Wiki ページのメンテナンス

第 1.0 版 – 2017/3/6

Wiki “Springhead 開発者向け情報” のページの ‘開発版–今朝のビルド’ の項目には、毎日実行される dailybuild の最新結果を反映させるための php スクリプト呼出しが埋め込まれている。この文書では、これらのスクリプトについて記述する。

参考：“Springhead 開発者向け情報” の URL
`http://springhead.info/wiki/index.php?devel`

目次

1 スクリプトの埋め込み

“開発者向け情報のページ” には、次のような php スクリプト呼出しが埋め込まれている。

1.1 テーブル (今朝のビルドの状況、Samples のビルドの状況) の作成

引数で指定されたファイルを読んで html テーブルタグを作成する。
呼出し形式：

```
#dailybuild_result(
        /export/home/WWW/docroots/springhead/daily_build,
        Test.report,
        result.log, 0, 4)
#dailybuild_result(
        /export/home/WWW/docroots/springhead/daily_build,
        Test.report,
        result.log, 4, 2)
```

1.2 レビジョン番号および日付の埋め込み (レビジョン ··· との比較)

引数で指定されたファイルを読み、比較の対象となったレビジョンとその日付を抽出してインラインで返す。
呼出し形式：

```
&dailybuild_result(
    /export/home/WWW/docroots/springhead/daily_build,
    Test.report);
```

1.3 今朝のビルドの状況 (過去のビルドの履歴)

開発者向け情報のページで「今朝のビルドの状況（過去のビルドの履歴)」のリンクを辿ると、

1. ページ “http://springhead.info/wiki/index.php?cmd=edit&page=**今朝のビルドの履歴**” に遷移する。
2. そこから次の形式でスクリプト build_history が呼び出される。

```
#build_history(http://springhead.info/wiki/index.php?
    cmd=build_history&
```

```
base=/export/home/WWW/docroots/springhead/daily_build&
file=History.log&
type=1&
span=1&
unit=month&
sort=revision);
```

※ プラグイン呼出しの引数は urlencode されていること。

※ クエリ引数の意味

引数	値	説　明
cmd	`build_history`	実行するスクリプト名
base	†1	`$file` で指定されるファイルのあるディレクトリ
file	`History.log`	Springhead2 の更新履歴ファイル名 †2
type	1	'1' なら今朝のビルド、'2' なら Samples のビルド
span	1	履歴の表示範囲
unit	'month'	履歴の表示単位
sort	'revision'	ソートキー

†1 `/export/home/WWW/docroots/springhaed/daily_build`

†2 svn log コマンドで出力したもの

3. 引数で指定された「過去のビルドの履歴」ページ (html) が作成され、そこに redirect する。

注意

このページは、呼び出されると直ちにプラグインを呼び出して遷移を引き起こすため、そのままでは編集することができない。このページを編集するための手順は次のとおり。

1. スクリプト `"build_history.inc.php"` の先頭にある関数
   `build_history_get_options()`
   の static array member '`delayed_jump`' の値を `1` とする。
2. この状態でリンクを辿ればこのページが 5 秒間表示されるので、その間に Wiki の「編集」メニューを押す。
3. 編集が終ったら 1. で再設定した値を元の値 `0` に戻すこと。

1.4 Samples のビルドの状況 (過去のビルドの履歴)

次の点を除いて 1.3 今朝のビルドの状況 (過去のビルドの履歴) と同じ。

リンクを辿ったときの遷移先：

`http://springhead.info/wiki/index.php?Samples` **のビルドの履歴**

スクリプト呼出しのクエリ引数：

`type=1` → `type=2`

1.5 スクリプトのソース

`"/home/WWW/docroots/springhead/wiki/plugin/dailybuild_result.inc.php"`

(2 dailybuild_result.inc.php 参照)

`"/home/WWW/docroots/springhead/wiki/plugin/build_history.inc.php"`

(3 build_history.inc.php 参照)

2 dailybuild_result.inc.php

2.1 関数構成

テーブルの作成

```
#dailybuild_result() in Wiki page
    └ plugin_dailybuild_result_convert()
        ├ dailybuild_make_array()
        └ dailybuild_make_table()
```

比較対象レビジョン・日付取出し

```
&dailybuild_result() in Wiki page
    └ plugin_dailybuild_result_inline()
```

2.2 関数の説明

2.2.1 plugin_dailybuild_result_convert()

Wiki から `#dailybuild_result()` で呼び出されたときのエントリーポイント。

処理

1. 引数の取り出し
   - `$base` 以下の引数で指定されるファイルのあるディレクトリ
   - `$file1` MakeReport が作成したレポートファイル名
   - `$file2` Dailybuild が作成した結果ファイル名
   - `$start` データ抽出開始フィールド位置 (‘tests’ なら 0, ‘Samples’ なら 4)
   - `$count` 抽出データ数 (‘tests’ なら 4, ‘Samples’ なら 2)
2. 現在の日付の取り出し

   `$file1` の 3 行目の第 4 フィールドから切り出す。
3. 結果ファイルの解析

$content　$file2 の全体

$lines　$content を ‘)’ で分解した配列

$lines[0]　tests のビルド成功モジュール名を含む

$lines[1]　tests のビルド失敗モジュール名を含む

$lines[2]　tests の実行成功モジュール名を含む

$lines[3]　tests の実行失敗モジュール名を含む

$lines[4]　Samples のビルド成功モジュール名を含む

$lines[5]　Samples のビルド失敗モジュール名を含む

$lines[6]　Samples の実行成功モジュール名を含む

$lines[7]　Samples の実行失敗モジュール名を含む

$lines の添字 $start から $count について、$result 連想配列にモジュール名を取り出す。

キー　$proc と $code のペア

$proc は "B" 又は "R"　"B" は Build, "R" は Run を表す

$code は "S" 又は "F"　"S" は Success, "F" は Failure を表す

値　モジュール名のカンマで区切りリスト

$lines[n] の ‘)’ より後ろの部分 ("Lib:Mod[,Lib:Mod]...")

4. dailybuild_result_make_array() を呼んで $result の内容を Lib 毎にまとめる。
5. テーブルの html コードを生成する (<table>···</table>)
   固定部分以外は dailybuild_make_table() を呼び出す。

2.2.2 dailybuild_result_make_table($items, $fcolor, $bcolor, $font)

引数

$items　テーブルコンテンツの配列

$fcolor　前面色 (テキスト色)

$bcolor　背面色

$font　テキストフォントスタイル

処理

1. $items の各要素は "Lib:Mod1,Mod2,..." という形式になっている。これを、

Lib:	
	Mod1,Mod2,...

というテーブルのコードにする。罫線は書かない。

2.2.3 dailybuild_result_make_array($data)

引数

`$data`　モジュール名 (`"Lib:Mod"`) のカンマ区切りリスト

処理

1. `$data` で与えられたリストを Lib 毎にまとめて次のような配列 `$o_ary` にする。

```
data = L1:M11,L1:M12,...,L1:M1m,...,Ln:Mn1,Ln:Mn2,...,Ln:Mnl
   ↓
$o_ary[0] = L1:M11,M12,...,M1m
   ⋮
$o_ary[n] = Ln:Mn1,Mn2,...,Mnl
```

2.2.4 plugin_dailybuild_result_inline()

Wiki から `&dailybuild_result()` で呼び出されたときのエントリーポイント。

処理

1. 引数の取り出し
   `$base`　以下の引数で指定されるファイルのあるディレクトリ
   `$file`　MakeReport が作成したレポートファイル名
2. 比較対象のレビジョン番号と日付の取り出し
   `$file` の 2 行目の第 3 フィールドからレビジョン番号を、第 4 フィールドから日付を切り出す。

3 build_history.inc.php

3.1 関数構成

過去のビルドの履歴ページの内容の作成

#build_history(http GET method) in Wiki page

- plugin_build_history_action()
  - build_history_get_options()
  - build_history_make_link()
    - build_history_make_uri()
      - build_history_get_options()
    - build_history_display_unit()
  - build_history_make_link2()
    - build_history_make_uri()
      - build_history_get_options()
  - build_history_make_table()
    - build_history_display_unit()
    - build_history_get_head_info()
      - build_history_get_repository_url()
    - build_history_get_one()
      - build_history_get_repository_url()
    - build_history_make_array()
    - build_history_edit_one()
    - build_history_make_inner_table()

過去のビルドの履歴ページ (実体作成へのリダイレクトコード) の作成

[[**過去のビルドの履歴**>{ **今朝**|Samples} **のビルドの履歴**]] in Wiki page

- plugin_build_history_convert()
  - build_history_get_options()
  - build_history_urlencode()
    - build_history_urlencode_1()
      - build_history_urlencode_1() // recurse
  - build_history_query_to_array()

3.2 関数の説明

3.2.1 build_history_get_options()

build_history で共通に使用するパラメータの初期値を決める。

`cookie_usage`:

`plugin_build_action()` にパラメータを cookie で渡すなら `1` とする。`0` ならばパラメータは query で渡されるものとする。ただし `"base"`, `"file"`, `"referer"` 以外のパラメータは動的に設定するため、常に query で渡されるものとする。

`delayed_jump`:

「今朝のビルドの履歴」及び「Samples のビルドの履歴」のページを表示するためのパラメータ。`1` を設定るすとこれらのページを `$msec` ミリ秒だけ表示する。`$msec` の設定値は 5000 (5 秒)。(3.2.14 `plugin_build_history_convert()` 参照)

3.2.2 plugin_build_history_action()

Wiki からこの plugin が `GET` method で呼び出されたときに呼び出されたときのエントリーポイント。ページとして表示する html を生成する。

処理

1. 共通パラメータ `cookie_usage` の値に従って cookie 又は query からパラメータを取り出す。

`$base`	`$file` があるディレクトリ名
`$file`	履歴ファイル名 (`"History.log"`)
`$type`	ページ種別 (1: 今朝のビルド、2: Samples のビルド)
`$span`	表示範囲
`$unit`	表示範囲の単位 (`"month"`, `"week"` or `"day"`)
`$sort`	表示のソート順序 (`"succ"` or `"revision"`)
`$referer`	呼出し元 URL (サーバが設定する)

2. ページタイトル `$pagetitle` の設定
3. ページ本体 `$text` の初期化 (style id には `"content_1_0"` を使用)
4. 表示切り替え指定リンク `$links` の作成 (table tag)

   表示範囲指定　`build_history_make_link()` で作成する。

   表示順序指定　`build_history_make_link2()` で作成する。
5. 履歴テーブル `$table` の作成 (`<table>···</table>`)

   `build_history_make_table()` で作成する。

6. 関連ページリンク $trailer の作成

7. $msg にページタイトルを、 $body に上記 2. から 6. までの連接を設定して返す。

3.2.3 build_history_make_uri($base, $file, $type, $span, $unit, $sort, $referer)

引数

3.2.2 plugin_build_history_action() の 1. 項参照

処理

1. 与えられた引数に従って URI を作成する (GET method)。cookie_usage の値により query が異なる。

3.2.4 build_history_make_link($base, $file, $type, $span, $unit, $sort, $referer, $curr_span)

引数

3.2.2 plugin_build_history_action() の 1. 項参照

$curr_span　現在表示中の表示範囲

処理

1. 与えられた引数に従って表示範囲指定リンクを作成する (···)

3.2.5 build_history_make_link2($base, $file, $type, $span, $unit, $sort, $referer, $curr_sort)

引数

3.2.2 plugin_build_history_action() の 1. 項参照

$curr_sort　現在表示中の表示順序

処理

1. 与えられた引数に従って表示順序指定リンクを作成する (···)

3.2.6 build_history_display_unit($unit)

引数

$unit　表示範囲の単位 ("month", "week" or "day")

処理

1. 表示範囲の単位を表す文字列を返す。引数が予期せぬ値のときは "month" が指定されたものとして扱う。

3.2.7 build_history_make_table($base, $file, $type, $span, $unit, $sort)

引数

3.2.2 plugin_build_history_action() の 1. 項参照

処理

1. 表示範囲の設定
   $epoch　レビジョン 6364 以前には Samples の記録がない
   $limit　$span で指定された範囲で一番古い日付
2. テーブル見出し $caption の作成
   ※ 最後の table 要素は行間をあけるためのもの
3. 履歴情報の取得
   $head　最新レビジョンの情報 (build_history_get_head_info())
   $histories　過去レビジョンの情報 (履歴ファイルを読む)
   $lines　$head と $histories をまとめたもの
4. 履歴情報の解析
   履歴情報は次のような構造の繰り返しと仮定している

   ↓↓ ここから ↓↓

   ```
   ------------------------------------------------------------------------
   r<nnnn> | <name> | yyyy-mm-dd hh:mm:ss +0900 ...
   (empty line)
   Autobuild done.
   ```

   ↑↑ ここまで ↑↑

   これらよりレビジョン番号 nnnn と日付 yyyy–mm–dd を取り出し、配列 $history にレビジョン番号をキーとして格納する。
5. 上で設定した $history の各 $revision, $date ペアについて、
   (a) $revision の "history.log" の読み出し
   $tmp ← build_history_get_one()
   $tmp のフォーマットは次のとおり

column	0〜3	レビジョン番号 (4 桁)
	4〜13	ビルド日付 (”yyyy-mm-dd”)
	14〜16	成功モジュール数
	17〜19	エラーモジュール数
	20〜	成功モジュール名

   (b) ソートキーの作成。ソートキーは、
   $sort が "revision" なら　$tmp の 0〜3 カラム
   $sort が "succ" なら　$tmp の 14〜16 + 0〜3 カラム

(c) ソートキーをキーとして $module に $tmp を格納

6. $module のソート (キーの逆順)
7. テーブルの見出し $table の作成
8. $module の各 ($key , $val) ペアについて、
   (a) $val の 20 カラム以降 (成功モジュール名) の編集

       $mod_names ← build_history_make_array()

   (b) $val と組み合わせて表示 1 行分の編集

       $table ← build_history_edit_one()

9. $caption と $table を連接して返す。

3.2.8 build_history_get_head_info()

処理

1. HEAD レビジョンの情報を log コマンドで取得して返す

3.2.9 build_history_get_one($revision, $date, $type)

引数

$revision	取得するデータのレビジョン番号
$date	取得するデータの日付
$type	ページ種別 (1: 今朝のビルド、2: Samples のビルド)

処理

1. $output ← 指定されたレビジョンの "result.log" の内容
2. ビルドの成功・失敗、実行の成功・失敗が記録されている行を設定する。

$b_succ_1	tests ビルド成功 †
$b_fail_1	tests ビルド失敗 †
$r_succ_1	tests 実行成功
$r_fail_1	tests 実行失敗
$b_succ_2	Samples ビルド成功 †
$b_fail_2	Samples ビルド失敗 †
$r_succ_2	Samples 実行成功
$r_fail_2	Samples 実行失敗

† レビジョン 7031 の前後で、result.log に記録されている「ビルド失敗」と「実行成功」の順序が入れ替わっている。

3. ビルドの成功モジュール数、失敗モジュール数を数える。tests を対象とするか Samples を対象とするかは引数 $type に従う。モジュール数とは、括弧の中のカンマで区切られた要素の数である。

4. 成功モジュール名リストを作成する。
   tests の場合　　実行成功モジュール名リストを作成
   Samples の場合　ビルド成功モジュール名リストを作成

5. 次の形式で結果を返す。
   column　0〜3　　レビジョン番号 (4 桁)
   4〜13　ビルド日付 (”yyyy-mm-dd”)
   14〜16　成功モジュール数
   17〜19　エラーモジュール数
   20〜　　成功モジュール名

3.2.10 build_history_get_repository_url()

処理

1. svn repository の URL(固定値) を返す。
   URL ←http://springhead.info/spr2/Springhead2/trunk/test

3.2.11 build_history_edit_one($data, $style)

処理

1. 3.2.12 で作成したデータを分解し、html のテーブル要素 1 行分を作成する。成功モジュール名の部分は build_history_make_inner_table() により、内部テーブル (<td><table> ···</table></td>) として作成する。

3.2.12 build_history_make_array($data)

引数

$data　モジュール名並び

処理

1. 引数で与えられたモジュール名並びの文字列 ($data) を分解し、lib 名毎にまとめた配列 ($o_ary) として返す。

```
$data   =  "Lib1:Mod11,Lib1:Mod12,...,LibN:ModN1,..."
  ↓
$o_ary  =  array("Lib1:Mod11,Mod12,...", ...,
                 "LibN:ModN1,ModN2,...", ...)
```

3.2.13 build_history_make_inner_table($data)

引数

`$data` 成功モジュール名リスト
`"Lib1:Mod11,Mod12,...;...;LibN:ModN1,ModN2,..."`
3.2.12 で作成した配列要素を ';' で結合した形式

処理

1. `$data` を ';' で分解した各要素は `"Lib1:Mod11,Mod12,..."` という形式になっている。
   これを、

Lib:	
	`Mod1,Mod2,...`

   というテーブルのコードにする。罫線は書かない。

3.2.14 plugin_build_history_convert()

Wiki から `#build_history()` で呼び出されたときのエントリーポイント。

処理

1. リダイレクトコード `$redirect_to` に次のものを設定する。
   (a) 共通パラメータ `cookie_usage` が `0` のとき：
   引数を取り出し (`func_get_args()`) 、referer を追加する。
   (b) 共通パラメータ `cookie_usage` が `1` のとき：
   引数を取り出し (`func_get_args()`) 、query を再構成する。
   次のキーについては、指定がなければデフォルト値を設定する。
   type → `1`
   span → `1`
   unit → `"month"`
   sort → `"succ"`
   キー `referer` を追加する。
   次のキーについては、 `cookie` に値を設定する。
   `"base"`, `"file"`, `"referer"`
2. 遅延ジャンプ実現のためのフック
   共通パラメータ `delayed_jump` が設定されていたならば、`$msec` に指定した時間だけ遅延するコードを生成する (JavaScript のタイマ機構を使用する)。

3. $redirect_to で指定したロケーションへジャンプする。
   これにより 3.2.2 plugin_build_history_action() が呼び出される。

3.2.15 build_history_urlencode($uri)

引数

$uri　変換する uri

処理

1. uri を scheme, path, query に分解する。
2. scheme 部分：
   htmlspecialchars()　特殊文字のサニタイズ
   urlencode()　url エンコード
3. path, query の部分も scheme 部分と同様であるが、各々セパレータ文字で区切られた各部分について処理をするため、build_history_urlencode_1() を呼び出す。
   path 部分　セパレータは '/'
   query 部分　主セパレータは '&'、副セパレータは '='

3.2.16 build_history_urlencode_1($str, $sep1, $sep2)

引数

$str　変換する文字列
$sep1　主セパレータ文字
$sep2　副セパレータ文字

処理

1. 変換する文字列を主セパレータ文字で分割した各々について、
   (a) 副セパレータが空ならば、htmlspecialchars() でサニタイズをした後、php の urlencode() で url エンコードする。
   (b) 副セパレータが空でなければ、それを主セパレータとして自分自身を再帰呼出しする (副セパレータは空に指定する)。
2. エンコードされた各部分を主セパレータ文字で結合して返す。

3.2.17 build_history_query_to_array($query)

引数

$query　クエリ文字列

処理

1. クエリ文字列は "key1=val1&...&keyN=valN" という形式をしている。これを、配列 $queries に次のように設定して返す。

```
$queries[key1] = val1
      ⋮
$queries[keyN] = valN
```