Examples for JQuery Terminal Emulator Plugin

[Originalartikel](#)

[Backup](#)

<html> <head><meta charset=„utf-8“/><title>Examples for JQuery Terminal Emulator Plugin</title><meta name=„author“ content=„Jakub Jankiewicz - jcubic@onet.pl“/><meta name=„Description“ content=„This is a bunch of usefull things that you can do with jQuery Terminal Emulator plugin. Live demos and source code likewise.“/><meta name=„keywords“ content=„jquery,terminal,interpreter,console,bash,history,authentication,ajax,server,client“/><!– Other files –><!– Terminal Files –><!–[if IE]>

```
  <![endif]--></head><body class="readabilityBody" readability="503">
  <header id="main" readability="12">
  <a href="http://terminal.jcubic.pl/" readability="21"><pre id="sig"
readability="33">
```

<p>

```
     __  ______                      _________                         __
    /  //  _   /__  __  ______  ____  __  _/__   ____/__  ____  _______  __  __   __  ____   /  /
```

/ _ _ / / _ _ \/ _ \/ / / / _ / / / / _ / / / / /\ / / _ __/ __ / / __/ / _/ / ___/ / _/ __/ \/ // 1.4.3 </p> <p> / _ / _/ _ _ / / / / / _ _ \/ _ \/ / / / / / _ / / / / /\ / / _ \ / __/ / _/ / ___/ / _/ __/ \/ 1.4.3 </p> <p> */ _ / _/ _ / / / _ _ / / / / / _ / / / / / _ \ /__/ /_/ / _/ \/ 1.4.3 </p> </pre><!– for FB bigger then gihub ribbon –> <pre class=„separator“>——</pre> </header> <section readability=„315“><article><header id=„examples“/>
JSON-RPC with authentication Simple AJAX example Autocomplete <abbr title=„Cross-Site Request Forgery“>CSRF</abbr> SQL Syntax highlighter Quake like terminal Terminal in jQuery UI Dialog Multiple interpreters Star Wars Animation Ask before executing a command Animation that emulate user typing Progress bar animation Spinners animation Less bash command Bash history commands
*

*
Smooth CSS3 cursor animation Virtual Keyboard with Terminal History API for commands Shell Commodore 64 404 Error Page In the wild </article><article><!– black wide –><ins class=„adsbygoogle“ data-ad-client=„ca-pub-6153701670678834“ data-ad-slot=„5835458303“/> </article><article id=„json_rpc_demo“ readability=„62“><header><h2>JSON-RPC with authentication</h2></header> See demo in action. (If you want to copy code from examples click “toogle highlight” first) Javascript code: <pre class=„javascript“>jQuery(function($) { $('#term').terminal(„json-rpc-service-demo.php“, { login: true, greetings: „You are authenticated“}); });</pre> PHP code (in rpc_demo.php): <pre class=„php“><?php require('json_rpc.php');   class Demo { static $login_documentation = „return auth token“; public function login($user, $passwd) { if (strcmp($user, 'demo') == 0 && strcmp($passwd, 'demo') == 0) {* If you need to handle more than one user you can

```
    // create new token and save it in database
    return md5($user . ":" . $passwd);
  } else {
    throw new Exception("Wrong Password");
  }
}
```



```
static $ls_documentation = "list directory if token is" .
   " valid";
public function ls($token, $path) {
  if (strcmp(md5("demo:demo"), $token) == 0) {
    if (preg_match("/\.\./", $path)) {
      throw new Exception("No directory traversal Dude");
    }
    $base = preg_replace("/(.*\/).*/", "$1",
                         $_SERVER["SCRIPT_FILENAME"]);
    $path = $base . ($path[0] != '/' ? "/" : "") . $path;
    $dir = opendir($path);
    while($name = readdir($dir)) {
      $fname = $path."/".$name;
      if (!is_dir($name) &amp;amp;&amp;amp; !is_dir($fname)) {
        $list[] = $name;
      }
    }
    closedir($dir);
    return $list;
  } else {
    throw new Exception("Access Denied");
```

```
  }
}
```



```
static $whoami_documentation = "return user information";
public function whoami() {
  return array(
      "user-agent" =&gt; $_SERVER["HTTP_USER_AGENT"],
      "your ip" =&gt; $_SERVER['REMOTE_ADDR'],
      "referer" =&gt; $_SERVER["HTTP_REFERER"],
      "request uri" =&gt; $_SERVER["REQUEST_URI"]);
}
```

}   handle_json_rpc(new Demo()); ?></pre>

```
      <p><strong>NOTE:</strong> If you use json_rpc.php file (which handle
json-rpc) from the <a href="http://terminal.jcubic.pl/#download">package</a>
you have always help function which display all methods or documentation
strings if you provide them.</p>
      <p>If you want secure login you should generate random token in login
JSON-RPC function, and store it in database.<br/>For example: md5(time()).
You can also use <a
href="http://en.wikipedia.org/wiki/Secure_Sockets_Layer">SSL</a>.</p>
      <p>See <a title="JSON-RPC demo"
href="http://terminal.jcubic.pl/rpc-demo.html">demo in action</a>. login is
"demo" and password is "demo". Available command are "ls", "whoami", "help"
and "help [rpc-method]"</p>
      <p><strong>Hint:</strong> if you want full access to the shell you can
pass all commands (through AJAX/JSON-RPC) to php passthru function or create
CGI script that will call the shell (Some hosting services block access to
the shell from php but not from cgi script). You can also implement "cd"
bash functionality by storing current path in variable and pass that
variable with every command send to the server, you can implement dynamic
prompt using the same variable.</p>
    </article><article id="simple_ajax" readability="46"><header><h2>Simple
AJAX example</h2></header><p>If you for some reason don't want to use JSON-
RPC you can create interpreter that will echo ajax responses and simple php
script.</p>
      <pre class="javascript">$(function() {
  $('body').terminal(function(command, term) {
      term.pause();
      $.post('script.php', {command: command}).then(function(response) {
          term.echo(response).resume();
      });
  }, {
      greetings: 'Simple php example',
      onBlur: function() {
          return false;
      }
```

```
  });
```

});</pre>

```
      <p>From version 1.0.0 you can simplify that code using:</p>
      <pre class="javascript">$(function() {
  $('body').terminal(function(command, term) {
      return $.post('script.php', {command: command});
  }, {
      greetings: 'Simple php example',
      onBlur: function() {
          return false;
      }
  });
```

});</pre>

```
      <p><strong>NOTE:</strong> if you return a promise from interpreter it
will call pause, wait for the response, then echo the response when it
arrive and call resume.</p>
      <pre class="php">&lt;?php
```

if (isset($_POST['command'])) {

```
  echo "you typed '" . $_POST['command'] . "'.";
```

}</pre>

```
      <p>You can use different server side language instead of php.</p>
    </article><article
readability="58"><header><h2>Autocomplete</h2></header><p>Adding
autocomplete to terminal is simple use complete option with array or
function as in <a
href="http://terminal.jcubic.pl/api_reference.php#completion">api
documentation</a> or true value if you use JSON-RPC with
<code>system.describe</code> or object as interpreter.</p>
         <p>You can also create custom completion, for instance add, menu
with items that you can click on that's added on keypress, From version
0.12.0 of the terminal there are two new api methods <code><a
href="http://terminal.jcubic.pl/api_reference.php#complete">complete</a></co
de> and <code><a
href="http://terminal.jcubic.pl/api_reference.php#before_cursor">before_curs
or</a></code> that simplify the code.</p>
      <pre class="javascript">var ul;
```

var cmd; var empty = {

```
  options: [],
```

```
  args: []
```

}; var commands = {

```
  'get-command': {
      options: ['name', 'age', 'description', 'address'],
      args: ['clear']
  },
  'git': {
      args: ['commit', 'push', 'pull'],
      options: ['amend', 'hard', 'version', 'help']
  },
  'get-name': empty,
  'get-age': empty,
  'get-money': empty
```

}; var ul; var term = $('body').terminal($.noop, {

```
  onInit: function(term) {
      var wrapper =
term.cmd().find('.cursor').wrap('&lt;span/&gt;').parent()
          .addClass('cmd-wrapper');
      ul = $('&lt;ul&gt;&lt;/ul&gt;').appendTo(wrapper);
      ul.on('click', 'li', function() {
          term.insert($(this).text());
          ul.empty();
      });
  },
  keydown: function(e) {
      var term = this;
      // setTimeout because terminal is adding characters in keypress
      // we use keydown because we need to prevent default action for
      // tab and still execute custom code
      setTimeout(function() {
          ul.empty();
          var command = term.get_command();
          var name = command.match(/^([^\s]*)/)[0];
          if (name) {
              var word = term.before_cursor(true);
              var regex = new RegExp('^' + $.terminal.escape_regex(word));
              var list;
              if (name == word) {
                  list = Object.keys(commands);
              } else if (command.match(/\s/)) {
                  if (commands[name]) {
                      if (word.match(/^--/)) {
                          list = commands[name].options.map(function(option)
{
                              return '--' + option;
                          });
                      } else {
```

```
                            list = commands[name].args;
                        }
                    }
                }
                if (word.length &gt;= 2 &amp;&amp; list) {
                    var matched = [];
                    for (var i=list.length; i--;) {
                        if (regex.test(list[i])) {
                            matched.push(list[i]);
                        }
                    }
                    var insert = false;
                    if (e.which == 9) {
                        insert = term.complete(matched);
                    }
                    if (matched.length &amp;&amp; !insert) {
                        ul.hide();
                        for (var i=0; i&lt;matched.length; ++i) {
                            var str = matched[i].replace(regex, '');
                            $('&lt;li&gt;' + str +
'&lt;/li&gt;').appendTo(ul);
                        }
                        ul.show();
                    }
                }
            }
        }, 0);
        if (e.which == 9) {
            return false;
        }
    },
    onBlur: function() {
        return false;
    }
```

});</pre>

```
      <p>See <a href="http://codepen.io/jcubic/pen/MJyYEx?editors=0110">demo
in action</a>.</p>
    </article><article id="csrf" readability="34"><header><h2><abbr
title="Cross-Site Request Forgery">CSRF</abbr></h2></header><p>Example that
add CSRF Protection to the terminal:</p>
      <pre class="javascript">jQuery(function($) {
  var CSRF_HEADER = "X-CSRF-TOKEN";
  var csrfToken;
  $('').appendTo('body').terminal("test.php", {
      request: function(jxhr, request) {
          if (csrfToken) {
              jxhr.setRequestHeader(CSRF_HEADER, csrfToken);
```

```
            }
        },
        response: function(jxhr, response) {
            if (!response.error) {
                csrfToken = jxhr.getResponseHeader(CSRF_HEADER);
            }
        },
        width: 600,
        height: 480,
    });
```

});</pre>

```
    </article><article id="syntax_highlight"
readability="273"><header><h2>SQL Syntax highlighter</h2></header><p>Here is
example to how to add syntax highlighting for mysql keywords</p>
      <pre class="javascript">// mysql keywords
```

var uppercase = [

```
  'ACCESSIBLE', 'ADD', 'ALL', 'ALTER', 'ANALYZE', 'AND', 'AS', 'ASC',
  'ASENSITIVE', 'BEFORE', 'BETWEEN', 'BIGINT', 'BINARY', 'BLOB',
  'BOTH', 'BY', 'CALL', 'CASCADE', 'CASE', 'CHANGE', 'CHAR',
  'CHARACTER', 'CHECK', 'COLLATE', 'COLUMN', 'CONDITION',
  'CONSTRAINT', 'CONTINUE', 'CONVERT', 'CREATE', 'CROSS',
  'CURRENT_DATE', 'CURRENT_TIME', 'CURRENT_TIMESTAMP', 'CURRENT_USER',
  'CURSOR', 'DATABASE', 'DATABASES', 'DAY_HOUR', 'DAY_MICROSECOND',
  'DAY_MINUTE', 'DAY_SECOND', 'DEC', 'DECIMAL', 'DECLARE', 'DEFAULT',
  'DELAYED', 'DELETE', 'DESC', 'DESCRIBE', 'DETERMINISTIC',
  'DISTINCT', 'DISTINCTROW', 'DIV', 'DOUBLE', 'DROP', 'DUAL', 'EACH',
  'ELSE', 'ELSEIF', 'ENCLOSED', 'ESCAPED', 'EXISTS', 'EXIT',
  'EXPLAIN', 'FALSE', 'FETCH', 'FLOAT', 'FLOAT4', 'FLOAT8', 'FOR',
  'FORCE', 'FOREIGN', 'FROM', 'FULLTEXT', 'GRANT', 'GROUP', 'HAVING',
  'HIGH_PRIORITY', 'HOUR_MICROSECOND', 'HOUR_MINUTE', 'HOUR_SECOND',
  'IF', 'IGNORE', 'IN', 'INDEX', 'INFILE', 'INNER', 'INOUT',
  'INSENSITIVE', 'INSERT', 'INT', 'INT1', 'INT2', 'INT3', 'INT4',
  'INT8', 'INTEGER', 'INTERVAL', 'INTO', 'IS', 'ITERATE', 'JOIN',
  'KEY', 'KEYS', 'KILL', 'LEADING', 'LEAVE', 'LEFT', 'LIKE', 'LIMIT',
  'LINEAR', 'LINES', 'LOAD', 'LOCALTIME', 'LOCALTIMESTAMP', 'LOCK',
  'LONG', 'LONGBLOB', 'LONGTEXT', 'LOOP', 'LOW_PRIORITY',
  'MASTER_SSL_VERIFY_SERVER_CERT', 'MATCH', 'MEDIUMBLOB', 'MEDIUMINT',
  'MEDIUMTEXT', 'MIDDLEINT', 'MINUTE_MICROSECOND', 'MINUTE_SECOND',
  'MOD', 'MODIFIES', 'NATURAL', 'NOT', 'NO_WRITE_TO_BINLOG', 'NULL',
  'NUMERIC', 'ON', 'OPTIMIZE', 'OPTION', 'OPTIONALLY', 'OR', 'ORDER',
  'OUT', 'OUTER', 'OUTFILE', 'PRECISION', 'PRIMARY', 'PROCEDURE',
  'PURGE', 'RANGE', 'READ', 'READS', 'READ_WRITE', 'REAL',
  'REFERENCES', 'REGEXP', 'RELEASE', 'RENAME', 'REPEAT', 'REPLACE',
  'REQUIRE', 'RESTRICT', 'RETURN', 'REVOKE', 'RIGHT', 'RLIKE',
  'SCHEMA', 'SCHEMAS', 'SECOND_MICROSECOND', 'SELECT', 'SENSITIVE',
  'SEPARATOR', 'SET', 'SHOW', 'SMALLINT', 'SPATIAL', 'SPECIFIC',
  'SQL', 'SQLEXCEPTION', 'SQLSTATE', 'SQLWARNING', 'SQL_BIG_RESULT',
```

```
  'SQL_CALC_FOUND_ROWS', 'SQL_SMALL_RESULT', 'SSL', 'STARTING',
  'STRAIGHT_JOIN', 'TABLE', 'TERMINATED', 'THEN', 'TINYBLOB',
  'TINYINT', 'TINYTEXT', 'TO', 'TRAILING', 'TRIGGER', 'TRUE', 'UNDO',
  'UNION', 'UNIQUE', 'UNLOCK', 'UNSIGNED', 'UPDATE', 'USAGE', 'USE',
  'USING', 'UTC_DATE', 'UTC_TIME', 'UTC_TIMESTAMP', 'VALUES',
  'VARBINARY', 'VARCHAR', 'VARCHARACTER', 'VARYING', 'WHEN', 'WHERE',
  'WHILE', 'WITH', 'WRITE', 'XOR', 'YEAR_MONTH', 'ZEROFILL'];
```

var keywords = uppercase.concat(uppercase.map(function(keyword) {

```
  return keyword.toLowerCase();
```

})); $.terminal.defaults.formatters.push(function(string) {

```
  return string.split(/((?:\s|&amp;nbsp;)+)/).map(function(string) {
      if (keywords.indexOf(string) != -1) {
          return '[[b;white;]' + string + ']';
      } else {
          return string;
      }
  }).join('');
```

});</pre>

```
      <p>If you want to add formatting for different sql command and not for
main interpterer you can use stack of formatters. It require version
&gt;=1.0 that introduce extra option for interpreter. The example will work
for any number of nested interpreters even you call push new in your mysql
command.</p>
      <pre class="javascript">// this regex will allow mixed case like
SeLect
```

var re = new RegExp('^(' + uppercase.join('|') + ')$', 'i'); function mysql_formatter(string) {

```
  return string.split(/((?:\s|&amp;nbsp;)+)/).map(function(string) {
      if (re.test(string)) {
          return '[[b;white;]' + string + ']';
      } else {
          return string;
      }
  }).join('');
```

} var formatters = [$.terminal.defaults.formatters]; $('body').terminal(function(command, term) {

```
  if (command.match(/^\s*mysql\s*$/)) {
      term.push(function(query) {
          term.echo('executing ' + query, {formatters: false});
      }, {
          prompt: 'mysql&gt; ',
```

```
          name: 'mysql',
          // extra property saved in interpreter
          formatters: [mysql_formatter],
          completion: keywords
      });
  }
```

}, {

```
  onPush: function(before, after) {
      $.terminal.defaults.formatters = after.formatters || [];
      formatters.push($.terminal.defaults.formatters);
  },
  onPop: function(before, after) {
      formatters.pop();
      if (formatters.length &gt; 0) {
          $.terminal.defaults.formatters = formatters[formatters.length-1];
      }
  }
```

});</pre>

```
    </article><article id="tilda" readability="38"><header><h2>Quake like
terminal</h2></header><p>See <a
href="http://terminal.jcubic.pl/tilda-demo.html">demo</a>.</p>
      <p>Below is code for small plugin called tilda.</p>
      <pre class="javascript">(function($) {
  $.fn.tilda = function(eval, options) {
      if ($('body').data('tilda')) {
          return $('body').data('tilda').terminal;
      }
      this.addClass('tilda');
      options = options || {};
      eval = eval || function(command, term) {
          term.echo("you don't set eval for tilda");
      };
      var settings = {
          prompt: 'tilda&gt; ',
          name: 'tilda',
          height: 100,
          enabled: false,
          greetings: 'Quake like console',
          keypress: function(e) {
              if (e.which == 96) {
                  return false;
              }
          }
      };
      if (options) {
          $.extend(settings, options);
      }
```

```
      this.append('&lt;div class="td"&gt;&lt;/div&gt;');
      var self = this;
      self.terminal = this.find('.td').terminal(eval,
                                                settings);
      var focus = false;
      $(document.documentElement).keypress(function(e) {
          if (e.charCode == 96) {
              self.slideToggle('fast');
              self.terminal.command_line.set('');
              self.terminal.focus(focus = !focus);
          }
      });
      $('body').data('tilda', this);
      this.hide();
      return self;
  };
```

})(jQuery);</pre>

```
      <p>See <a
href="http://terminal.jcubic.pl/tilda-demo.html">demo</a>.</p>
    </article><article readability="64"><header id="dterm"><h2>Terminal in
jQuery UI Dialog</h2></header><p>Bellow is small plugin dterm.</p>
      <pre class="javascript">(function($) {
  $.extend_if_has = function(desc, source, array) {
      for (var i=array.length;i--;) {
          if (typeof source[array[i]] != 'undefined') {
              desc[array[i]] = source[array[i]];
          }
      }
      return desc;
  };
  $.fn.dterm = function(interpeter, options) {
      var defaults = Object.keys($.terminal.defaults);
      var op = $.extend_if_has({}, options, defaults);
```



```
      var term = this.append('&lt;div/&gt;').
            terminal(interpeter, op);
      if (!options.title) {
          options.title = 'JQuery Terminal Emulator';
      }
      if (options.logoutOnClose) {
          options.close = function(e, ui) {
              term.logout();
              term.clear();
          };
      } else {
```

```
          options.close = function(e, ui) {
              term.focus(false);
          };
      }
      var self = this;
      var dialog = this.dialog($.extend(options, {
          resize: function(e, ui) {
              var c = self.find('.ui-dialog-content');
              term.resize(c.width(), c.height());
          },
          open: function(e, ui) {
              term.focus();
          },
          closeOnEscape: false
      }));
      this.terminal = term;
      return this;
  };
```

})(jQuery);</pre>

```
      <p id="biwascheme"><strong>Demo Scheme interpreter inside JQuery UI
Dialog.</strong></p>
      <p>Click on button to  with scheme interpreter inside UI Dialog.</p>
      <p><strong>Hint:</strong> you can use JQuery from scheme. There is
defined $ function and functions for all jquery object methods, they names
start with coma and they always return jquery object so you can do
chaining.</p>
      <p><strong>NOTE:</strong> you should include jQuery Terminal css file
after jQuery UI one otherwise you will have white text in terminal, insided
of gray.</p>
      <p>Interpreter allow to use <strong>multiline expressions</strong>.
When you type not finished S-Expresion it change the prompt with set_prompt,
contatenate current command with previous not finished expression and when
you close last parentises end press enter it evaluate whole expression.</p>
      <p>If you want to call:</p>
      <pre class="javascript">$("body").css("background-color",
"black");</pre>
      <p>use</p>
      <!-- only for syntax highlight -->
      <pre class="javascript">(.css ($ "body") "background-color"
"black")</pre>
      <p>To attach event you can use lambda expressions.</p>
      <pre class="javascript">(.click ($ ".terminal") (lambda () (display
"click")))</pre>
      <p>this will attach click event to terminal.</p>
    </article><article id="multiple-interpreters"
readability="72"><header><h2>Multiple interpreters</h2></header><p>All
interpreters are stored on the stack which which you can manipulate with
terminal methods pop an push.</p>
      <p>See <a title="JQuery Terminal Emulator Demo"
```

```
href="http://terminal.jcubic.pl/multiply-interpreters-demo.html">demo</a>.</
p>
      <p>In belowed code there are defied three commands:</p>
      <ul><li>js - which run javascript interpreter</li>
        <li>mysql - which call json-rpc service to execute mysql
commands.</li>
        <li>test - it display "pong" if you type "ping" </li>
      </ul><pre class="javascript">jQuery(function($) {
$('html').terminal(function(cmd, term) {
  if (cmd == 'help') {
    term.echo("available commands are mysql, js, test");
  } else if (cmd == 'test'){
    term.push(function(cmd, term) {
      if (command == 'help') {
        term.echo('type "ping" it will display "pong"');
      } else if (cmd == 'ping') {
        term.echo('pong');
      } else {
        term.echo('unknown command "' + cmd + '"');
      }
    }, {
      prompt: 'test&gt; ',
      name: 'test'});
  } else if (command == "js") {
    term.push(function(command, term) {
      var result = window.eval(command);
      if (result != undefined) {
        term.echo(String(result));
      }
    }, {
      name: 'js',
      prompt: 'js&gt; '});
    } else if (command == 'mysql') {
      term.push(function(command, term) {
        term.pause();
        //$.jrpc is helper function which
        //creates json-rpc request
        $.jrpc("mysql-rpc-demo.php",
          "query",
          [command],
          function(data) {
            term.resume();
            if (data.error) {
              if (data.error.error &amp;&amp; data.error.error.message) {
                term.error(data.error.error.message); // php error
              } else {
                term.error(data.error.message); // json rpc error
              }
            } else {
              if (typeof data.result == 'boolean') {
```

```
                term.echo(data.result ?
                          'success' :
                          'fail');
              } else {
                var len = data.result.length;
                for(var i=0;i&lt;len; ++i) {
                  term.echo(data.result[i].join(' | '));
                }
              }
            }
          },
          function(xhr, status, error) {
            term.error('[AJAX] ' + status +
                       ' - Server reponse is: \n' +
                       xhr.responseText);
                       term.resume();
                   }); // rpc call
        }, {
          greetings: "This is example of using mysql"+
            " from terminal\n you are allowed to exe"+
            "cute: select, insert, update and delete"+
            " from/to table:\n   table test(integer_"+
            "value integer, varchar_value varchar(255))",
          prompt: "mysql&gt; "});
        } else {
          term.echo("unknow command " + command);
        }
       }, {
         greetings: "multiply terminals demo use help"+
               " to see available commands"
     });});</pre>
      <p>If you want to display ascii table like real mysql command, take a look at <a href="https://github.com/jcubic/leash/blob/1843d8f4dd9f2e4696f2086184c23624027acb9f/leash-src.js#L511">asci_table function in leash project</a>, it use <a href="https://github.com/timoxley/wcwidth">wcwidth</a> to calcuate the width of the characters but if you don't care about chenese characters you can replace it with <code>string.length</code>.</p>
      <p>PHP code for mysql service: </p>
      <pre class="php">&lt;?php
```

require('json_rpc.php'); $conn = mysql_connect('localhost', 'user', 'password'); mysql_select_db('database');   class MysqlDemo {

```
public function query($query) {
  if (preg_match("/create|drop/", $query)) {
    throw new Exception("Sorry you are not allowed to ".
                        "execute '" . $query . "'");
  }
  if (!preg_match("/(select.*from *test|insert *into *".
                  "test.*|delete *from *test|update *t".
```

```
                  "est)/", $query)) {
    throw new Exception("Sorry you can't execute '" .
                        $query . "' you are only allow".
                        "ed to select, insert, delete ".
                        "or update 'test' table");
  }
  if ($res = mysql_query($query)) {
    if ($res === true) {
      return true;
    }
    if (mysql_num_rows($res) &gt; 0) {
      while ($row = mysql_fetch_row($res)) {
        $result[] = $row;
      }
      return $result;
    } else {
      return array();
    }
  } else {
    throw new Exception("MySQL Error: ".mysql_error());
  }
}
```

}   handle_json_rpc(new MysqlDemo()); ?></pre>

```
    <p>See <a title="JQuery Terminal Emulator Demo"
href="http://terminal.jcubic.pl/multiply-interpreters-demo.html">demo</a>.</
p>
    </article><article id="starwars" readability="39"><header><h2>Star Wars
Animation</h2></header><p>This is Star Wars ASCIIMation created by Simon
Jansen <br/><a
href="http://www.asciimation.co.nz/">http://www.asciimation.co.nz/</a></p>
      <pre class="javascript">$(function() {
  var frames = [];
  var LINES_PER_FRAME = 14;
  var DELAY = 67;
  //star_wars is array of lines from 'js/star_wars.js'
  var lines = star_wars.length;
  for (var i=0; i&lt;lines; i+=LINES_PER_FRAME) {
      frames.push(star_wars.slice(i, i+LINES_PER_FRAME));
  }
  var stop = false;
  //to show greetings after clearing the terminal
  function greetings(term) {
      term.echo('STAR WARS ASCIIMACTION\n'+
                'Simon Jansen (C) 1997 - 2008\n'+
                'www.asciimation.co.nz\n\n'+
                'type "play" to start animation, '+
                'press CTRL+D to stop');
```

```
}
function play(term, delay) {
    var i = 0;
    var next_delay;
    if (delay == undefined) {
        delay = DELAY;
    }
    function display() {
        if (i == frames.length) {
            i = 0;
        }
        term.clear();
        if (frames[i][0].match(/[0-9]+/)) {
            next_delay = frames[i][0] * delay;
        } else {
            next_delay = delay;
        }
        term.echo(frames[i++].slice(1).join('\n')+'\n');
        if (!stop) {
            setTimeout(display, next_delay);
        } else {
            term.clear();
            greetings(term);
            i = 0;
        }
    }
    display();
}
$('#starwarsterm').terminal(function(command, term){
    if (command == 'play') {
        term.pause();
        stop = false;
        play(term);
    }
}, {
    width: 500,
    height: 230,
    prompt: 'starwars&gt; ',
    greetings: null,
    onInit: function(term) {
        greetings(term);
    },
    keypress: function(e, term) {
        if (e.which == 100 &amp;&amp; e.ctrlKey) {
            stop = true;
            term.resume();
            return false;
        }
    }
});
```

});</pre>

```
    </article><article id="ask" readability="34"><header><h2>Ask before
executing a command</h2></header><p>Someone ask me how to create, command
that ask users before executing, and here is the code, it will keep asking
until eather yes or no will be entered (or short y/n).</p>
      <pre class="javascript">$('#term').terminal(function(command, term) {
  if (command == 'foo') {
      var history = term.history();
      history.disable();
      term.push(function(command) {
          if (command.match(/^(y|yes)$/i)) {
              term.echo('execute your command here');
              term.pop();
              history.enable();
          } else if (command.match(/^(n|no)$/i)) {
              term.pop();
              history.enable();
          }
      }, {
          prompt: 'Are you sure? '
      });
  }
```

});</pre>

```
    </article><article id="user-typing"
readability="61"><header><h2>Animation that emulate user
typing</h2></header><p>Someone else aks if it's posible to create animation
like user typing. Here is the code that emulate user typing on
initialization of the terminal and before every ajax call, which can finish
after animation.</p>
      <pre class="javascript">$(function() {
  var anim = false;
  function typed(finish_typing) {
      return function(term, message, delay, finish) {
          anim = true;
          var prompt = term.get_prompt();
          var c = 0;
          if (message.length &gt; 0) {
              term.set_prompt('');
              var interval = setInterval(function() {
                  term.insert(message[c++]);
                  if (c == message.length) {
                      clearInterval(interval);
                      // execute in next interval
                      setTimeout(function() {
                          // swap command with prompt
                          finish_typing(term, message, prompt);
```

```
                        anim = false
                        finish &amp;&amp; finish();
                    }, delay);
                }
            }, delay);
        }
    };
}
var typed_prompt = typed(function(term, message, prompt) {
    // swap command with prompt
    term.set_command('');
    term.set_prompt(message + ' ');
});
var typed_message = typed(function(term, message, prompt) {
    term.set_command('');
    term.echo(message)
    term.set_prompt(prompt);
});
$('body').terminal(function(cmd, term) {
    var finish = false;
    var msg = "Wait I'm executing ajax call";
    term.set_prompt('&gt; ');
    typed_message(term, msg, 200, function() {
        finish = true;
    });
    var args = {command: cmd};
    $.get('commands.php', args, function(result) {
        (function wait() {
            if (finish) {
                term.echo(result);
            } else {
                setTimeout(wait, 500);
            }
        })();
    });
}, {
    name: 'xxx',
    greetings: null,
    width: 500,
    height: 300,
    onInit: function(term) {
        // first question
        var msg = "Wellcome to my terminal";
        typed_message(term, msg, 200, function() {
            typed_prompt(term, "what's your name:", 100);
        });
    },
    keydown: function(e) {
        //disable keyboard when animating
        if (anim) {
            return false;
```

```
          }
      }
  });
```

});</pre>

```
    </article><article id="progress-bar"
readability="41"><header><h2>Progress bar animation</h2></header><p>You can
test it by executing command `progress 30`.</p>
      <p>Here is the code for progres bar animation:</p>
      <pre class="javascript">jQuery(function($) {
  function progress(percent, width) {
      var size = Math.round(width*percent/100);
      var left = '', taken = '', i;
      for (i=size; i--;) {
          taken += '=';
      }
      if (taken.length &gt; 0) {
          taken = taken.replace(/=$/, '&gt;');
      }
      for (i=width-size; i--;) {
          left += ' ';
      }
      return '[' + taken + left + '] ' + percent + '%';
  }
  var animation = false;
  var timer;
  var prompt;
  var string;
  $('body').terminal(function(command, term) {
      var cmd = $.terminal.parse_command(command);
      if (cmd.name == 'progress') {
          var i = 0, size = cmd.args[0];
          prompt = term.get_prompt();
          string = progress(0, size);
          term.set_prompt(progress);
          animation = true;
          (function loop() {
              string = progress(i++, size);
              term.set_prompt(string);
              if (i &lt; 100) {
                  timer = setTimeout(loop, 100);
              } else {
                  term.echo(progress(i, size) + ' [[b;green;]OK]')
                      .set_prompt(prompt);
                  animation = false
              }
          })();
      }
```

```
  }, {
      keydown: function(e, term) {
          if (animation) {
              if (e.which == 68 &amp;&amp; e.ctrlKey) { // CTRL+D
                  clearTimeout(timer);
                  animation = false;
                  term.echo(string + ' [[b;red;]FAIL]')
                      .set_prompt(prompt);
              }
              return false;
          }
      }
  });
```

});</pre>

```
    </article><!--
    TODO:
    <article>
http://labs.funkhausdesign.com/examples/terminal/cmd_controlled_terminal.htm
l
    </article>
    --><article id="spinners" readability="43"><header><h2>Spinners
animation</h2></header><p>Spinner animations from <a
href="https://github.com/sindresorhus/cli-spinners">sindresorhus/cli-spinner
s</a>.</p>
      <pre class="javascript">$(function() {
  var url = 'https://rawgit.com/sindresorhus/cli-spinners/master/' +
            'spinners.json';
  $.getJSON(url, function(spinners) {
      var animation = false;
      var timer;
      var prompt;
      var spinner;
      var i;
      function start(term, spinner) {
          animation = true;
          i = 0;
          function set() {
              var text = spinner.frames[i++ % spinner.frames.length];
              term.set_prompt(text);
          };
          prompt = term.get_prompt();
          term.find('.cursor').hide();
          set();
          timer = setInterval(set, spinner.interval);
      }
      function stop(term, spinner) {
          setTimeout(function() {
              clearInterval(timer);
              var frame = spinner.frames[i % spinner.frames.length];
```

```
                term.set_prompt(prompt).echo(frame);
                animation = false;
                term.find('.cursor').show();
            }, 0);
        }
        $('#spinners .term').terminal({
            spinner: function(name) {
                spinner = spinners[name];
                if (!spinner) {
                    this.error('Spinner not found');
                } else {
                    this.echo('press CTRL+D to stop');
                    start(this, spinner);
                }
            },
            help: function() {
                var text = Object.keys(spinners).join('\t');
                this.echo('Available spinners: ' + text, {
                    keepWords: true
                });
            }
        }, {
            greetings: false,
            onInit: function(term) {
                term.echo('Spinners, type [[b;#fff;]help] to display '+
                          'available spinners or [[b;#fff;]spinner &lt;n'+
                          'ame&gt;] for animation', {
                    keepWords: true
                });
            },
            completion: true,
            keydown: function(e, term) {
                if (animation) {
                    if (e.which == 68 &amp;&amp; e.ctrlKey) { // CTRL+D
                        stop(term, spinner);
                    }
                    return false;
                }
            }
        });
    });
```

});</pre>

```
    </article><article id="less" readability="42"><header><h2>Less bash
command</h2></header><p>Here is implementation of bash less command (not all
commands implemented)</p>
      <pre class="javascript">var resize = [];
```

$('<SELECTOR>').terminal(function(command, term) {

```
if (command.match(/ *less +[^ ]+/)) {
  term.pause();
  $.ajax({
    // leading and trailing spaces and keep those inside argument
    url: command.replace(/^\s+|\s+$/g, '').
      replace(/^ */, '').split(/(\s+)/).slice(2).join(''),
    method: 'GET',
    dataType: 'text',
    success: function(source) {
      term.resume();
      var export_data = term.export_view();
      var less = true;
      source = source.replace(/&amp;/g, '&amp;amp;').
        replace(/\[/g, '&amp;#91;').
        replace(/\]/g, '&amp;#93;');
      var cols = term.cols();
      var rows = term.rows();
      resize = [];
      var lines = source.split('\n');
      resize.push(function() {
        if (less) {
          cols = term.cols();
          rows = term.rows();
          print();
        }
      });
      var pos = 0;
      function print() {
        term.clear();
        term.echo(lines.slice(pos, pos+rows-1).join('\n'));
      }
      print();
      term.push($.noop, {
        keydown: function(e) {
          if (term.get_prompt() !== '/') {
            if (e.which == 191) {
              term.set_prompt('/');
            } else if (e.which === 38) { //up
              if (pos &gt; 0) {
                --pos;
                print();
              }
            } else if (e.which === 40) { //down
              if (pos &lt; lines.length-1) {
                ++pos;
                print();
              }
            } else if (e.which === 34) { // Page up
              pos += rows;
```

```
              if (pos &gt; lines.length-1-rows) {
                pos = lines.length-1-rows;
              }
              print();
            } else if (e.which === 33) { // page down
              pos -= rows;
              if (pos &lt; 0) {
                pos = 0;
              }
              print();
            } else if (e.which == 81) { //Q
              less = false;
              term.pop().import_view(export_data);
            }
            return false;
          } else {
            if (e.which === 8 &amp;amp;&amp;amp; term.get_command() === '') {
              term.set_prompt(':');
            } else if (e.which == 13) {
              var command = term.get_command();
              // basic search find only first
              // instance and don't mark the result
              if (command.length &gt; 0) {
                var regex = new RegExp(command);
                for (var i=0; i&lt;lines.length; ++i) {
                  if (regex.test(lines[i])) {
                    pos = i;
                    print();
                    term.set_command('');
                    break;
                  }
                }
                term.set_command('');
                term.set_prompt(':');
              }
              return false;
            }
          }
        },
        prompt: ':'
      });
    }
  });
}
```

},{

```
onResize: function(term) {
  for (var i=resize.length;i--;) {
```

```
    resize[i](term);
  }
}
```

});</pre>

```
      <p>Improved less command can be found in <a
href="https://github.com/jcubic/leash/blob/194e3e2574d6cc437c2d3b530ff84f41d
f9fb8d0/leash-src.js#L915">leash browser shell</a>.</p>
    </article><article id="bash-history" readability="60"><header><h2>Bash
history commands</h2></header><p>If you want to add bash history commands
like !!, !$ or !* here is the code:</p>
      <pre class="javascript">$('body').terminal(function(command, term) {
  var cmd = $.terminal.parse_command(command);
  if (command.match(/![*$]|\s*!!(:p)?\s*$|\s*!(.*)/)) {
      var new_command;
      var history = term.history();
      var last = $.terminal.parse_command(history.last());
      var match = command.match(/\s*!(?![!$*])(.*)/);
      if (match) {
          var re = new RegExp($.terminal.escape_regex(match[1]));
          var history_data = history.data();
          for (var i=history_data.length; i--;) {
              if (re.test(history_data[i])) {
                  new_command = history_data[i];
                  break;
              }
          }
          if (!new_command) {
              var msg = $.terminal.defaults.strings.commandNotFound;
              term.error(sprintf(msg,
$.terminal.escape_brackets(match[1])));
          }
      } else if (command.match(/![*$]/)) {
          if (last.args.length) {
              var last_arg = last.args[last.args.length-1];
              new_command = command.replace(/!\$/g, last_arg);
          }
          new_command = new_command.replace(/!\*/g, last.rest);
      } else if (command.match(/\s*!!(:p)?/)) {
          new_command = last.command;
      }
      if (new_command) {
          term.echo(new_command);
      }
      if (!command.match(/![*$!]:p/)) {
          if (new_command) {
              term.exec(new_command, true);
          }
      }
  } else if (cmd.name == 'echo') {
```

```
      term.echo(cmd.rest);
  }
```

}, {

```
  // we need to disable history for bash history commands
  historyFilter: function(command) {
      return !command.match(/![*$]|\s*!!(:p)?\s*$|\s*!(.*)/);
  }
```

});</pre>

```
    </article><article id="css-cursor" readability="63"><header><h2>Smooth
CSS3 cursor animation</h2></header><p>From version 0.8 terminal use CSS
animation for blinking so you can change it without touching JavaScript
code.</p>
      <p>Here is different looking cursor blinking animation that can be use
with terminal.</p>
      <pre class="css">@keyframes blink {
  50% {
      color: #000;
      background: #0c0;
      -webkit-box-shadow: 0 0 5px rgba(0,100,0,50);
      box-shadow: 0 0 5px rgba(0,100,0,50);
  }
```

} @-webkit-keyframes blink {

```
  50% {
      color: #000;
      background: #0c0;
      -webkit-box-shadow: 0 0 5px rgba(0,100,0,50);
      box-shadow: 0 0 5px rgba(0,100,0,50);
  }
```

} @-ms-keyframes blink {

```
  50% {
      color: #000;
      background: #0c0;
      -webkit-box-shadow: 0 0 5px rgba(0,100,0,50);
      box-shadow: 0 0 5px rgba(0,100,0,50);
  }
```

} @-moz-keyframes blink {

```
  50% {
      color: #000;
      background: #0c0;
```

```
      -webkit-box-shadow: 0 0 5px rgba(0,100,0,50);
      box-shadow: 0 0 5px rgba(0,100,0,50);
  }
```

} .terminal {

1. -background: #000;
2. -color: #0c0;

text-shadow: 0 0 3px rgba(0,100,0,50); } .cmd .cursor.blink {

1. webkit-animation: 1s blink infinite;

animation: 1s blink infinite;

1. webkit-box-shadow: 0 0 0 rgba(0,100,0,50);

box-shadow: 0 0 0 rgba(0,100,0,50);

```
  border: none;
  margin: 0;
```

}</pre>

```
    </article><article id="virtual" readability="23"><header><h2>Using
Virtual Keyboard with Terminal</h2></header><p>There are problems with
terminal on touch devices. I've found a project <a
href="https://github.com/Mottie/Keyboard">Keyboard</a> that create virtual
keyboard using jQuery UI. I've created a demo of working terminal with
keyboard. The code still need tweeks to work full screen.</p>
      <p>See <a
href="http://terminal.jcubic.pl/virtualKeyboard.html">demo</a></p>
    </article><article id="history" readability="78"><header><h2>Using
History API for commands</h2></header><p>As a response for this <a
href="https://github.com/jcubic/jquery.terminal/issues/148">issue on
github</a> I came up with a way to keep every command response in history
using HTML5 History API, so you can click back and forward buttons and it
will show you previous and next commands.</p>
      <pre class="javascript">$(function() {
  var save_state = [];
  var terminal = $('#term').terminal(function(command, term) {
      var cmd = $.terminal.split_command(command);
      var url;
      if (cmd.name == 'open') {
          term.pause();
          // open html and display it on terminal as it is
          url = cmd.args[0];
          $.get(url, function(result) {
              term.echo(result, {raw:true}).resume();
              save_state.push(term.export_view());
              history.pushState(save_state.length-1, null, url);
          }, 'text');
```

```
      } else {
          // store all other commands
          save_state.push(term.export_view());
          url = '/' + cmd.name + '/' + cmd.args.join('/');
          history.pushState(save_state.length-1, null, url);
      }
  });
  save_state.push(terminal.export_view()); // save initial state
  $(window).on('popstate', function(e) {
      if (save_state.length) {
          terminal.import_view(save_state[history.state || 0]);
      }
  });
```

});</pre>

```
      <p>Each command after it finish need to call this:
      </p><pre class="javascript">save_state.push(term.export_view());
```

history.pushState(save_state.length-1, null, '<NEW URL>');</pre>

```
      <p>So it keep current view of the terminal (after the command
finishes) in <code>save_state</code> array and index in push state (I've try
to put whole view in <code>history.state</code> but it didn't work). On
back/forward buttons click it will get that value from array and restore the
view of the terminal.</p>
      <p>Version 0.9.0 introduced similar API but using url hash. To enable
it use <code>historyState</code> option and to execute hash on load use
<code>execHash</code> option.</p>
    </article><article id="shell"
readability="24"><header><h2>Shell</h2></header><p>You can also check my
project <a href="http://leash.jcubic.pl">LEASH - Browser Shell</a> you will
have shell without need to install anything on the server (so you don't need
root access), it use lot of features of jQuery terminal, like better <a
href="http://terminal.jcubic.pl/examples.php#less">less command</a> or
python interpreter.</p>
      <p>You can also use <a
href="https://github.com/jcubic/leash-cordova">cordova application</a> that
use leash to have access to android shell.</p>
    </article><article id="c64" readability="8"><header><h2>Commodore
64</h2></header><p>You can check <a
href="http://terminal.jcubic.pl/commodore64">Commodore64 Demo inside vintage
monitor</a></p>
    </article><article id="wild"><header><h2>In the
wild</h2></header><ul><li>Games
          <ul><li><strike><a href="http://plusmineus.com/">PlusMineus</a>
&amp;#8212; a Survival Roleplay Minecraft Server.</strike></li>
            <li><a
href="http://rdebath.github.io/LostKingdom/">LostKingdom</a> &amp;#8212; text
```

```
based game.</li>
            <li><a href="http://gfc.albertocongiu.com/thelab/">The Lab</a>
&amp;#8212; game where you code in javascript.</li>
            <li><a
href="http://www.gamecreation.org/game/lunatix">Lunatix</a> &amp;#8212; a unix-
inspired educational text-based adventure game.</li>
            <li><a
href="http://textadventure.audio/">textadventure.audio</a> &amp;#8212; text
based Adventure Game.</li>
            <li><a
href="http://major-jack-bouler.azurewebsites.net/">major-jack-bouler</a>
&amp;#8212; another game using jQuery Terminal.</li>
            <li><a href="http://ihavard.net/">ihavard.net</a> &amp;#8212; text
based game that simulate life.</li>
            <li><a href="https://cmdchallenge.com/">cmdchallenge.com</a>
&amp;#8212; game in which you need to enter one liner bash commands.</li>
            <li><a
href="https://facundoolano.github.io/advenjure/">advenjure</a> &amp;#8212; Text
adventure engine written in Clojure and ClojureScript</li>
          </ul></li>
        <li>Interpreters, interfaces, Tools, APIs
          <ul><li><a href="http://warlab.info/">Tools for webmasters and
geeks by warlab.info.</a></li>
            <li><a href="http://biwascheme.org">BiwaScheme</a> &amp;#8212; use
the same code as in example.</li>
            <li><a
href="https://npmjs.org/package/node-web-repl">node-web-repl</a> &amp;#8212; Add
a web-based read/eval/print/loop to your Node.js app.</li>
            <li><a
href="http://niutech.github.io/typescript-interpret/">Typescript
Interpreter</a>.</li>
            <li><a
href="https://github.com/bearstech/PloneTerminal">PloneTerminal</a> &amp;#8212;
terminal for plone.</li>
            <li><a href="https://www.drupal.org/project/terminal">Drupal
Terminal</a> &amp;#8212; terminal for drupal.</li>
            <li><a href="https://github.com/cixtor/phpshellgen">PHP-Shell
Generator</a>.</li>
            <li><stike><a
href="https://www.docker.io/gettingstarted/">Docker</a> &amp;#8212; Docker.io
use terminal in it's interactive tutorial.</stike></li>
            <li><a href="https://github.com/glejeune/ews">Elixir Web
Shell</a>.</li>
            <li><a href="http://apps.splunk.com/app/1607/">Web Terminal for
Splunk</a>.</li>
            <li><a href="http://isay.monogra.fi/manhole/">Manhole</a>
&amp;#8212; A simple REPL into a running aspnet application.</li>
            <li><a href="http://leash.jcubic.pl">leash</a> &amp;#8212; unix
shell from the browser, lot of features of terminal.</li>
            <li><strike><a href="http://toretto.x10.mx/terminal.html">simple
use of terminal.</a></strike></li>
```

```
            <li><a
href="https://github.com/avalanche123/node-console">node-console</a> &#8212;
using of socket IO that respond to events.</li>
            <li><a href="http://try-groonga.herokuapp.com/">Try Groonga</a>
&#8212; Groonga is an open-source fulltext search engine and column store.
It lets you write high-performance applications that requires fulltext
search.</li>
            <li><a href="http://agnostic.housegordon.org/">AGNOSTIC</a>
&#8212; UNIX Shell Javascript Emulation</li>
            <li><a href="http://the-james-burton.github.io/sshw/">sshw</a>
&#8212; SSH client in a browser, via a JEE webapp.</li>
            <li><a href="http://calc.nutpan.com/">Online
calculator</a>.</li>
            <li><a href="http://www.kvstore.io/">kvstore.io</a> &#8212; The
Simple &lt;key,value&gt; Storage Service.</li>
            <li><a href="http://www.web-console.org/">web-console</a>
&#8212; Web Console is a web-based application that allows to execute shell
commands on a server directly from a browser (web-based SSH).</li>
            <li><a href="http://samy.pl/keysweeper/">KeySweeper</a> &#8212;
use terminal to show live keyboard keystrokes logged.</li>
            <li><a href="http://jobfeeds.info/devops/">devops jobs</a>.</li>
            <li><a
href="https://github.com/AlexNisnevich/ECMAchine">ECMAchine</a> &#8212;
Lisp-based in-browser toy operating system.</li>
            <li><a
href="https://www.npmjs.com/package/lightpost">lightpost</a> &#8212; A
lightweight language based on postfix notation.</li>
            <li><a
href="https://packagist.org/packages/samdark/yii2-webshell">yii2-webshell</a
> &#8212; A web shell that allows to run yii console commands and create
your own commands.</li>
            <li><a
href="https://github.com/hauckd/terminalCV">terminalCV</a> &#8212; A command
line CV for sysadmins.</li>
            <li><a
href="http://www.datacentred.co.uk/blog/introducing-openstack-browser-termin
al/">datacenter.co.uk</a> &#8212; have The OpenStack Browser Terminal that's
created using jQuery Terminal.</li>
            <li><a href="http://lib.haxe.org/p/dconsole/">Haxe
Interpreter</a> &#8212; The Cross-platform Toolkit</li>
            <li><a
href="https://www.mimuw.edu.pl/~szynwelski/nlambda/console/">intereter for
nlambda</a> &#8212; Functional Programming over Infinite Structures.</li>
            <li><a
href="http://www.crashub.org/1.3/reference.html">CRaSH</a> &#8212; web
interface for The Common Reusable SHell (CRaSH).</li>
            <li><a
href="https://github.com/mattlo/angular-terminal">angular-terminal</a>
&#8212; A port of jQuery.terminal into AngularJS.</li>
            <li><a href="https://insect.sh/">insect</a> &#8212; a fast,
```

```
repl-style scientific calculator.</li>
            <li><a
href="https://packagist.org/packages/recca0120/laravel-tracy">laravel-tracy<
/a> &amp;#8212; A Laravel Package to integrate Nette Tracy Debugger. With <a
href="https://cdn.rawgit.com/recca0120/laravel-tracy/master/docs/tracy-excep
tion.html">demo</a>. It use <a
href="https://packagist.org/packages/recca0120/terminal">laravel-terminal</a
> build with jQuery Terminal.</li>
            <li><a href="http://algebrite.org/">Algebrite</a> &amp;#8212;
Computer Algebra System in Javascript use jQuery Terminal on <a
href="http://algebrite.org/sandboxes/latest-stable/sandbox.html">sanbox
page</a>.</li>
          </ul></li>
        <li>Home Pages
          <ul><li><a href="http://dhruvbird.com/">Dhruv Matani</a> &amp;#8212;
use tilda for navigation.</li>
            <li><strike><a href="http://kidsoftheapocalypse.org/">Kids of
the Apocalypse</a> &amp;#8212; use of overlay on top of terminal that give
vintage look.</strike></li>
            <li><a href="http://huy.im/">Huy Doan</a> &amp;#8212; black/green
fullscreen.</li>
            <li><stike><a href="http://awaxman.com/">Adam Waxman</a> &amp;#8212;
part of the site, stylized window, custom style.</stike></li>
            <li><a href="http://adva.io/">Nicol&amp;#242; Paternoster</a>
&amp;#8212; black/green fullscreen.</li>
            <li><a href="http://butchewing.com/">Butch Ewing</a> &amp;#8212;
black/grey fullscreen.</li>
            <li><a href="http://jesperdahlback.com/">jesperdahlback.com</a>
&amp;#8212; full screen with ASCII art.</li>
            <li><stike><a
href="http://projects.stashcat.me/">projects.stashcat.me</a> &amp;#8212;
commodore 64 themed home page.</stike></li>
            <li><stike><a href="http://www.ohmycode.fr/">ohmycode.fr</a>
&amp;#8212; fullscreen with colors. Try command <strong>team</strong> that show
ASCII art for each author.</stike></li>
            <li><stike><a href="http://vermillion.ws/">vermillion.ws</a>
&amp;#8212; fullscreen terminal.</stike></li>
            <li><strike><a
href="http://www.madhuakula.com/">madhuakula.com</a> &amp;#8212; fullscreen
green text, fake filesystem using GitHub API (cd,ls,cat) as
resume.</strike></li>
            <li><a href="https://github.com/bbody/CMD-Resume">CMD-Resume</a>
&amp;#8212; Resume build with terminal.</li>
            <li><a href="http://www.hacklover.net/">hacklover.net</a>
&amp;#8212; use terminal inside draggable window.</li>
            <li><a href="http://www.ronniepyne.com/">ronniepyne.com</a>
&amp;#8212; full sreen terminal.</li>
            <li><strike><a
href="http://kunhernowoputra.com/">kunhernowoputra.com</a> &amp;#8212; full
screen terminal.</strike></li>
            <li><strike><a href="http://keon.io/">keon.io</a> &amp;#8212; full
```

```
screen terminal.</strike></li>
            <li><a href="http://robertqualls.com/">Robert Qualls</a> &#8212;
terminal that stick in the header of the page.</li>
            <li><strike><a href="http://nbau21.github.io/">Noel Bautista</a>
&#8212; full screen terminal with colors.</strike></li>
            <li><a
href="http://www.masraniglobal.com/terminal/system/desktop.html">masraniglob
al</a> &#8212; Jurassic world themed terminal in dialog box.</li>
            <li><strike><a href="http://iprometheus.co.uk/">David Sekula</a>
&#8212; full screen with ascii art.</strike></li>
            <li><a href="http://www.talhahavadar.com/">Talha Havadar</a>
&#8212; full screen teerminal.</li>
            <li><a href="http://demlinks.com/">demlinks.com</a> &#8212;
terminal in a popup.</li>
            <li><strike><a href="http://sumyblog.me/">sumyblog.me</a>
&#8212; terminal in transparent dialog (link on bottom right
corner).</strike></li>
            <li><strike><a href="http://kabla.me/">kabla.me</a> &#8212; full
screen terminal with animated greetings.</strike></li>
            <li><a href="http://www.roqueterrani.com/">roqueterrani.com</a>
&#8212; full screen terminal.</li>
            <li><a
href="http://chebotkines.pythonanywhere.com/">chebotkines.pythonanywhere.com
</a> &#8212; full screen blue terminal with audio playback.</li>
            <li><a
href="http://philipyoo.github.io/">philipyoo.github.io</a> &#8212; full
screen terminal which echo html.</li>
            <li><a href="http://huntergregal.com/">huntergregal.com</a>
&#8212; Hunter Gregal's personal website, full screen.</li>
            <li><a href="http://www.pigeonlabs.com/">pigeonlabs.com</a>
&#8212; Nicola Ridolfi, full screen terminal with colors.</li>
          </ul></li>
        <li>Unusual use of terminal
          <ul><li><a href="https://duckduckgo.com/tty/">Duck Duck Go</a>
&#8212; use terminal as search interface.</li>
            <li><a href="http://jasonb.io/redditshell/">redditshell</a>
&#8212; Reddit shell. (<a
href="https://github.com/jasonbio/reddit-shell">repo</a>)</li>
            <li><a href="http://thedirectorsbureau.com">Directors Bureau</a>
&#8212; interface of this portfolio like site is exhanded by terminal</li>
            <li><a href="http://color64.com/">color64.com</a> &#8212;
Color64 BBS Homepage.</li>
            <li><strike><a href="http://m26.node-42.rv4a3.org/">ArmA 3 <abbr
title="Alternate Reality Game">ARG</abbr></a> &#8212; more info about it <a
href="http://www.gamebreaker.tv/pc-games/new-arma-3-arg/">here</a>.</strike>
</li>
            <li><a href="http://wedding.jai.im/">wedding.jai.im</a> &#8212;
use terminal to make OSX like terminal as invitation for a wedding.</li>
            <li><a href="http://premjith.in/">premjith.in</a> &#8212;
Another wedding invitation using Ubuntu command line.</li>
```

```
            <li><a
href="http://projectaon.org/staff/christian/gamebook.js/">gamebook.js</a>
&amp;#8212; an <a
href="http://en.wikipedia.org/wiki/Interactive_fiction">IF</a>-style
gamebook engine create by <a href="http://christianjauv.in/">Christian
Jauvin</a>.</li>
            <li><a href="http://heiswayi.github.io/w4y1/">w4y1</a> &amp;#8212;
An AI program created by Heiswayi Nrird as a fragment of his memories.</li>
          </ul></li>
        <li>Inside biger chunk of code
          <ul><li><a
href="http://code.google.com/p/os2online/">os2online</a> &amp;#8212; Web based
simulation of OS/2 Warp 3.0 operating system use jquery terminal.</li>
            <li><a
href="https://code.google.com/p/orongocms/">OrongoCMS</a>.</li>
            <li><strike><a
href="http://realhub.org/dev/apps/default/?node=central">WISDM</a> &amp;#8212;
Web-Interactive Scientific Data Manager.</strike></li>
            <li><a
href="http://alessandrorosa.altervista.org/complex/circles/">Circles</a>
&amp;#8212; ploting app for <a
href="https://en.wikipedia.org/wiki/Kleinian_groups">Kleinian groups</a>
&amp;#8212; it have terminal as a tool.</li>
            <li><a
href="https://worksheets.codalab.org/worksheets">codalab</a> &amp;#8212; use
terminal on worksheet page.</li>
            <li><a href="http://www.scripthica.com/">scripthica</a> &amp;#8212;
generating music with code (<a
href="https://gabrielsanchez.gitbooks.io/an-introduction-to-collective-algor
ithmic-music-c/">tutorial how to use it</a>)</li>
          </ul></li>
      </ul></article></section><!-- Piwik --><noscript><p><img
src="http://piwik.jcubic.pl/piwik.php?idsite=1" alt=""/></p></noscript>
  <!-- End Piwik Code -->
    </body>
```

</html>