Customizing PowerSchool web pages

Brent Johnson

Introduction

How custom pages work

Folder structure

- Custom web_root: <PS install folder>/data/custom/web_root
- Production web_root: <install folder>/system/server/resources/web_root

Enabling customizations on your server can be done by changing the setting in the admin portal at System → System Settings → Customization.

The synchronize custom web_root link does not need to be used to used in order to enable customizations. All this link will do is copy an empty directory structure into the custom web_root that mirrors the structure of the production web_root. It is easier to quickly figure out where custom pages actually exist in the custom web_root by not using the empty folder sync.

Special Page extensions

.htmlr	Render the page in the browser but do not process special PowerSchool tags.
.htmlt	Dump page's source code to the browser window.
.htmle	IE browser will prompt to download current page's source code.
?ac=structure	Show PowerSchool table/field structure with table/field numbers.

Code Insertion Tags

Wildcards

Wildcard files are simply .txt files with snippets of html in them. They are used in PowerSchool as a way to generate navigation menus and other commonly used portions of the web pages.

Tag Structure:

```
~[wc:{pagename}]
```

{pagename}
Required. Name of the wildcard file to insert. Do not include the file extension.

The contents of the text file referenced by the *{pagename}* parameter is inserted into the web page. Any special PowerSchool commands in the file are processed first and then the results are inserted. The file must be located in the web_root/wildcards folder. If a wildcard in the custom web_root/wildcards folder is changed, the enable customizations command must be rerun for PowerSchool to load the changes.

Insert File

Tag Structure:

```
~[x:insertfile;{pageurl}]
```

{pageurl}
Required. Url of page to insert relative to the current page.

This tag works similar to wildcard tags. Changes to files referenced by this tag do not need customizations re-enabled for the system to load the changes. Pages referenced in this tag can also exist anywhere in the server's web_root. Inserting files in this manner seems to cause longer load times when compared to wildcards.
The reason to use this tag instead of the wildcard tag is because pages inserted in this manner do not count against PowerSchool's 32k file size limit when rendering pages.

Modify Records

Table number

You must know the special number for the table that contains the records you want to modify. For example to modify a student record you will need the number for the students table which is 001. To view tables and their numbers you can add *?ac=structure* to the end of any url on your PowerSchool server.

DCID

The dcid is a unique value used in PowerSchool tables to identify records. Combining the table number and dcid gives PowerSchool the context for what record you are trying to view or edit. This leads us to what is called the frn...

FRN

The frn can be defined as follows: *<tablenumber><dcid>* Passing this value as a url parameter in PowerSchool will define the record to be used for tags on the page. If the dcid portion of the frn is set to -99 then PowerSchool will prepare the web form to create a new record in the table when the form is submitted.

Using tags on a page that has a frn defined...
~(rn) Returns the record number(dcid) from the current frn.

PowerSchool Data

Tag Structure:

```
~([{tablenumber}]{fieldname})
```

Examples:

```
~([01]grade_level)
~(grade_level)
~([03]course_name)
```

[{tablenumber}]
Optional. Number of the table to pull data from. If this is omitted, then PowerSchool will assume the table number from the frn on the current page. The ?ac=structure request can help find table numbers.

{fieldname}
Name of field to pull data from.

When pulling data in this manner, the page needs to be set to a record as a base point for pulling data. This is accomplished by passing a frn number to the page in the url parameters.

Creating Web Forms

Pass a FRN number to the page. This tells PowerSchool what table record should be used for the web form. Field names should be placed in the name attribute of the <input> tag. Checkbox types must have the value attribute set to 1. Text type inputs must have the value attribute set to an empty string
Examples:

```
<input name="[01]fieldname1" type="text" value="" />
<input name="[01]fieldname2" type="checkbox" value="1" />
```

Forms can be submitted to any valid page in PowerSchool. Records will not be saved unless a hidden input value is included in the web form...
Admin Portal: <input name="ac" value="prim" type="hidden" />
PowerTeacher Portal: <input name="ac" value="webasmt" type="hidden" />

Page Variables

GPV Tag
Get parameter variable. This tag will extract the value of a parameter that has been passed to the page through a GET or POST request.
Tag Structure:

```
~[gpv:{varname}] or
~(f.gpv;{varname}) or
~(gpv.{varname})
```

User Variables
guv/suv system:
Set variable

```
~(f.suv;name={varname};value={value})
```

Get variable

```
~(f.guv;name={varname})
```

{varname}
Required. Unique variable name to use on the page. If the variable has not been defined on the page, it will be created automatically

{value}
Required. Value to set into the variable.

f.variable system (also works in object reports):
Create variable

```
~(f.variable_create;name={varname};type={type};when=NEVER;)
```

Set variable value

```
~(f.variable_set;name={varname};value={value})
```

Get variable value

```
~(v.{varname})
```

{varname}
Required. Unique variable name to use on the page.

{type}
Required. Known values: TEXT, INT, DATE

{value}
Required. Value to set into the variable.

Preference Tags

~[displaypref:{prefname}]
~[dipslayprefyearschool:{prefname}]
~([pref]pref_name) Get Prefs value where Name = pref_name.
~([prefs]pref_name) Same as above.
~([prefschool]pref_name) Get Prefs value where Name = pref_name-S<curschoolID>.
~([prefuser]pref_name) Get Prefs value where Name = pref_name and UserID = current user.
~([prefyearschool]pref_name) Get Prefs value where Name = pref_name, schoolID = curschoolID and YearID = curyeardID.
~([prefyear]pref_name) Get Prefs value where Name = pref_name and YearID = curyeardID.
~([prefyearuser]pref_name) Get Prefs value where Name = pref_name, YearID = curyeardID and UserID = current user.

Conditional Logic Tags

If Tag

Tag Structure:

```
~[if{#identifier}.{expression}]
{true_render}
[else]
{false_render}
[/if]
```

{#identifier}
Optional. PS6 only. This portion of the tag is used when nesting multiple IF tags. The identifier must be prefixed with a number sign(#).

{expression}
Required. A comparison expression to test for a true/false result.

{true_render}
Optional. Rendered by PowerSchool if the if statement expression returns true.

{false_render}
Optional. Rendered by PowerSchool if the if statement expression returns false.

Example:

```
~[if#rntest.~(rn)=-99]
New
~[if#VarCheck.~[gpv:isShiny]=1]
 Shiny Item.
[else#VarCheck]
 Dull Item.
[/if#VarCheck]
[else#rntest]
Edit
[/if#rntest]
```

There are also special if statements available:
~[if.district.office]
~[if.is.a.school]
Checks if current school is set to District Office or not.

~[if.[displaypref;*prefname*]=*expression*]
~[if.pref.*prefname=expression*]
~[if.prefschool.*prefname=expression*]
Evaluates preferences in PowerSchool.

~[if.database.sql]
True if using PowerSchool Premier, false if using Pro. Useful for determining whether or not to use the ~[tlist_sql] tag.

~[if.CNFG.*serverconfigname=expression*]
Evaluate server configuration variables

~[if.mac]
~[if.win]
Checks client's workstation OS.

~[if.server.mac]
~[if.server.macx]
~[if.server.win]
Checks server's OS.

~[if.stateAbbr.*abbr*]
Check server's state abbreviation.

~[if.isstudent]
~[if.isguardian]
Check if student or guardian is logged into the parent portal in PowerSchool.

Case Tag
Tag Structure:

```
~[case.{variable}]
[of.{expression1}]...
[/of]
[of.{expression_n}]...
[/of]
...
[/case]
```

{variable}
Required. Item that will be tested for each case expression.

{expression1}
Required. Value to compare against {variable}. If the expression evaluates to true, then the text enclosed by the [of] tag will be rendered.

{expression_n}
Optional. Additional expressions can be used to check against {variable}.

SQL Tags

Tlist_SQL Tag

This command allows you to create your own sql queries and placing the results in a PowerSchool web page.
Tag Structure:

```
~[tlist_sql;{query};alternatecolor;nonemessage={none_message}]
 {row_template}
[/tlist_sql]
```

{query}
Required. Any valid Oracle sql SELECT statement. The query can contain any whitespace or new line characters and it will not break the tag.

{row_template}
Required. Replicated for each result row from the sql query. This template includes column substitution tags.

alternatecolor
Optional. This can be used to have PowerSchool render an alternating color background for the template.

{none_message}
Optional. PowerSchool will render {none_message} instead of the {row_template} when the query does not return any rows, or when the query fails.

Tlist_SQL column substitutes

Column substitutes will pull data from the result set in the same order that it is declared in the query.
Tag Structure:

```
~({name};{type};{modifier};)
```

Examples:

```
~(rel;t;if.blank.then=;else=~(rel;t)<br />;)
~(fname;t)
~(intval;l;if.test>10;then=High;else=Low;)
```

{name}
Unique name for the column binding. If this name is used again in the row template, it will pull the column data and not create a new binding.

{type}
Optional. Default = t. Possible Values: t = text, l = longint, d = date, r = real(double precision).

{modifier}
Optional. Conditional statements can be used here to replace the column's actual value.
{modifier} structure:

```
if.blank.then=blank_substitue;else=else_substitute
if.test=testvalue;then=then_substitute;else=else_substitute
```

Misc Tags

Duplicate Tag

Tag Structure:

```
~[duplicate;{n};{placeholdertext};
{substitute};counterstart={countstart};alternatecolor;]
{template}
[/duplicate]
```

Example:

```
~[duplicate;10;xxReplacexx;count;counterstart=5;alternatecolor;]
<tr bgcolor="#EDF3FE">
<td>Quantity: xxReplacexx</td>
<td>Price: $xxReplacexx</td>
</tr>
[/duplicate]
```

{n}
Required. Number of times to repeat duplicated text template.

{placeholdertext}
Optional. Text that will be replaced with a substitute value in the template.

{substitute}
Optional. Possible values: count.
Inserted any location in the template that matches the {placeholdertext} parameter.

counterstart={counterstart}
Optional. Default=1. Changes the starting value of the substitute count parameter.

alternatecolor
Optional. This can be used to have PowerSchool render an alternating color background for the template.

{template}
Required. Text that will be repeated by the tag.

Math Tags
Perform basic math operations on multiple values at once. If a date value is used as the first value, then any following integer values will be treated as a number of days.
Tag Structure:

```
~(f.add;{var1};{var2};...{varn})
~(f.sub;{var1};{var2};...{varn})
~(f.div;{var1};{var2};...{varn})
~(f.mult;{var1};{var2};...{varn})
```

Web Address Tags

`~[directory]`
Determines what portal of PowerSchool the page is on and returns: admin, teachers, or guardian.

`~[self]`
Returns url of the page this tag is on, but excludes http://<servername>/.

`~[referer]`
Returns the full url of the referring web page. If the page containing this tag is browsed directly, it will default to http://<servername>/admin/changesrecorded.white.html.

`~[referer.short]`
Same as ~[referer] tag, but without the http://<servername>/

Style Tags

`~[evenoddrow]`
Using this tag in the class attribute for a html tag will alternate between setting *evenRow* and *oddRow* as a class.
`~[evenoddrow;reset]`
Resets internal even/odd counter on the page back to 1 and then does the base tag operation.
`~[evenoddrow;maintain]`
Applies the same result as the previous ~[evenoddrow] tag.

Version History

v1.0 (11/11/2009)

- Initial Documentation release.