**Global Reporter Reference Guide**

**(ref v0.55.0)**

**program version : v0.55**

# 변경사항

V0.55.0 - /make\_pdf 기능 추가

v0.54.0 – coer page 추가

v0.53.0 - 파일 인코딩 표현 방법 수정.(:인코딩에서 [인코딩]으로).TestNgine 2에 적용. word로 레퍼런스 작성.(앞으로는 ms-word로 배포)

v0.52.0 - 테이블 동적 생성 기능 추가.

v0.51.0 - Data파일 만들기 예제 추가. Quick-Start추가

v0.50.0 - 베타버전 릴리즈. 기초수준 레퍼런스 배포

# 시스템 요구사양:

.Net framework 4.0

# 목차

[**0. Quick-Start**](#_Quick-Start)

[**1. Data파일 만들기**](#_Data파일_만들기_규칙.)

[1-1) #테이블=테이블제목1,테이블제목2,...](#_1-1)_#테이블=테이블제목1,테이블제목2,...)

[1-2) %TITLE:값,값,...,값](#_1-2)_%TITLE:값,값,...,값)

[1-3) 값,값,...,값](#_1-3)_값,값,...,값)

[1-4) %ROW(n):값,값,...,값](#_1-4)_%ROW(n):값,값,...,값)

[1-5) //주석](#_1-5)_//주석)

[1-6) >>추가문자열](#_1-6)_>>추가문자열)

[1-7) 속성 ,예약어 및 명령](#_1-7)_속성_,예약어)

1-7-1) @속성명

1-7-2) %예약어

[1-7-3) 예약어목록:](#_1-7-3)_예약어목록:)

[1-7-4) 예약 함수 목록](#_1-7-4)_예약_함수)

[1-8) 차트 추가](#_1-8)_차트_추가)

[1-9) 테이블 동적 추가 및 변경](#_1-9)_테이블_동적)

[1-9-1) 테이블 추가](#_1-9-1)_테이블_추가)

[1-9-2) 기존 테이블 참조하여 추가](#_1-9-2)_기존_테이블)

[1-9-3 ) 새로 만든 테이블 모양 변경](#_1-9-3_)_새로)

[1-9-3-1 ) 셀 합치기](#_1-9-3-1_)_셀)

[1-9-3-2 ) 셀 크기 조정](#_1-9-3-2_)_셀)

[1-9-3-3 ) 셀 스타일 조정](#_1-9-3-3_)_셀)

[**2. 디자인파일 만들기**](#_디자인파일_만들기)

[2-1) 공통규칙.](#_2-1)_공통규칙.)

[2-2) basic파일(Config/report\_form\_basic.html) 만들기 규칙](#_1-9-3-2_)_셀)

[2-3) 테이블파일(ReportForms/\*) 만들기 규칙](#_2-3)_테이블파일(ReportForms/*)_만들기)

[2-4) 레포트 표지(Covr page) 만들기 규칙](#_2-4)_표지문서_(Cover)

[**3. 설정 파일정의** :](#_설정_파일정의_:)

[3-1) Config/Properties.txt](#_3-1)_Config/Properties.txt)

[3-2) Config/Files\_to\_parse.txt](#_3-2)_Config/Files_to_parse.txt)

[3-3) Config/Path\_to\_save.txt](#_3-3)_Config/Path_to_save.txt)

[**4. 실행시 arguments**](#_실행시_arguments)

4-1) /current\_path=실행파일경로

4-2) /properties=속성명1:속성,속성명2:속성,…

4-2-1) /property\_file=속성정의파일경로[Encoding]

4-3) /src = 레포트데이터파일1.txt[Encoding],레포트데이터파일2.txt[Encoding],...

/files\_to\_parse=레포트데이터파일1.txt[Encoding],레포트데이터파일2.txt[Encoding],...

4-4) /dst=레포트 pdf파일 저장 경로.

4-5) /basic\_form=기본디자인폼 경로

4-6) /forms=테이블파일1.html,테이블파일2.html,...

4-7) /make\_pdf

4-8) /print\_now

# Quick-Start

[\\192.168.1.90\doc\ProjectsBackup\GlobalReporter](file:///\\192.168.1.90\doc\ProjectsBackup\GlobalReporter) 에 접근하여 가장 최신의 파일을 가져온다.

여기서는 GlobalReporter\_20140423\_오후0837버전의 GlobalReporter\_20140423\_오후0837\_exe\_v0.53.zip 를 사용한다.

이 파일을 자신의 어플리케이션 폴더에 풀어준다. 만일 어플리케이션의 경로가 [C:\Application1](file:///C:\Application1) 이라고 하면,

압축파일 내부에 있는 폴더가 [C:\Application1\GlobalReporter\_exe](file:///C:\Application1\GlobalReporter_exe) 가 되도록 풀어준다.

Global\_Reporter\_exe 폴더 내부에는

|  |
| --- |
| Config  data  externals  Lib  Pechkin  Report\_forms  Webkit  report |

폴더가 있다.

이 중, 알아야 할 폴더는 Config, data, reports 밖에 없다.

일단, Config폴더 내부에 FILES\_TO\_PARSE.txt를 보자.

이 파일은 레포트를 만들기 위해 처리해야 할 파일 경로를 나타내고 있다.

내용은 다음과 같다.

|  |
| --- |
| //data/FCC\_data.txt[MULTIBYTE] //data/result.txt[MULTIBYTE] data/sample.txt[MULTIBYTE] //%REMOVE\_THIS\_FILES |

여기서 앞에 //표시가 있는 것은 주석처리된 것이다.

활성화된 data/sample.txt [MULTIBYTE] 를 보자.

data폴더 내부에 sample.txt 를 MULTIBYTE로 인코딩하여 레포트를 만든다는 뜻이다.

이 파일은 아래의 1. Data파일 만들기 규칙에 따라 만들어졌다.

data폴더의 sample.txt를 보자.

내용은 대략적으로 다음과 같을 것이다.

|  |
| --- |
| #4CELL\_TABLE=제목1,제목2  %TITLE:열1,열2,열3,열4  값1,값2,값3,성공  값1,값2,값3,성공  값1,값2,값3,성공 |

만일 아니라면 위와 같이 만들고  [C:\Application1\GlobalReporter\_exe](file:///C:\Application1\GlobalReporter_exe)\WebkitReporter.exe 를 실행해 보자.

다음과 같이 나타날 것이다.



위 결과는 data\Properties.txt의 내용과 Config\Report\_form\_BASIC.html, 그리고 Report\_forms\report\_form1.html 에서 #4CELL\_TABLE 이 조합되어 나타난 것이다.

일단  Config\Report\_form\_BASIC.html 파일을 웹브라우저로 열어보자.



위의 레포트와 모양이 비슷하게 생겼다. 여기서 보면 [@REPORT\_TITLE:보고서명] 등의 [ ]로 둘러싸인 부분들이 치환되었음을 알 수 있다.

이제 data\Properties.txt의 내용을 보자.

NAME:

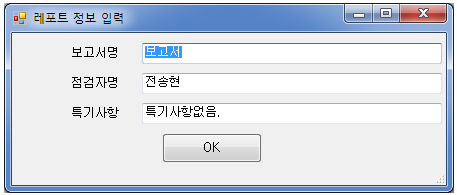
ID:n50126

TEST\_LOCATION:리얼타임웨이브

REPORT\_TITLE:

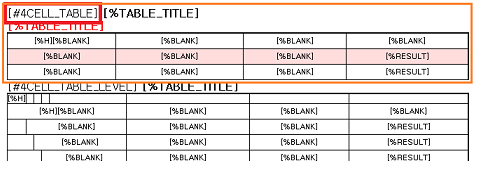
TEST\_CONTENT:시험적으로 레포트 뽑아봄.

그리고 실행하면 나타나는 팝업도 살펴보자.



이 두 가지 정보가 바로 report\_form\_BASIC.html 내부로 치환되었음을 알 수 있다.

마지막으로, [@CONTENT:800] 이라고 된 부분은 데이터 파일과 연관되는데, Report\_forms\report\_form1.html 을 브라우저로 열어보자.



data\sample.txt에 적어넣은 내용을 보면,

**#4CELL\_TABLE**=제목1,제목2

%TITLE:열1,열2,열3,열4

값1,값2,값3,성공

값1,값2,값3,성공

값1,값2,값3,성공

같은 이름을 찾아볼 수 있다. 이런 식으로 테이블을 선택하면 내용을 채울 수 있다.

사용자의 Application 내부에서  [C:\Application1\GlobalReporter\_exe](file:///C:\Application1\GlobalReporter_exe)\WebkitReporter.exe 를 실행하기 위해서는 argument로 다음과 같이 주어야 한다.

[C:\Application1\GlobalReporter\_exe](file:///C:\Application1\GlobalReporter_exe)\WebkitReporter.exe /current\_path=[C:\Application1\GlobalReporter\_exe](file:///C:\Application1\GlobalReporter_exe)

더 자세한 내용은 1. Data파일 만들기를 참조하라.

# Data파일 만들기 규칙.

## 1-1) #테이블=테이블제목1,테이블제목2,...

:미리 디자인해 놓은 테이블을 선택하고, 그 안에서 [%TABLE\_TITLE]의 내용들이 테이블제목으로 하나씩 치환된다.

## 1-2) %TITLE:값,값,...,값

: 테이블을 선택해 놓은 상황에서 제목줄에 들어갈 내용을 정의한다. 해당 제목줄은 테이블이 나뉘어지게 되면 반복되어 들어간다.

## 1-3) 값,값,...,값

:테이블을 선택해 놓은 상황에서 값을 치환한다.

예제>

[#4CELL\_TABLE] **[%TABLE\_TITLE]**  
**[%TABLE\_TITLE]**

|  |  |  |  |
| --- | --- | --- | --- |
| [%H][%BLANK] | [%BLANK] | [%BLANK] | [%BLANK] |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |

디자인이 위와 같을 때,

[#4CELL\_TABLE] 이 테이블의 이름이다. 이때 테이블은 바로 아래에 나타난 테이블을 가리킨다.

위에서 [%TABLE\_TITLE]이라고 된 부분은 테이블의 제목을 나타내는 것으로 디자인영역에 없어도 무방하다.

하지만 있다면 아래와 같이,

#4CELL\_TABLE=제목1,제목2

라고 써서 [%TABLE\_TITLE]부분을 각각 제목1 과 제목2로 치환할 수 있다.

결과는 다음과 같이 될 것이다.(테이블은 아직 나타나지 않음.)

**제목1**  
**제목2**

|  |  |  |  |
| --- | --- | --- | --- |
| [%H][%BLANK] | [%BLANK] | [%BLANK] | [%BLANK] |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |

다음으로 제목줄에 값을 채워넣어야 한다.

디자인 안에[%H]라고 되어 있는 줄은 제목줄로 취급된다.

제목줄이라 함은, 테이블이 화면을 넘어갔을 때, 반복되면서 그 열이 어떤 열인지 나타내 주는 HEADER라고 할 수 있다.

이 제목줄은 사용되지 않을 수도 있으므로, 아래와 같이 %TITLE키워드를 사용하여 일반 데이터와 구분한다.

([%H]태그가 없다면 %TITLE 키워드로 정의된 줄은 무시될 것이다. - 이는 디자인과 데이터의 독립성을 위한 것으로, TITLE을 보고 싶지 않을 때는 디자인에서 [%H]태그만 없애면 되도록 하기 위해서이다.)

다음과 같이 적으면 결과는 아래와 같이 된다.(데이터행은 아직 나타나지 않음)

#4CELL\_TABLE=제목1,제목2

%TITLE:열1,열2,열3,열4

**제목1**  
**제목2**

|  |  |  |  |
| --- | --- | --- | --- |
| 열1 | 열2 | 열3 | 열4 |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |

마지막으로 데이터를 채워넣을 것이다.

데이터는 그냥 ,로 구분하여 테이블에 들어갈 데이터를 넣어주기만 하면 된다.

열이 4개 이므로 데이터는 4개가 되어야 한다. 그 이상되면, 무시된다.

#4CELL\_TABLE=제목1,제목2

%TITLE:열1,열2,열3,열4

값1,값2,값3,값4

**제목1**  
**제목2**

|  |  |  |  |
| --- | --- | --- | --- |
| 열1 | 열2 | 열3 | 열4 |
| 값1 | 값2 | 값3 | **값4** |

데이터가 여러 줄이면 그냥 반복하면 자동으로 행이 생겨난다. 이 때, 생겨나는 행의 순서는 디자인에서의 데이터 행이 2줄이라면 번갈아서 나타나게 된다.

#4CELL\_TABLE=제목1,제목2

%TITLE:열1,열2,열3,열4

값1,값2,값3,값4

값1,값2,값3,값4

값1,값2,값3,값4

값1,값2,값3,값4

값1,값2,값3,값4

**제목1**  
**제목2**

|  |  |  |  |
| --- | --- | --- | --- |
| 열1 | 열2 | 열3 | 열4 |
| 값1 | 값2 | 값3 | **값4** |
| 값1 | 값2 | 값3 | **값4** |
| 값1 | 값2 | 값3 | **값4** |
| 값1 | 값2 | 값3 | **값4** |
| 값1 | 값2 | 값3 | **값4** |

그런데 자세히 보면 가장 우측 열의 내용이 빨갛게 되어있는 것을 볼 수 있다.

테이블 원형을 자세히 보면 다음과 같이 되어있다.

|  |  |  |  |
| --- | --- | --- | --- |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |

맨 우측 열에 [%RESULT]라고 되어있다.

이는 값의 PASS/FAIL을 판단하여 색깔로 나타내겠다는 뜻이다.

기본적으로 OK/성공/PASS 등의 키워드가 기본적으로 PASS로 설정되어있다.

값을 아래와 같이 바꾸고 다시 만들어보면 이렇게 보인다.

#4CELL\_TABLE=제목1,제목2

%TITLE:열1,열2,열3,열4

값1,값2,값3,OK

값1,값2,값3,성공

값1,값2,값3,PASS

값1,값2,값3,실패

값1,값2,값3,그외

**제목1**   
**제목2**

|  |  |  |  |
| --- | --- | --- | --- |
| 열1 | 열2 | 열3 | 열4 |
| 값1 | 값2 | 값3 | OK |
| 값1 | 값2 | 값3 | 성공 |
| 값1 | 값2 | 값3 | PASS |
| 값1 | 값2 | 값3 | 실패 |
| 값1 | 값2 | 값3 | 그외 |

만일, (그럴일은 없겠지만) '실패' 라는 문자열과 '그외'라는 문자열을 PASS로 인식하게 하고 싶다면, 다음과 같이 디자인원형을 바꾸면 된다.

|  |  |  |  |
| --- | --- | --- | --- |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT:실패,그외] |

결과는 아래와 같이 나온다.

**제목1**   
**제목2**

|  |  |  |  |
| --- | --- | --- | --- |
| 열1 | 열2 | 열3 | 열4 |
| 값1 | 값2 | 값3 | OK |
| 값1 | 값2 | 값3 | 성공 |
| 값1 | 값2 | 값3 | PASS |
| 값1 | 값2 | 값3 | 실패 |
| 값1 | 값2 | 값3 | 그외 |

## 1-4) %ROW(n):값,값,...,값

:테이블 디자인에서 TITLE줄 다음에 있는 내용줄의 순서(n)대로 선택하여 추가된다. n은 0부터이다.

(주의! title열이 2줄 이상이라도 내용줄은 한줄씩만 호출 가능하다.)

위에서 디자인의 데이터 행이  2개 이상일 때 반복되어 나타나는 것을 보았다. 그러나 상황에 따라서 각 줄을 선택해서 보여줘야 할 때도 있다.

다음과 같은 디자인이 있다고 하자.

[#4CELL\_TABLE\_LEVEL] **[%TABLE\_TITLE]**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| [%H] |  |  |  |  |  |  |  |
| [%H][%BLANK] | | | | | [%BLANK] | [%BLANK] | [%BLANK] |
|  | [%BLANK] | | | | [%BLANK] | [%BLANK] | [%RESULT] |
|  | | [%BLANK] | | | [%BLANK] | [%BLANK] | [%RESULT] |
|  | | | [%BLANK] | | [%BLANK] | [%BLANK] | [%RESULT] |
|  | | | | [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |

위 테이블의 이름은 #4CELL\_TABLE\_LEVEL 이고, [%TABLE\_TITLE]태그는 하나가 있다.

이 테이블을 선택하고 헤더영역에 TITLE을 넣고 데이터를 추가해 보자.

#4CELL\_TABLE\_LEVEL=제목1

%TITLE:열1,열2,열3,열4

값1,값2,값3,성공

값1,값2,값3,성공

값1,값2,값3,성공

값1,값2,값3,성공

값1,값2,값3,성공

값1,값2,값3,성공

값1,값2,값3,성공

값1,값2,값3,성공

**제목1** 

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 열1 | | | | | 열2 | 열3 | 열4 |
|  | 값1 | | | | 값2 | 값3 | 성공 |
|  | | 값1 | | | 값2 | 값3 | 성공 |
|  | | | 값1 | | 값2 | 값3 | 성공 |
|  | | | | 값1 | 값2 | 값3 | 성공 |
|  | 값1 | | | | 값2 | 값3 | 성공 |
|  | | 값1 | | | 값2 | 값3 | 성공 |
|  | | | 값1 | | 값2 | 값3 | 성공 |
|  | | | | 값1 | 값2 | 값3 | 성공 |

헤더영역이 2줄이 있으므로, 보이지는 않지만 헤더는 2줄이 나타난다. 맨 윗줄은 아래의 제일 좌측 CELL이 한단계씩 들어가 보이도록 디자인상의 테크닉을 발휘한 것이다.(하지만 내용이 하나도 없고서 높이가 0으로 셋팅되어 보이지 않는다.)

그런데 아랫쪽의 5줄의 디자인영역이 반복되어 원했던 모양이 아니다.

원했던 모양은, 1레벨이 1개, 2레벨이 2개, 3레벨이 3개, 4레벨이 2개가 차례대로 나타나는 것이라고 하자.

그러면 데이터 열 앞에 %ROW(행순서): 를 넣어서 행을 선택해 줄 수 있다.

행 순서는 0부터 시작한다.

#4CELL\_TABLE\_LEVEL=제목1

%TITLE:열1,열2,열3,열4

%ROW(0):값1,값2,값3,성공

%ROW(1):값1,값2,값3,성공

%ROW(1):값1,값2,값3,성공

%ROW(2):값1,값2,값3,성공

%ROW(2):값1,값2,값3,성공

%ROW(2):값1,값2,값3,성공

%ROW(3):값1,값2,값3,성공

%ROW(3):값1,값2,값3,성공

**제목1** 

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 열1 | | | | | 열2 | 열3 | 열4 |
|  | 값1 | | | | 값2 | 값3 | 성공 |
|  | | 값1 | | | 값2 | 값3 | 성공 |
|  | | 값1 | | | 값2 | 값3 | 성공 |
|  | | | 값1 | | 값2 | 값3 | 성공 |
|  | | | 값1 | | 값2 | 값3 | 성공 |
|  | | | 값1 | | 값2 | 값3 | 성공 |
|  | | | | 값1 | 값2 | 값3 | 성공 |
|  | | | | 값1 | 값2 | 값3 | 성공 |

원하는 결과가 나왔다.

## 1-5) //주석

:주석을 추가한다. 실제 레포트에서는 보이지 않는다.

주석을 사용할 일이 있을 지는 모르겠지만, 원하는 레포트를 수동으로 만든다고 할 때, 주석이 나오지 않으면 무슨 내용인지 알아보기가 힘들 것이다.

위의 1-4)의 마지막 예제의 데이터에 주석을 달면 아래와 같은 모습이 될 것이다.

#4CELL\_TABLE\_LEVEL=제목1

%TITLE:열1,열2,열3,열4

//LEVEL 1 1개

%ROW(0):값1,값2,값3,성공

//LEVEL 2 2개

%ROW(1):값1,값2,값3,성공

%ROW(1):값1,값2,값3,성공

//LEVEL 3 3개

%ROW(2):값1,값2,값3,성공

%ROW(2):값1,값2,값3,성공

%ROW(2):값1,값2,값3,성공

//LEVEL 4 2개

%ROW(3):값1,값2,값3,성공

%ROW(3):값1,값2,값3,성공

**제목1**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 열1 | | | | | 열2 | 열3 | 열4 |
|  | 값1 | | | | 값2 | 값3 | 성공 |
|  | | 값1 | | | 값2 | 값3 | 성공 |
|  | | 값1 | | | 값2 | 값3 | 성공 |
|  | | | 값1 | | 값2 | 값3 | 성공 |
|  | | | 값1 | | 값2 | 값3 | 성공 |
|  | | | 값1 | | 값2 | 값3 | 성공 |
|  | | | | 값1 | 값2 | 값3 | 성공 |
|  | | | | 값1 | 값2 | 값3 | 성공 |

주석을 달아도 결과에는 변함이 없다.

## 1-6) >>추가문자열

:일반 문자열을 추가한다.

추가 문자열은 테이블 하나만을 사용할 때는 필요가 없을지도 모른다. 하지만 여러 개의 테이블을 사용하는 레포트를 만들 때는 테이블 사이에 추가적인 정보를 표시해야 할 일이 있다.

#4CELL\_TABLE\_LEVEL=제목1

%TITLE:열1,열2,열3,열4

%ROW(0):값1,값2,값3,성공

**>>추가적인 정보사항입니다.**

#4CELL\_TABLE\_LEVEL=제목1

%TITLE:열1,열2,열3,열4

%ROW(0):값1,값2,값3,성공

위와 같이 두 개의 테이블 사이에 추가문자열을 넣어주면 아래와 같이 나타난다.

**제목1** 

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 열1 | | | | | 열2 | 열3 | 열4 |
|  | 값1 | | | | 값2 | 값3 | 성공 |

추가적인 정보사항입니다.  
**제목1**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 열1 | | | | | 열2 | 열3 | 열4 |
|  | 값1 | | | | 값2 | 값3 | 성공 |

HTML태그를 사용하여 스타일을 변경할 수도 있다.

#4CELL\_TABLE\_LEVEL=제목1

%TITLE:열1,열2,열3,열4

%ROW(0):값1,값2,값3,성공

//색깔을 빨간색으로 나타내 줌.

**>><span style="color:#ff0000;">추가적인 정보사항입니다.</span>**

#4CELL\_TABLE\_LEVEL=제목1

%TITLE:열1,열2,열3,열4

%ROW(0):값1,값2,값3,성공

**제목1** 

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 열1 | | | | | 열2 | 열3 | 열4 |
|  | 값1 | | | | 값2 | 값3 | 성공 |

추가적인 정보사항입니다.   
**제목1**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 열1 | | | | | 열2 | 열3 | 열4 |
|  | 값1 | | | | 값2 | 값3 | 성공 |

## 1-7) 속성 ,예약어 및 명령

1-7-1) @속성명 : 위 값에 해당되는 부분에 @속성명 을 넣으면 Properties.txt 안에 정의된 속성으로 치환된다.

1-7-2) %예약어 : 아래 예약어 목록에 정의된 예약어를 사용하여 임시변수나 명령을 수행한다.

값에 해당되는 부분에 @속성명 으로 써 주면, 미리 지정한 속성으로 치환할 수 있다. 이 속성을 지정하는 방법은 3가지가 있다.

1. 프로그램 실행시에 argument로 넘겨준다. ex>  /properties=속성명1:속성값1,속성명2:속성값2

2. Config/Properties.txt 내부에 정의해 준다.

ex>

속성명1:속성값1

속성명2:속성값2

3. Config/report\_from\_BASIC.html 의 디자인 내부에서 [@속성명:속성없을때 보여질 이름] 을 사용하여 정의한다.

이 중, 3번에 해당되는 방법은, 값을 지정할 수는 없지만, 일단 지정했는데 1번과 2번 방법으로 정의해 주지 않으면 팝업을 띄워서 그 값을 가져온다.

|  |  |  |  |
| --- | --- | --- | --- |
| **[@REPORT\_TITLE:보고서명]** | | | |
| 일련 번호 | [@ID:일련번호] | 점검자명 | [@NAME:점검자명] |
| 점검 항목 | [@TEST\_CONTENT:점검항목] | 페이지번호 | [%PAGE\_NUM] |
| 점검 장소 | [@TEST\_LOCATION:점검장소] | 점검일시 | [%TEST\_DATE:yyyy년 MM월 dd일 HH시 mm분 ss초] |
| 특기 사항 | [@SPECIFIC\_CONTENTS:특기사항] | | |

위와 같은 디자인이 있다고 하자. 위에서 정의된 속성은

[@REPORT\_TITLE:보고서명],

[@ID:일련번호],

[@NAME:점검자명],

[@TEST\_CONTENT:점검항목],

[@TEST\_LOCATION:점검장소],

[@SPECIFIC\_CONTENTS:특기사항]

 이 있다.

그런데 Config/Properties.txt의 내용이 다음과 같다.

NAME:

ID:

TEST\_LOCATION:리얼타임웨이브

REPORT\_TITLE:

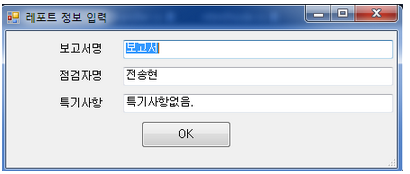
TEST\_CONTENT:시험적으로 레포트 뽑아봄.

필요한 속성은, REPORT\_TITLE,ID,NAME,TEST\_CONTENT,TEST\_LOCATION,SPECIFIC\_CONTENTS 로 5개인데,

Properties.txt 파일에서는 TEST\_LOCATION,TEST\_CONTENT 2가지밖에 없으므로 다음과 같은 팝업창이 뜬다.

아래 팝업창에서 보고서명,점검장명,특기사항 이라는 글자는 디자인에서 [@REPORT\_TITLE:보고서명] 와 같이 이미 정의한 글자 그대로 나오며,

이렇게 정의하지 않고 그냥 [@REPORT\_TITLE]과 같이 정의했다면 "보고서명" 대신 "REPORT\_TITLE"이 보였을 것이다.



이렇게 입력된 속성을 데이터에 추가해 보자.

#4CELL\_TABLE\_LEVEL=제목1

%TITLE:열1,열2,열3,열4

값1,값2,@REPORT\_TITLE,성공

**제목1** 

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 열1 | | | | | 열2 | 열3 | 열4 |
|  | 값1 | | | | 값2 | 보고서 | 성공 |

위 테이블에서 @REPORT\_TITLE이 위에서 입력받은 "보고서" 라는 글자로 바뀌어 있는 것을 확인할 수 있다.

위에서 사용된 %TEST\_DATE 나 %PAGE\_NUM 도 사용가능하다.

#4CELL\_TABLE\_LEVEL=제목1

%TITLE:열1,열2,열3,열4

값1,%PAGE\_NUM,@REPORT\_TITLE,%TEST\_DATE

**제목1** 

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 열1 | | | | | 열2 | 열3 | 열4 |
|  | 값1 | | | | 1 | 보고서 | 2014-04-21 23:43:19 |

### 1-7-3) 예약어목록:

%TEST\_DATE : 레포트가 생성된 날짜를 자동으로 가져옴.

%PAGE\_NUM : 페이지 번호를 가져옴.

%INDEX : 테이블 내에서 행의 번호

테이블을 만들 때, 궂이 줄의 번호를 넣고 싶지 않을 수도 있다. 이 때는 %INDEX를 이용한다.

#4CELL\_TABLE=제목1,제목2

%TITLE:열1,열2,열3,열4

%INDEX(1),값2,값3,OK

%INDEX,값2,값3,OK

%INDEX,값2,값3,OK

%INDEX,값2,값3,OK

%INDEX,값2,값3,OK

%INDEX,값2,값3,OK

위에서 %INDEX(1) 의 의미는 1부터 시작하겠다는 의미이다. 그냥 %INDEX 를 쓰면 하나씩 증가된다.

|  |  |  |  |
| --- | --- | --- | --- |
| 열1 | 열2 | 열3 | 열4 |
| 1 | 값2 | 값3 | OK |
| 2 | 값2 | 값3 | OK |
| 3 | 값2 | 값3 | OK |
| 4 | 값2 | 값3 | OK |
| 5 | 값2 | 값3 | OK |
| 6 | 값2 | 값3 | OK |

응용하여 4번째줄부터 다시 1부터 시작하고 싶다면 아래와 같이 하면 된다.

#4CELL\_TABLE=제목1,제목2

%TITLE:열1,열2,열3,열4

**%INDEX(1)**,값2,값3,OK

%INDEX,값2,값3,OK

%INDEX,값2,값3,OK

**%INDEX(1)**,값2,값3,OK

%INDEX,값2,값3,OK

%INDEX,값2,값3,OK

|  |  |  |  |
| --- | --- | --- | --- |
| 열1 | 열2 | 열3 | 열4 |
| 1 | 값2 | 값3 | OK |
| 2 | 값2 | 값3 | OK |
| 3 | 값2 | 값3 | OK |
| 1 | 값2 | 값3 | OK |
| 2 | 값2 | 값3 | OK |
| 3 | 값2 | 값3 | OK |

### 1-7-4) 예약 함수 목록

%TABLE\_RESULT:PASS/FAIL

: TABLE에 해당되는 결과를 저장한다.

OK를 하면  [%NUM\_OF\_PASS\_RESULT] 의 COUNT가 올라가고,

FAIL을 하면 [%NUM\_OF\_FAIL\_RESULT] 의 COUNT가 올라간다.

[%NUM\_OF\_TABLE\_RESULT] 의 숫자는 호출할 때마다 올라간다.

나중에 [%TOTAL\_RESULT:조건,PASS글자/FAIL글자] 에서 사용된다.

%NEXT\_PAGE

: 다음 페이지로 넘어간다. BASIC\_FORM을 다시 만들어 해당 CONTENT영역에 테이블을 추가하기 시작한다.

테이블을 만들다 보면 너무 많은 줄을 한 페이지에 추가하는 것이 보기가 좋지 않아 일부러 두 개 이상의 테이블로 나눌 때가 있다.

혹은 두 개 이상의 테이블을 사용할 때, 다음 테이블을 반드시 다른 페이지에서 시작하게 하고 싶을 때도 있다.

이 때, %NEXT\_PAGE를 데이터파일의 중간에 넣으면 페이지가 강제로 넘어간다.

#4CELL\_TABLE=제목1,제목2

%TITLE:열1,열2,열3,열4

값1,값2,값3,OK

값1,값2,값3,OK

값1,값2,값3,OK

값1,값2,값3,OK

%NEXT\_PAGE

값1,값2,값3,OK

값1,값2,값3,OK

**제목1**   
**제목2**

|  |  |  |  |
| --- | --- | --- | --- |
| 열1 | 열2 | 열3 | 열4 |
| 값1 | 값2 | 값3 | OK |
| 값1 | 값2 | 값3 | OK |
| 값1 | 값2 | 값3 | OK |
| 값1 | 값2 | 값3 | OK |

(다음페이지)

|  |  |  |  |
| --- | --- | --- | --- |
| 열1 | 열2 | 열3 | 열4 |
| 값1 | 값2 | 값3 | OK |
| 값1 | 값2 | 값3 | OK |

## 1-8) 차트 추가

차트를 추가하려면 일단 차트를 정의한 후, 정의할 때 사용한 차트ID를 사용하여 CELL에 값을 넣어주면 된다.

아래는 차트를 정의하는 문법이다.

**%CHART**:차트타입,차트id,["항목명1","항목명2","항목명3"],[[x,y1,y2,y3],[x,y1,y2,y3],...]

:차트의 데이터를 설정하고 나중에 사용하도록 한다.

차트타입: line,pie,bar,area,radar,scatter가 가능하다. javascript에 차트 추가하면 됨.

세부적으로 line,spline,pie,3dpie,donut,bar,sbar(stacked bar),barh(bar horizontal),sbarh(stacked bar horizontal),area,sarea(stacked\_area),radar,scatter

차트id : 테이블 데이터를 설정할 때, CELL의 속성이 [%CHART] 라면 데이터를 직접 적는 대신 여기에 적은 차트id를 적으면 해당 차트로 치환된다. **(주의! 차트 id는 중복될 수 없음)**

항목명 : 범례에 표시될 항목명으로서, 위의 y1,y2,y3가 각각 무엇을 나타내는지를 써 넣는다.

위에서 x는 x축의 기준이다. 숫자를 써도 되지만, "문자"로 쓰게 되면 해당 문자가 x축에 표시된다.

이후에는 차트에서 받을 데이터를 배열형식으로 쓴다.

ex>

%CHART:sarea,stacked\_area\_chart,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:area,area\_chart,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:line,line\_chart,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:bar,bar\_chart,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:sbar,s\_bar\_chart,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:barh,bar\_h\_chart,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:sbarh,s\_bar\_h\_chart,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:spline,spline\_chart,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:scatter,scatter\_chart,["name1","name2"],[[[100,200],[1,2],[5,7],[10,12]],[[10,20],[14,25],[56,72],[11,19]]]

%CHART:pie,pie\_chart,["n1","n2","n3","n4"],[100,30,50,15]

%CHART:3dpie,3dpie\_chart,["n1","n2","n3","n4"],[100,30,50,15]

%CHART:donut,donut\_chart,["n1","n2","n3","n4"],[100,30,50,15]

%CHART:radar,radar\_chart,["n1","n2","n3","n4"],[100,30,50,15]

차트를 이용하기 위해서는 위와 같이 일단 차트를 정의해야 한다. 차트의 정의는

차트타입,차트id,차트데이터

로 이루어지는데, 차트 데이터는 대부분 비슷하지만 차트의 구조를 알아야 정확한 차트를 만들기가 쉽다.

단, 디자인원형에서 [%CHART:가로 세로] 로 정의되어 있어야 한다.

[#CHART\_SAMPLE] [%TABLE\_TITLE]

|  |  |  |  |
| --- | --- | --- | --- |
| img | [%CHART:250 250] | [%CHART:250 250] | [%CHART:250 250] |

위 테이블은 차트만을 표시하는 테이블이다.(헤더영역 없음)

이 영역에 차트를 넣기 위해서는 일단 차트를 정의해야 한다.

%CHART:line,line\_chart1,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:line,line\_chart2,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:line,line\_chart3,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

위에서 차트는 line차트로 3개를 정의하였다.

그리고 각각의 차트 이름은 line\_chart1, line\_chart2, line\_chart3 로 지정했다.

위 차트는 두 개의 값그룹을 가지는데, 여기서 x축에 표시될 이름이 "test1", "test2", "test3", "test4", "test5"에 해당한다.

이 값을 0,1,2,3,4,5 와 같은 숫자로 해도 상관없다.

그리고 첫 번째 값인 ["test1", 10,99] 에서 10에 해당되는 부분이 첫번째 값 그룹이고, 99에 해당되는 부분이 두 번째 값 그룹으로서

각각의 값 그룹은 "name1"과 "name2"라는 이름을 가지게 될 것이다.

하지만 위 데이터로는 차트를 아직 표시할 수 없다. 차트 테이블 선택을 해 주지 않아서이다.

차트 테이블을 선택해 주자. 차트 테이블의 이름은 위의 디자인원형에서 보이듯이 #CHART\_SAMPLE 이다.

디자인에 [%CHART:250 250] 이라고 된 곳이 3곳이 있으므로 차트는 한 줄에 3개까지 넣을 수 있다.

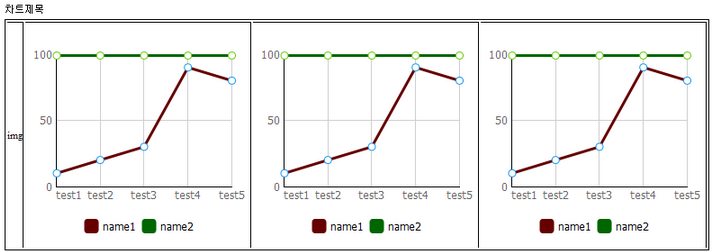
%CHART:line,line\_chart1,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:line,line\_chart2,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

%CHART:line,line\_chart3,["name1","name2"],[["test1", 10,99], ["test2", 20,99], ["test3", 30,99], ["test4", 90,99], ["test5", 80,99]]

#CHART\_SAMPLE=차트제목

line\_chart1,line\_chart2,line\_chart3



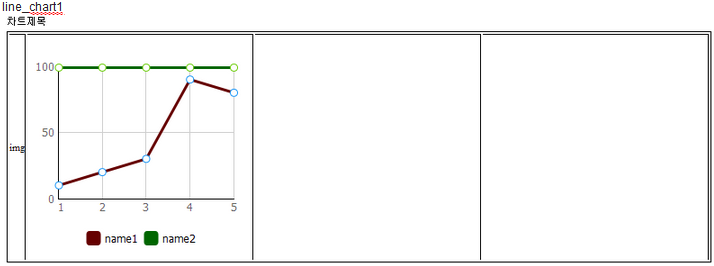
일단 편의를 위해 하나의 그래프만 쓰자.

값을 좀 바꾸어보겠다. 첫번째 차트의 값을 바꿀텐데, X축을 1,2,3,4,5로 바꾸면 다음과 같이 된다.

%CHART:line,line\_chart1,["name1","name2"],[[1, 10,99], [2, 20,99], [3, 30,99], [4, 90,99], [5, 80,99]]

#CHART\_SAMPLE=차트제목

line\_chart1



X축의 이름이 1,2,3,4,5로 바뀐 것을 볼 수 있다.

데이터도 바꾸어보자. 첫번째 그래프의 Y축을 10,20,30,40,50, 두번째 그래프의 Y축을 100,40,50,20,60으로 바꾸어보겠다.

그리고 첫번째 그래프의 이름을 GRAPH1, 두번째 그래프의 이름을 GRAPH2로 바꾸어보겠다.

%CHART:line,line\_chart1,["GRAPH1","GRAPH2"],[[1, 10,100], [2, 20,40], [3, 30,50], [4, 40,20], [5, 50,60]]

#CHART\_SAMPLE=차트제목

line\_chart1



그래프의 모양이 변했다.

## 1-9) 테이블 동적 추가 및 변경

### 1-9-1) 테이블 추가

**:**테이블을 디자인으로 등록할 수도 있지만, 동적으로 테이블을 만들고 싶을 수도 있다.

예를 들어, 디자인 해 놓은 테이블의 COLUMN이 4개가 최대인데, 8개나 10개를 만들고 싶다고 하자.

그렇다면 데이터 파일에서 아래의 NEW\_TABLE명령을 이용하여 새로운 테이블을 만들 수 있다.

**%NEW\_TABLE**: ID(테이블ID)[,BASE(기본디자인)]

ex1> %NEW\_TABLE : ID(TABLE\_TEST)

ex2> %NEW\_TABLE : ID(TABLE\_TEST), BASE(4CELL\_TABLE)

ex3> %NEW\_TABLE : ID(TABLE\_TEST), BASE(#4CELL\_TABLE)

위에서 ID를 TABLE\_TEST로 지정했으면 다음부터 테이블을 선택할 때, #TABLE\_TEST라는 이름으로 선택할 수 있다.

일단 테이블을 만들었으니 내부의 내용을 만들어보자.

%NEW\_TABLE:ID(TABLE\_TEST)

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

위에서 %TITLE: 로 시작하면 그 행은 HEADER가 된다.

나머지는 일반 데이터열이 된다.

항목에 [%BLANK]라고 적으면, 해당 열에는 데이터 치환이 가능해 진다.

위처럼 새 테이블을 정의한 후, 값을 넣으려면 정해진 테이블ID인 TABLE\_TEST로 테이블을 선택한다.

%NEW\_TABLE:ID(TABLE\_TEST)

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

#TABLE\_TEST

%TITLE:열1,열2,열3,열4

값1,값2,값3,값4

값1,값2,값3,값4

결과는 다음과 같이 나온다.



### 1-9-2) 기존 테이블 참조하여 추가

테이블을 새로 만드는 것은 했지만, 테이블의 스타일이나 디자인이 맘에 들지 않으면 기존의 테이블에서 가져올 수도 있다.

이 경우, 테이블을 만들 때, BASE(기존테이블ID) 를 추가하여 만들면 된다.

%NEW\_TABLE:ID(TABLE\_TEST), BASE(4CELL\_TABLE)

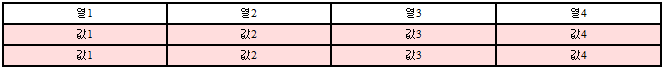
이후 똑같이 TITLE과 열을 추가해 준다.

%NEW\_TABLE:ID(TABLE\_TEST), BASE(4CELL\_TABLE)

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

값을 위와 같이 추가하고 실행하면 결과는 다음과 같이 나올 것이다.



본래 #4CELL\_TABLE 테이블은 아래와 같이 생겼다.

**[%TABLE\_TITLE]**  
**[%TABLE\_TITLE]**

|  |  |  |  |
| --- | --- | --- | --- |
| [%H][%BLANK] | [%BLANK] | [%BLANK] | [%BLANK] |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |
| [%BLANK] | [%BLANK] | [%BLANK] | [%RESULT] |

여기서 [%TABLE\_TITLE] 부분은 값이 없었으므로 아무것도 표시되지 않았다.

 값을 주고 다시 테이블을 만들어보자.

%NEW\_TABLE:ID(TABLE\_TEST), BASE(4CELL\_TABLE)

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

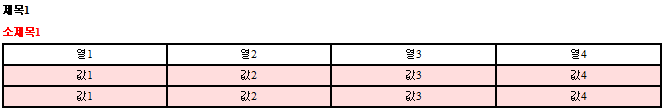
[%BLANK],[%BLANK],[%BLANK],[%BLANK]

#TABLE\_TEST=**제목1,소제목1**

%TITLE:열1,열2,열3,열4

값1,값2,값3,값4

값1,값2,값3,값4



위에서 테이블의 스타일이 본래의 스타일과 조금 다른 것을 눈치챘을 것이다. 데이터줄의 색깔이 번갈아가며 나오게 되어있었는데

지금은 한 줄만 반복되는 것을 볼 수 있다. 이는 테이블을 정의할 때 한 줄만 정의했기 때문이다.

%NEW\_TABLE:ID(TABLE\_TEST), BASE(4CELL\_TABLE)

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

**[%BLANK],[%BLANK],[%BLANK],[%BLANK]**

두 줄 모두의 스타일을 가져오기 위해서는 똑같이 두 줄로 테이블을 만들어 주어야 한다.

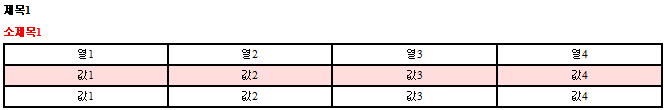
%NEW\_TABLE:ID(TABLE\_TEST), BASE(4CELL\_TABLE)

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

**[%BLANK],[%BLANK],[%BLANK],[%BLANK]**

**[%BLANK],[%BLANK],[%BLANK],[%BLANK]**

다시 실행시키면 아래와 같이 바뀐다.



### 1-9-3 ) 새로 만든 테이블 모양 변경

테이블의 디자인은 사용자가 원하는 대로 디자인할 수도 있다.

기본적으로 셀을 합치는 %SPAN, 셀의 크기를 변경하는 %SIZE, 스타일을 변경하는 %ROW\_STYLE, %CELL\_STYLE(혹은 %STYLE) 을 사용하여

테이블을 여러가지 모양으로 만들 수 있다.

4CELL\_TABLE을 레퍼런스 하지 않은 순수한 테이블을 만들어보자.

%NEW\_TABLE:ID(TABLE\_TEST)

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

#TABLE\_TEST

%TITLE:열1,열2,열3,열4

값1,값2,값3,값4

값1,값2,값3,값4

테이블의 모양은 다음과 같이 될 것이다.



여기에서 헤더를 하나 더 추가해 보자.

%NEW\_TABLE:ID(TABLE\_TEST)

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

#TABLE\_TEST

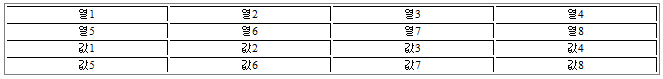
%TITLE:열1,열2,열3,열4

%TITLE:열1,열2,열3,열4

값1,값2,값3,값4

값1,값2,값3,값4

결과는 아래와 같이 나올 것이다.



#### 1-9-3-1 ) **셀 합치기**

위 예제에서 header영역이 2개가 되었는데, 이 중 위의 헤더영역의 두 개의 열을 합쳐보자.

%NEW\_TABLE:ID(TABLE\_TEST)

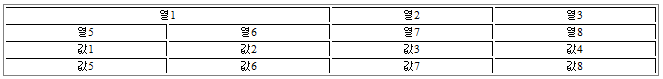
%TITLE:[%BLANK]**%SPAN(2)**,[%BLANK],[%BLANK]

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

위처럼 %SPAN(2)라고 하면 두 개의 COLUMN이 합쳐진다. COLUMN이 합쳐졌으므로, 치환영역을 나타내는 [%BLANK]를 하나 없애주었다.



아래 값 영역에서도 마찬가지 문법이 적용가능하다.

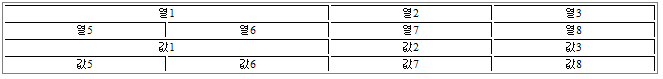
%NEW\_TABLE:ID(TABLE\_TEST)

%TITLE:[%BLANK]%SPAN(2),[%BLANK],[%BLANK]

%TITLE:[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK]**%SPAN(2)**,[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]



또는 세로 두 열을 합칠 수도 있다. 이는 %SPAN( **,**n)을 사용할 수 있다.

아래 행과 합칠 것이므로,

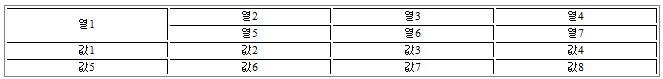
%NEW\_TABLE:ID(TABLE\_TEST)

%TITLE:[%BLANK]**%SPAN(,2)**,[%BLANK],[%BLANK],[%BLANK]

%TITLE:[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]



#### 1-9-3-2 ) **셀 크기 조정**

테이블을 만들 때 기본적으로 크기는 내부의 내용이 많으면 커지고 내부의 내용이 적으면 작아진다.

그러나 때로는 보기 좋게 하기 위하여 크기를 인위적으로 고정해야 할 때가 있다.

이를 위해 %SIZE(WIDTH, HEIGHT) 명령을 쓸 수도 있다.

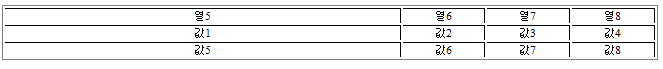
%TITLE:[%BLANK]**%SIZE(500)**,[%BLANK]**%SIZE(100)**,[%BLANK]**%SIZE(100)**,[%BLANK]**%SIZE(100)**

#TABLE\_TEST

%TITLE:열5,열6,열7,열8

값1,값2,값3,값4

값5,값6,값7,값8



위에 정의된 크기(비율)대로 크기가 변경된 것을 볼 수 있다.

#### 1-9-3-3 ) 셀 스타일 조정

색깔을 포함한 폰트크기,모양등을 바꿀 수 있는 옵션인 %STYLE(options) 에 대해 알아보겠다.

%STYLE(options) 명령에서 options에 해당하는 부분은 HTML의 CSS문법대로 사용되는데, 기본적인 폼은 다음과 같다.

스타일명1:스타일속성1; 스타일명2:스타일속성1

또는

스타일명:스타일속성1 스타일속성2 스타일속성3 ; 스타일명2:스타일속성1

ex> background-color:#ffdddd ; width:100px; border:solid 1px

그러기 위해서는 HTML 문법의 스타일을 알 필요가 있다.

스타일 에 대해서는 많은 REFERENCE( <http://www.w3schools.com/css/css_table.asp>  ) 가 있으니 참조하기 바란다.

이 예제에서는 가장 기본적으로 배경색을 넣어 볼 것이다.

%NEW\_TABLE:ID(TABLE\_TEST)

%TITLE:**%ROW\_STYLE(background-color:#ffdddd)**[%BLANK],[%BLANK],[%BLANK],[%BLANK]

[%BLANK],[%BLANK],[%BLANK],[%BLANK]

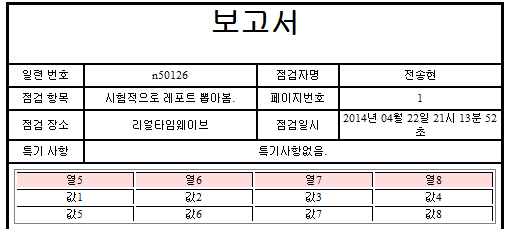
[%BLANK],[%BLANK],[%BLANK],[%BLANK]

#TABLE\_TEST

%TITLE:열5,열6,열7,열8

값1,값2,값3,값4

값5,값6,값7,값8



# 디자인파일 만들기

## 2-1) 공통규칙.

**[@속성명:표시이름]** -  /properties=속성파일.txt 플래그나 INFO/PROPERTIES.txt 안에 추가한다.

   표시이름은 고정된 속성이 아닐 시 추가할 때 텍스트박스에 나타날 이름이다.

표시이름은 한 번만 정의하면 되며, 이후부터는 [@속성]이라고 적어두기만 하면 파일에 정의된 값으로 치환된다.

**[%PAGE\_NUM]** - 페이지의 번호가 자동으로 치환된다.

**[%TEST\_DATE]** - 레포트하는 날짜/시간이 yyyy-MM-dd HH:mm:ss 형식으로 자동으로 치환된다.

**[%TEST\_DATE:format]** - 레포트하는 날짜/시간이 format 형식으로 자동으로 치환된다. format은 y,M,d,H,m,s를 조합하여

만들면 된다.

## 2-2) basic파일(Config/report\_form\_basic.html) 만들기 규칙

**[%CONTENT]** - 각 PAGE에서 내부 내용으로 치환될 영역.

**[%CHARTLIST]** - 데이터파일에서 정의된 차트의 목록을 내부 차트 생성 명령으로 치환한다.

## 2-3) 테이블파일(**ReportForms/\***) 만들기 규칙

아래는 테이블이 선택된 상황에서 내용으로 치환될 명령.

**[%H]**- HEADER가 될 줄의 칸 안에 [%H]를 넣어주면(아무 칸이나 하나만 넣어주면 됨) 그 행은 헤더로 취급된다.

**[%BLANK]** - 값으로 치환된다.

**[%COND]** - [%BLANK] 와 동일하지만 나중에 %RESULT가 나오면 비교하기 위한 임시값으로 저장된다.

**[%RESULT] -** 결과값이 나타남. report\_form\_BASIC.html 의 style태그 안에 정의된 PASS/FAIL 스타일을 사용하여

  성공시 PASS 스타일, 실패 시 FAIL 스타일로 나타낸다. 일반적으로 PASS 스타일은 녹색, FAIL스타일은 적색이다.

[%RESULT:성공문자열1,성공문자열2,...] - 성공문자열과 동일한 결과일 때 PASS 스타일을 사용한다.

[%RESULT:%COND] - 바로 이전에 나타난 [%COND]의 내용과 동일할 경우 PASS스타일을 사용한다.

[%RESULT:%COND(n)] - n번째 문자까지만 비교하여 맞으면 PASS.

[%RESULT:%COND(.)] - 정수부분만 비교하여 맞으면 PASS

**[%RESULT:%COND(.n)]** - 소숫점이후 n번째까지만 비교하여 맞으면 PASS.

**[%IMG:width height] -** width와 height 크기의 이미지로 치환. 내용은 이미지의 path가 되어야 한다.

**[%CHART:width height] -** 차트로 치환. 내용은 차트의 이름이 되어야 하고, 해당 차트의 데이터는 해당 부분에 오기 전에

    %CHART:명령으로 미리 지정되어 있어야 한다.

**[%NUM\_OF\_PASS\_RESULT]** - 레포트 데이터에서 %TABLE\_RESULT=OK(또는 PASS,성공,정상) 의 숫자

**[%NUM\_OF\_FAIL\_RESULT]**- 레포트 데이터에서 %TABLE\_RESULT=FAIL(사실은 PASS/OK/성공/정상이 아닐 때) 의 숫자

**[%NUM\_OF\_TABLE\_RESULT]** - 레포트 데이터에서 %TABLE\_RESULT 의 총 숫자

**[%TOTAL\_RESULT:조건,성공글자/실패글자]** - 총 결과 문자열로 치환된다.

EX> [%TOTAL\_RESULT:%NUM\_OF\_PASS\_RESULT>1,성공/실패] <-- %TABLE\_RESULT=OK

## 2-4) 표지문서 (Cover Page) 만들기 규칙

레포트를 작성할 때 제일 처음장에는 다른 형식의 레포트가 나와야 하는경우가 있다.

이 경우, 레포트 시작 시,

@@COVER

를 추가해 주면, Config/Report\_form\_BASIC.html 파일 대신 Config/cover\_page.html 파일을 제일 첫 장의 스타일로 적용하여 레포트 파일을 만든다. 기본적으로 cover\_page.html에 적용되는 모든 명령/속성/스타일은 Config/Report\_form\_BASIC.html 과 동일하다.

위 @@COVER 대신, @@COVER\_PAGE, @@COVERPAGE 도 동일하게 작동한다.

만일, 표지문서를 두 장 이상 써야 할 경우, @@COVER대신에 @@COVERS로 시작하고, COVERPAGE가 끝나고 일반 문서로 작성되어야 할 시점에 @@BASIC 이라고 쓰면 일반 페이지가 작성된다.

# 설정 파일정의 :

## 3-1) Config/Properties.txt

디자인의 [@속성]이 치환될 속성(고정값)을 적는다. 나중에 입력받을 값은 적지 않는다.

속성명1:값1

속성명2:값2

## 3-2) Config/Files\_to\_parse.txt

 - 레포트를 만들 소스파일(text파일) 의 목록을 정의한다. 인코딩을 지정할 수 있다.

데이터파일경로1(파일명포함):인코딩

데이터파일경로2(파일명포함):인코딩

...

%REMOVE\_THIS\_FILES

인코딩은  MULTIBYTE/UTF-8/EUC-KR/UNICODE/ASCII 를 지원한다.(DEFAULT는 UTF-8)

%REMOVE\_THIS\_FILES 명령은 선택사항이다. 쓰게되면 해당 데이터파일들은 레포트를 모두 만든 후 삭제된다.

## 3-3) Config/Path\_to\_save.txt

 - 만들어질 파일이 저장될 경로와 파일의 이름을 지정한다.

해당 파일이 없거나 파일명을 지정하지 않으면 REPORT.HTML 로 저장된다.

아래 예약어로 파일명에 추가사항을 적을 수 있다.

%date%: yyyyMMdd 형식의 연월일

%time%: HHmmss 형식의 시분초

%yyyy% : 연

%MM% : 월

%dd% : 일

%HH% : 시

%mm% : 분

%ss% : 초

%@속성% : 사용자가 지정한 속성.

# 실행시 arguments

## 4-1) /current\_path=실행파일경로

: 프로그램 내부에서 process로 실행시킬 경우, Reporter를 띄우는 프로그램이 현재 경로로 인식되므로, 해당 argument를 통해 메인경로를 지정해야 함.

만일 reporter를 실행하는 실행파일이 같은 경로에 있다면 해줄 필요 없음.

## 4-2) /properties=속성명1:속성,속성명2:속성,…

: 레포트에서 @속성 으로 정의된 항목과 치환될 항목을 직접 지정해 준다. 속성이 많아서 argument의 한계를 넘어갈 경우 4-2-3)의 property\_file을 사용한다.

ex> /properties=ID:REPORT-N0123,START\_DATE:2013-01-02%s01\:02\:04

### 4-2-1) /property\_file=속성정의파일경로[Encoding]

: 레포트에서 @속성 으로 정의된 항목과 치환될 항목이 정의된 파일의 경로를 적는다. 경로를 적은 이후 [인코딩]을 추가로 적으면 해당 인코딩으로 인코딩한다.

인코딩은 MULTIBYTE/UTF-8/UTF-16/UTF-32/ASCII/EUC-KR/UNICODE 중에 선택 가능하다. (UTF-16 과 UNICODE는 동일)

인코딩이 생략되거나 해당 ENCODING중의 하나가 아니면 UNICODE로 취급된다.

4-3) /src = 레포트데이터파일1.txt[Encoding],

레포트데이터파일2.txt[Encoding],...

/files\_to\_parse=레포트데이터파일1.txt[Encoding],레포트데이터파일2.txt[Encoding],...

: 레포트를 만드는 데 필요한 데이터파일의 경로를 적는다. 경로를 적은 이후[인코딩]을 추가로 적으면 해당 인코딩으로 인코딩한다.

인코딩은 MULTIBYTE/UTF-8/UTF-16/UTF-32/ASCII/EUC-KR/UNICODE 중에 선택 가능하다. (UTF-16 과 UNICODE는 동일)

인코딩이 생략되거나 해당 ENCODING중의 하나가 아니면 UNICODE로 취급된다.

마지막에 %REMOVE\_THIS\_FILES 를 넣으면 해당 데이터파일은 레포트 생성 이후 삭제된다.

## 4-4) /dst=레포트pdf파일 저장 경로.

:3-3 항목에 정의한 예약어가 그대로 적용된다.

## 4-5) /basic\_form=기본디자인폼 경로

: 기본 디자인폼은 레포트를 만드는 데 필요한 기본 디자인이 들어있는 html파일로서, 프로그램마다 다를 수 있다.

이 기본 디자인을 변경하면 디자인을 마음대로 바꿀 수 있으며, 그 디자인 내용은 일반 웹 디자인툴(나모웹에디터, 드림위버등)로 디자인하여 적용가능하다.

## 4-6) /forms=테이블파일1.html,테이블파일2.html,...

: 테이블이 정의되어있는 html파일들의 경로.

## 4-7) /make\_pdf

: pdf 파일을 preview창을 띄우지 않고 바로 만듬.

/path\_to\_save.txt파일이나 /dst에 지정한 파일명으로 저장된다.

## 4-8) /print\_now

:창이 뜨자 마자 프린트를 시작함.