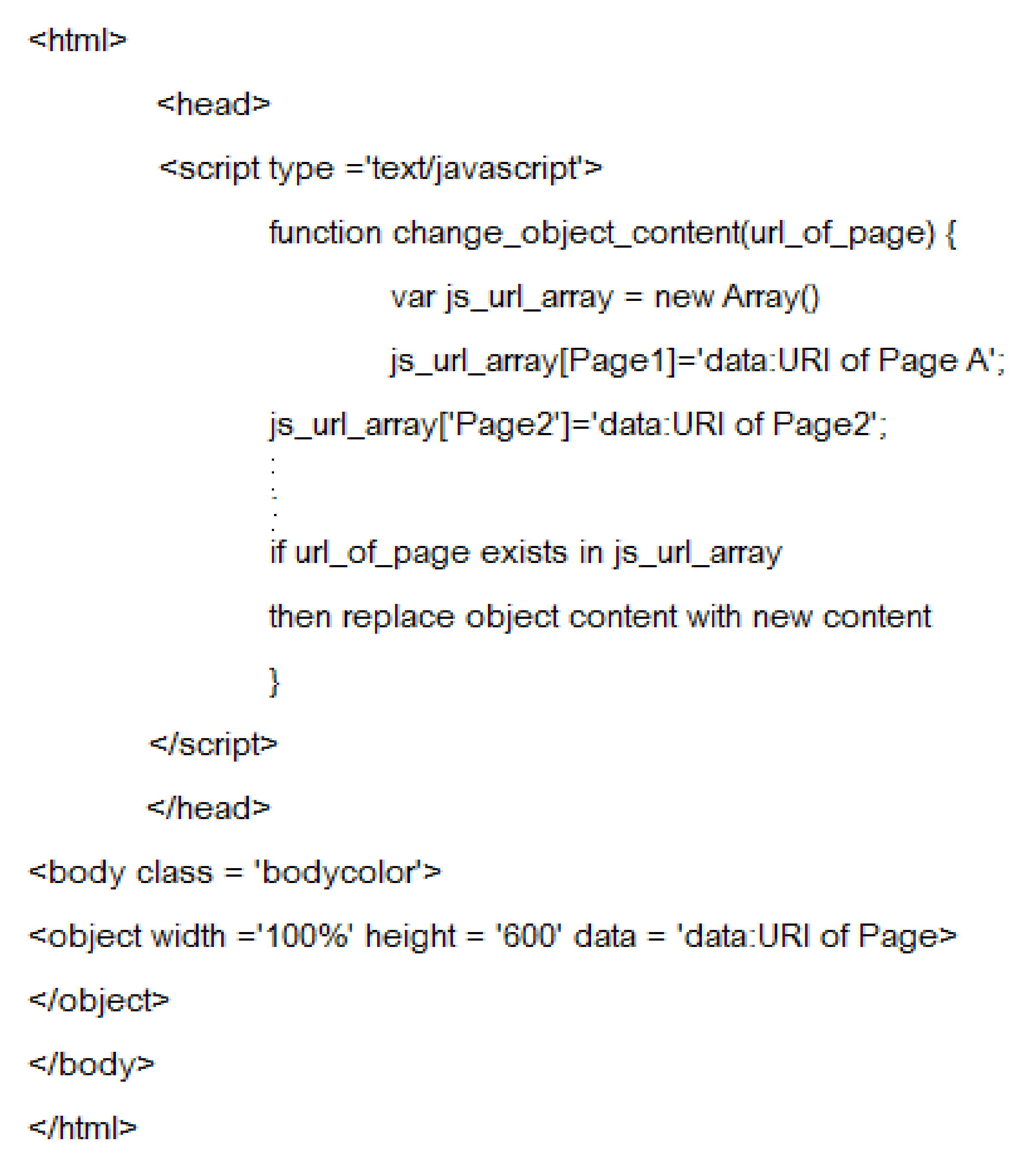
- A neat way to convert entire websites into a single long string of data
- All you need is a browser
- Can browse complete websites when offline
- Larger in size than actual file size of all pages but more straight forward than caching individual pages
- Will not consume cache memory and it is just like saving any other file
- Using compression techniques will be beneficial

Conclusions cont'd.

- Speeding up function/s to fetch images will be an enhancement
- Re-using already fetched web pages, image files, CSS and Javascript files will also enhance the system
- Suitable for pages with small data items

Thank You

Q & A



```
<html>
        <head>
        <script type ='text/javascript'>
                function change_object_content(url_of_page) {
                        var js_url_array = new Array()
                        js_url_array[Page1]='data:URI of Page A';
                js_url_array['Page2']='data:URI of Page2';
                :
                :
                .
                if url_of_page exists in js_url_array
                then replace object content with new content
                }
        </script>
        </head>
<body class = 'bodycolor'>
<object width ='100%' height = '600' data = 'data:URI of Page>
</object>
</body>
</html>
```