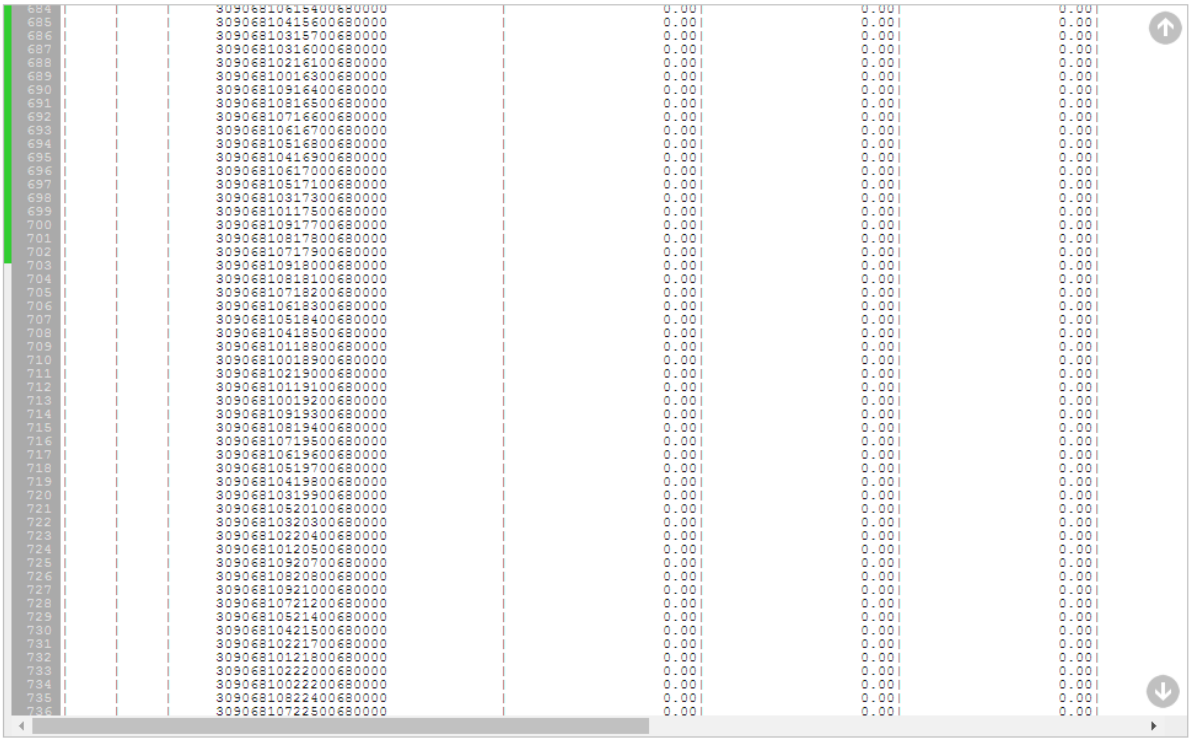
Lazy Text View widget



```
684 |  |  |  30906810615400680000  |      0.00|      0.00|      0.00|
685 |  |  |  30906810415600680000  |      0.00|      0.00|      0.00|
686 |  |  |  30906810315700680000  |      0.00|      0.00|      0.00|
687 |  |  |  30906810316000680000  |      0.00|      0.00|      0.00|
688 |  |  |  30906810216100680000  |      0.00|      0.00|      0.00|
689 |  |  |  30906810016300680000  |      0.00|      0.00|      0.00|
690 |  |  |  30906810916400680000  |      0.00|      0.00|      0.00|
691 |  |  |  30906810816500680000  |      0.00|      0.00|      0.00|
692 |  |  |  30906810716600680000  |      0.00|      0.00|      0.00|
693 |  |  |  30906810616700680000  |      0.00|      0.00|      0.00|
694 |  |  |  30906810516800680000  |      0.00|      0.00|      0.00|
695 |  |  |  30906810416900680000  |      0.00|      0.00|      0.00|
696 |  |  |  30906810617000680000  |      0.00|      0.00|      0.00|
697 |  |  |  30906810517100680000  |      0.00|      0.00|      0.00|
698 |  |  |  30906810317300680000  |      0.00|      0.00|      0.00|
699 |  |  |  30906810117500680000  |      0.00|      0.00|      0.00|
700 |  |  |  30906810917700680000  |      0.00|      0.00|      0.00|
701 |  |  |  30906810817800680000  |      0.00|      0.00|      0.00|
702 |  |  |  30906810717900680000  |      0.00|      0.00|      0.00|
703 |  |  |  30906810918000680000  |      0.00|      0.00|      0.00|
704 |  |  |  30906810818100680000  |      0.00|      0.00|      0.00|
705 |  |  |  30906810718200680000  |      0.00|      0.00|      0.00|
706 |  |  |  30906810618300680000  |      0.00|      0.00|      0.00|
707 |  |  |  30906810518400680000  |      0.00|      0.00|      0.00|
708 |  |  |  30906810418500680000  |      0.00|      0.00|      0.00|
709 |  |  |  30906810118800680000  |      0.00|      0.00|      0.00|
710 |  |  |  30906810018900680000  |      0.00|      0.00|      0.00|
711 |  |  |  30906810219000680000  |      0.00|      0.00|      0.00|
712 |  |  |  30906810119100680000  |      0.00|      0.00|      0.00|
713 |  |  |  30906810019200680000  |      0.00|      0.00|      0.00|
714 |  |  |  30906810919300680000  |      0.00|      0.00|      0.00|
715 |  |  |  30906810819400680000  |      0.00|      0.00|      0.00|
716 |  |  |  30906810719500680000  |      0.00|      0.00|      0.00|
717 |  |  |  30906810619600680000  |      0.00|      0.00|      0.00|
718 |  |  |  30906810519700680000  |      0.00|      0.00|      0.00|
719 |  |  |  30906810419800680000  |      0.00|      0.00|      0.00|
720 |  |  |  30906810319900680000  |      0.00|      0.00|      0.00|
721 |  |  |  30906810520100680000  |      0.00|      0.00|      0.00|
722 |  |  |  30906810320300680000  |      0.00|      0.00|      0.00|
723 |  |  |  30906810220400680000  |      0.00|      0.00|      0.00|
724 |  |  |  30906810120500680000  |      0.00|      0.00|      0.00|
725 |  |  |  30906810920700680000  |      0.00|      0.00|      0.00|
726 |  |  |  30906810820800680000  |      0.00|      0.00|      0.00|
727 |  |  |  30906810921000680000  |      0.00|      0.00|      0.00|
728 |  |  |  30906810721200680000  |      0.00|      0.00|      0.00|
729 |  |  |  30906810521400680000  |      0.00|      0.00|      0.00|
730 |  |  |  30906810421500680000  |      0.00|      0.00|      0.00|
731 |  |  |  30906810221700680000  |      0.00|      0.00|      0.00|
732 |  |  |  30906810121800680000  |      0.00|      0.00|      0.00|
733 |  |  |  30906810222000680000  |      0.00|      0.00|      0.00|
734 |  |  |  30906810022200680000  |      0.00|      0.00|      0.00|
735 |  |  |  30906810822400680000  |      0.00|      0.00|      0.00|
736 |  |  |  30906810722500680000  |      0.00|      0.00|      0.00|
```

Screenshot

What does it do and how it works

Lazy Text View widget is intended to display text on a web-page. The key feature is that it does not load whole text in the browser memory, but it displays only fragment (frame) of file. This allows to display large, very large, huge texts. The widget allows to scroll text up and down using `arrow keys`, `PgUp`, `PgDn`, `Home`, `End` as well as using mouse. On the left side line numbers are displayed and green indicator, which displays the current view position. Unfortunately, current version supports incremental scrolling only, it doesn't have vertical scrollbar, which could allow to jump to any random view position.

Requirements

Lazy Text View widget is written using JavaScript and jQuery. So, it is required to include jQuery. Widget was tested with jQuery v. 1.4.1. Also jquery-mousewheel and jquery-toaster plugins are required.

Architecture

The widget provides user interface for text display and requires server-side data source. You have to implement server-side component yourself, it's logic is quite simple. When the widget needs next chunk of text, it queries server (using POST-method) for the next chunk. The request specifies data source ID, starting number of string and quantity of requested strings. Also request specifies a direction: forward or reverse. So, you have to write program (servlet, controller etc...), which would be able to output some number of text strings starting from specified string in forward or reverse direction. You may use any programming language: Java, PHP, Pyton.... it doesn't matter. Your controller just must accept POST-request and output JSON-data. The widget accepts data in JSON format. The structure of request and expecting JSON-data are described below.

Embedding widget to web-page

Embedding is quite simple. Sample code is below:

```
<html>
   <head>
     <-- STEP 1: Include needed scripts and CSS -->
     <script type="text/javascript" src="jquery.js" ></script> <-- Include jQuery
-->
     <script type="text/javascript" src="jquery.mousewheel.js" > <-- Include
jQuery Mouse Wheel support -->
     <script type="text/javascript" src="jquery.toaster.js" > <-- Include jQuery
Toaster support -->
     <script type="text/javascript" src="lazyviewer.js"></script> <-- Include Lazy
View widget script -->
     <script type="text/javascript" src="lazyvwajax.js"></script> <-- Include Lazy
View ajax executor script -->

     <link rel="stylesheet" href="lazyviewer.css"> <-- Include Lazy View widget
CSS -->
     <link rel="stylesheet" href="toaster.css"> <-- Include jQuery Toaster CSS -->
  ...

</head>
<body>

  ...

   <-- STEP 2: Create a placeholder for widget -->
   <div id="lazyviewer" class="lazyview" ></div>

  ...

   <script type="text/javascript" language="javascript" >

     // STEP 3: Run Lazy View widget
     var lzv = new LazyViewer("lazyviewer", "/resources/img/");
     lzv.init(10, "lzv.htm", lzvSessId, autoSize);

   </script>
```

STEP 1: Include needed scripts and CSS

Include `jquery.js`, `jquery.mousewheel.js`, `jquery.toaster.js` scripts to your page. Also include`lazyviewer.js` and `lazyvwajax.js` scripts - main LZV widget scripts. Also include `toaster.css` and`lazyviewer.css`.

STEP 2: Create a placeholder for widget

Create a `<div>` tag in your HMTL code. This `<div>` will host future widget. The `<div>` must have some `ID` and compulsory `class lazyview`. The `ID` is specified in widget constructor to connect widget with `<div>` (see STEP 4).

STEP 3: Initialize LZV widget

Just call the following script:

```
var lzv = new LazyViewer("lazyviewer", "/resources/img/");
lzv.init(10, "lzv.htm", lzvSessId, autoSize);
```

The script constructs and initializes LZV widget. Constructor receives 2 arguments:

1. `ID` of `<div>`-placeholder (see STEP 3)
2. dowload path to folder containing images `u.gif` and `v.gif.` Images are stored in widget distributive archive.

Function `init()` has 4 parameters:

1. margin of widget (in pixels)
2. URL of server-side controller, which gives strings of text
3. data source ID (`lzvSessId`)
4. `autoSize –` boolean flag, which enables automatic resize (see below)

Data source ID links server-side data source with client-side widget. When widget queries server for some text to display, it needs to specify which text it requires. It may be some name of text file, may be a date of a logfile etc... some value, which identifies displaying text.

Server-side controller

When the widget needs next chunk of text, it queries server (using POST-method) for the next chunk. The request specifies data source ID, starting number of string and quantity of requested strings. Also request specifies a direction: forward or reverse. So, you have to write program (servlet, controller etc...), which would be able to output some number of text strings starting from specified string in forward or reverse direction. You may use any programming language: Java, PHP, Pyton.... it doesn't matter. Your controller just must accept POST-request and output JSON-data.

POST-request to server contains 4 fields:

1. `id` - data source ID. The type is string.
2. `ix` - starting number of string (zero-based). If ix is set to -1, this means the last string of text. The type is integer number.
3. `cnt` - count of requested strings. The type is integer number.
4. `dir` - direction. It may contain one of two values: "F" or "R". "F" means forward, "R" means reverse. The type is string.

Note, `ix` specifies the starting number of string in text. It is zero-based. Also `"ix"` can be set to -1, this means the last string of text. Widget requests last strings of text when user presses `End` key. So, your controller should be able to find the beginning of the last string. Sometimes it is not easy while working with streaming texts.

Data structure

Server-side controller should return JSON data in the following structure:

```
{ "data" : {
  "id" : data_source_id,
  "count" : string_count,
  "dir" : direction,
  "end" : last_index,
  "items" : [
     { "ix" : 0,   "txt" : "string one" },
     { "ix" : 1,   "txt" : "string two" },
     { "ix" : 2,   "txt" : "string three" }
     . . . . .
    ]
  }
}
```

- `id` - data source ID
- `count` - count of strings in "items" collection
- `dir` - direction. It may contain one of two values: "F" or "R". "F" means forward, "R" means reverse
- `end` - last index of whole text (whole text, not chunk!).
- `items` - collection of strings

Each string is described by 2 fields:

- `ix` - index of string (zero-based)
- `txt` - string value

Widget distributive contains "`java`" folder, which contains Java-files, describing serializable data compatible with LZV.

Resize modes

The widget supports 2 resize modes. The mode is controlled by parameter #4 of `init()` function (see STEP 3). If this parameter (`autoSize`) is set to `true`, widget automatically resizes to fit whole screen. In this mode widget must have no neighbour elements on the right, on the left and on the bottom. Widget sets it's width equal to width of browser window and aligns it's bottom border to the browser bottom border.

It autosize mode is disabled, widget is resized as common HTML element (`<div>`), so, you have to control it's size using script and CSS. It is recommended to call widget's `.onResize()` function when widget is resized. This function repositions widget and it's elements.

License

Donation

The software is free, and I hope that you find it useful. If you'd like to support future development and new product features, please make a donation via PayPal - a secure online banking service.

Please, use one of the the following links to make a donation from either your PayPal account or using one of the major credit cards.

Donate **$5**	http://www.paypal.me/ipolyakoff/5usd
Donate **$10**	http://www.paypal.me/ipolyakoff/10usd
Donate **$15**	http://www.paypal.me/ipolyakoff/15usd
Donate **$20**	http://www.paypal.me/ipolyakoff/20usd

The above method of donating is secure. PayPal guarantees your privacy and security. I never receive details of your payment other than the amount, your name, and your optional entered information.

Many thanks to those of you have already made a donation. It is truly appreciated. Thank you for your support!

Contacts

Internet: http://polyakoff.ucoz.net/ e-mail: i.polyakoff@inbox.ru