

넘버쓰리 PHP

응용 · 실전 끝장내기

넘버쓰리 P.H.P

이호진 지음

응용·실전 끝장내기



저자 소개

대학에서 전기전자를 전공하였습니다. 마이크로프로세서, 하드웨어 설계 및 펌웨어, 로봇 등을 주로 개발하였으며 산업용 레이저 프린터, ARM, 언어 분석, 알고리즘 개발도 했습니다.

외국계 기업에서 Linux 운영체제, 백업 및 서버 가상화 등 다양한 비즈니스 경험으로 최근에는 웹과 모바일 관련한 개발을 하고 있습니다. 현재는 경험과 지식을 정리하고 후배들에게 전달하기 위해서 집필 활동에 전념하고 있습니다.

이호진

프로그램 언어가 발전하고 대중화되기 위해서는 몇 가지 요소들이 필요합니다. 먼저 사용자층이 많아야 하고, 다양한 응용 함수들이 폭넓게 존재해야 합니다. 프로그램의 언어 구조 처리가 아무리 좋아도 관련 기본 함수들이 많지 않다면 확산 속도는 오래 걸릴 것입니다. 기본 함수들을 통하여 다양한 응용 라이브러리가 만들어지고 그로 인해 프로그램 언어들은 더욱더 힘을 받게 됩니다.

이 세상의 코드들은 여러 사람들의 기여와 공여로 유기적인 시스템으로 동작합니다. 프로그램을 개발하면서 처음부터 끝까지 일일이 만들어 사용하기에는 너무나 많은 시간이 필요합니다. 대부분의 개발자들은 새로운 기능을 만드는 것보다는 기존에 만들어 놓은 기본 함수, 라이브러리를 이용하여 새로운 서비스를 만드는 데 더 많은 시간을 들입니다.

PHP는 빠르게 응용프로그램을 개발할 수 있도록 다양한 기본 함수들을 제공합니다. 이렇게 기초적인 동작을 하는 함수들을 제공하는 것은 프로그램 언어들이 순수한 문법만 가지고는 동작할 수 없기 때문입니다. 심지어 화면에 “hello world!” 글자를 출력하는 것도 기본 함수가 필요합니다. 초보자일수록 언어에서 제공하는 기본 함수들을 학습하고 습득하는 것은 중요합니다. 기본 함수들을 많이 알수록 응용프로그램을 만드는 속도는 짧아지고 코드는 간결해질 것입니다.

PHP는 C 언어로 개발된 인터프리터 언어로 C 언어의 내장 함수들과 상당 부분 유사합니다. 또한 많은 시간 동안 웹 서비스 용도로 응용되면서 다양한 함수들이 추가되었습니다. 이러한 유사한 함수들과 기능들은 기존에 작성된 코드들을 포팅하여 개발하는 데도 상당한 도움이 됩니다.

책의 후반부에는 PHP의 라이브러리의 바다라고 할 수 있는 컴포저 및 패키지에 대해서 간략하게 소개합니다. 컴포저를 통하면 전 세계의 수많은 개발자들이 만들어 놓은 패키지

라이브러리를 다운로드하여 사용이 가능합니다. 또한 여러 종류의 PHP 프레임워크도 다운로드해서 사용할 수 있습니다. 독자 여러분이 만든 라이브러리나 소스 등을 배포할 수도 있습니다.

이처럼 개발을 위한 기본 함수들을 튼튼하게 학습하는 것은 매우 중요합니다. 필자는 1권과 더불어 PHP의 기본 함수들을 체계적으로 학습하기 위해서 별도의 3권을 집필하였습니다. PHP의 전반적인 내부 함수들을 살펴보고 응용 서비스를 개발하는 데 도움되는 함수들을 설명하고자 합니다.

3권은 1권 문법에서 설명하지 못한 부분을 추가적으로 다시 설명합니다. PHP는 20년 이상의 세월과 수많은 개발자들의 기여로 발전된 고급 웹 언어입니다. 그만큼 설명할 부분도 많고, 관련 기술 또한 방대합니다. 1권에서는 지면 관계상 다양한 함수를 응용하여 설명하지 못한 부분을 예제로 좀 더 응용이 가능하도록 하였습니다.

또한 PHP를 학습하면서 지루하지 않도록 1권의 내용과 별도로 분리하여 내장된 함수들과 응용 방법에 대해서 3권으로 집필이 되었습니다. 전체적인 함수들의 종류와 개념, 실습을 위하여 간략하게 정리했습니다.

사전 학습

3권을 학습하기 위해서는 먼저 1권의 기본 문법이 학습되어야 합니다. 3권은 PHP의 내장된 함수들과 응용 방법에 대해서 설명하기 때문입니다. 따라서 PHP의 기본적인 문법과 구조를 미리 습득해야 합니다. 1권 문법을 옆에 두고 함께 학습하는 것도 좋은 방법입니다.

프로젝트 준비

본격적인 프로젝트 개발을 시작하기에 앞서 다양한 PHP의 내장된 함수들을 학습합니다. 내장된 함수들을 학습하는 것은 프로젝트를 개발할 때 PHP의 내장 함수를 통하여 빠르게 코드를 작성하기 위함입니다. 또한 향후 개발될 제품의 코딩 시간을 줄여주는 효과도 있습니다.

내장된 함수를 학습하는 것은 자신이 구현하고자 하는 기능의 개발 범위를 아는 데 도움이 될 것입니다. 기존 함수들을 이용하여 프로젝트를 구성해야 할지, 또는 새로운 추가 기

능을 구현하는 로직을 만들어야 할지를 판별할 수 있습니다.

코드분석

PHP에서 제공하는 내장 함수들을 많이 아는 것은 다른 사람들의 코드를 분석할 때도 도움이 됩니다. 예를 들어 기초 라이브러리와 같은 코드들은 직접 만들어내는 함수보다는 기존 함수들을 이용하여 새롭게 생성된 함수나 클래스 등이 대부분입니다. 해당 코드들이 어떻게 동작하는지, 새로 만들어진 함수인지를 판별할 수 있습니다.

내장 함수

PHP의 응용 함수들은 www.php.net 공식 사이트를 방문하면 수많은 함수들의 동작 매뉴얼을 확인할 수 있습니다. 하지만 영어 설명과 부족한 내용으로 초보자들이 쉽게 각각의 기능을 이해하는 것은 쉽지 않습니다.

함수별 정리

PHP는 수많은 내장 함수들을 제공하고 있습니다. 이를 다 파악하고 사용하기에는 어렵습니다. 3권은 내장된 함수들을 유사한 기능 중심으로 정리하여 설명했습니다. 또한 실제 업무에서 함수들을 쉽게 찾을 수 있도록 레퍼런스처럼 인덱스화했습니다.

찾아볼 수 있는 책

PHP는 오랜 시간 동안 개발되고 누적된 다양한 응용 함수들이 제공됩니다. 함수들은 너무나도 방대하여 심지어 어떤 기능의 함수가 있는지조차 확인하기도 어렵습니다. 3권에서는 PHP의 함수들을 분류별로 정리하고 동작을 설명합니다. 또한 프로그램을 작성하면서 찾아볼 수 있도록 정리하였습니다.

간결한 코드

책에서 설명한 특정 기능을 학습하기 위해서 짧은 코드로 구성했습니다. 전반적으로 10~20줄 이내의 코드로 쉽게 소스를 이해하고 학습할 수 있습니다.

응용 예제

파일 처리, JSON 등 기본 함수들을 통하여 실제적인 작업을 할 수 있도록 예제를 구성했습니다. 응용 예제를 통하여 실제 프로젝트에 코드를 응용하여 사용할 수 있습니다.

저자 사이트

방대한 양의 정보를 모두 책 한 권으로 설명하기에는 부족함 부분이 많습니다. 필자는 이 책에서 설명하지 못한 부분을 독자와 공유하기 위해서 별도의 사이트를 준비했습니다. 이 책을 다 읽은 후에 <http://hojin.io> 사이트에 접속하여 부족한 내용들을 추가 학습하면 독자 여러분께 도움이 될 것입니다.

소스 코드

이 책의 예제 코드는 깃허브를 통해서 다운로드할 수 있습니다. 깃허브 주소는 다음과 같습니다. <https://github.com/infohojin/php>

저자 소개	v
머리말	vi
대상 독자	viii
책의 방향	x

Chapter 01 필터 1

01.1 필터의 유용성	2
01.1.1 필터 목록	3
01.1.2 유효성 검사	4
01.2 필터 상수	5
01.2.1 정수 확인: FILTER_VALIDATE_INT	5
01.2.2 정수 확인 및 범위	6
01.2.3 IP 주소 유효성: FILTER_VALIDATE_IP	7
01.2.4 IPv6 주소 유효성	8
01.2.5 쿼리스트링과 URL 유효성 검사: FILTER_VALIDATE_URL	8
01.2.6 이메일 주소 검사	9
01.2.7 문자열 체크: FILTER_SANITIZE_STRING	10
01.2.8 URL 유효성 검사	11
01.2.9 정규표현	11
01.2.10 특수문자	12
01.2.11 함수 적용	17
01.3 필터 함수	17

Chapter 02 배열 22

02.1 배열 생성	22
02.1.1 변수 할당	22

02.1.2	값과 키	26
02.1.3	변수 결합과 분리	30
02.1.4	결합 및 분리	32
02.1.5	출력	36
02.1.6	추가 및 삭제	36
02.2	배열 검사	41
02.2.1	검사	41
02.2.2	중복	43
02.3	배열 위치	44
02.3.1	위치 이동	45
02.3.2	위치 확인	47
02.4	배열 정렬	48
02.4.1	오름차순	48
02.4.2	내림차순	49
02.4.3	정렬 예제	49
02.4.4	자연 순서 정렬	51
02.4.5	다차원 정렬	53
02.5	배열 외부 함수	54
02.5.1	외부 함수 호출	54
02.5.2	외부 함수 정렬	56
02.5.3	콜백	59
02.6	배열 연산	62
02.6.1	개수	62
02.6.2	범위	64
02.6.3	연산 처리	65
02.6.4	비정렬	66
02.6.5	변환	69
02.6.6	집합	70
02.7	비교	75
02.7.1	값의 비교	75
02.7.2	키의 비교	81

03.1 문자열 변수.....	83
03.1.1 문자열 길이.....	84
03.1.2 문자열 카운트.....	86
03.1.3 단어 개수 측정.....	87
03.2 문자열 자르기.....	88
03.2.1 문자열 제거.....	88
03.2.2 공백 제거.....	89
03.3 문자열 검색.....	90
03.3.1 첫 번째 검색.....	91
03.3.2 마지막 검색.....	92
03.3.3 첫 번째 위치.....	92
03.3.4 마지막 위치.....	93
03.3.5 마스크 필터.....	94
03.3.6 매칭 검색.....	96
03.4 문자열 비교.....	97
03.4.1 문자열 비교.....	97
03.4.2 문자열 Binary safe	99
03.4.3 자연 순서.....	101
03.4.4 유사성	103
03.5 문자열 치환.....	106
03.6 문자	109
03.7 구분화.....	112
03.7.1 토큰	112
03.7.2 문자열 분리.....	114
03.7.3 래핑	115
03.7.4 변수 해석.....	116
03.8 문자열 조작.....	117
03.8.1 문자열 순서.....	117
03.8.2 대소문자.....	117
03.8.3 낙타 표기.....	118

03.9 변환	120
03.9.1 숫자 표기	120
03.9.2 통화 표시	121
03.10 인코딩	123
03.11 랜덤	125
03.12 해시 및 암호화	127
03.12.1 해시	127
03.12.2 MD5	129
03.12.3 crc32	130
03.12.4 sha1	131
03.12.5 crypt	132
03.12.6 str_rot13	133
03.13 문자열 출력	134
03.13.1 출력	134
03.13.2 포맷 출력	135
03.13.3 포맷 입력	139
03.14 html 문자열	140
03.14.1 백슬래시	140
03.14.2 라인 브레이크	142
03.14.3 태그 제거	142
03.14.4 html entities	143
03.14.5 메타	148
03.14.6 escape	149
03.15 로케일 및 코드	151

Chapter 04 JSON 153

04.1 JSON 문법	154
04.1.1 문자열 표현	154
04.1.2 데이터 표현	154
04.1.3 배열 표현	155

04.1.4 객체 표현	155
04.1.5 객체 이중화 표현	157
04.1.6 객체 키 표현	157
04.1.7 주식	158
04.2 JSON 인코딩	158
04.3 JSON 디코딩	160
04.3.1 배열 디코딩	160
04.3.2 다중 배열 디코딩	162
04.4 객체 직렬화	164

Chapter 05 날짜 167

05.1 실시간 환경 설정	167
05.2 날짜	168
05.3 시간	169
05.3.1 시/분/초	169
05.3.2 마이크로 초	173
05.3.3 타임 스탬프	174
05.3.4 일출/일몰	176
05.4 출력 포맷	178
05.4.1 시간/날짜	178
05.4.2 포맷	180
05.5 유효성	187
05.6 DateTime 클래스	188
05.6.1 클래스 정의	188
05.6.2 메서드	189
05.7 달력	199
05.7.1 설정	199
05.7.2 Julian	201
05.7.3 Gregorian	205
05.7.4 Jewish	206
05.7.5 French Republican	207

05.7.6 부활절	208
05.8 타임존	209
05.8.1 클래스	209
05.8.2 메서드	210
05.8.3 타임존 명칭	216

Chapter 06 파일 제어 225

06.1 파일 시스템	226
06.2 디렉터리	226
06.2.1 디렉터리 확인	227
06.2.2 디렉터리 생성	228
06.2.3 서브 디렉터리	230
06.2.4 파일 목록	231
06.2.5 경로	234
06.2.6 리얼 패스	236
06.2.7 디렉터리 삭제	237
06.2.8 디렉터리 이름 변경	238
06.2.9 경로 변경	239
06.2.10 디렉터리 접근	240
06.2.11 디렉터리 용량 표시	244
06.3 권한 설정	245
06.3.1 소유권 변경	245
06.3.2 모드 변경	246
06.3.3 그룹 변경	250
06.4 파일 정보	251
06.4.1 파일 확인	251
06.4.2 날짜와 시간	253
06.4.3 파일 종류	256
06.4.4 파일 경로 및 확장자	257
06.4.5 링크	258
06.4.6 그 외 처리	262

06.5 파일 열기.....	263
06.5.1 파일 모드.....	264
06.5.2 파일 닫기.....	265
06.6 파일 데이터 읽기.....	266
06.6.1 파일 끝.....	266
06.6.2 한 글자 읽기.....	267
06.6.3 한 줄 읽기.....	269
06.6.4 파일 크기.....	271
06.6.5 지정 용량 읽기.....	272
06.6.6 포인터 조작.....	273
06.6.7 파일 콘텐츠 출력.....	275
06.7 파일 쓰기.....	275
06.7.1 쓰기(W).....	276
06.7.2 쓰기 추가(a).....	277
06.7.3 파일 권한.....	277
06.7.4 임시 파일.....	279
06.7.5 파일 잠금과 해제.....	281
06.8 파일 삭제.....	282
06.9 파일 복사.....	283
06.10 file.....	286
06.11 그 외 함수.....	288

Chapter 07 CSV 293

07.1 샘플 데이터.....	293
07.2 CSV 쓰기.....	294
07.3 CSV 읽기.....	295

Chapter 08 정규표현식 299

08.1 정규 패턴.....	299
-----------------	-----

08.2 패턴 변환.....	301
08.3 패턴 분리.....	302

Chapter 09 URL 305

09.1 네트워크 연결.....	305
09.1.1 IP 주소.....	305
09.1.2 IP 주소의 구성.....	306
09.1.3 IPv6.....	308
09.1.4 IP 및 맥 주소 확인 방법.....	308
09.1.5 도메인 주소.....	309
09.1.6 URL 쿼리스트링.....	310
09.2 URL 함수.....	311
09.2.1 URL 구분.....	311
09.3 암호화.....	313
09.3.1 URL 인코딩.....	313
09.3.2 base64 URL 처리.....	314
09.3.3 RFC1738 URL 처리.....	315
09.4 호스트 정보.....	316
09.4.1 IP 주소 확인.....	316
09.4.2 호스트명.....	317
09.4.3 포트.....	318
09.5 DNS 정보.....	319

Chapter 10 아파치 함수 323

Chapter 11 통신 331

11.1 소켓 통신.....	331
11.2 AJAX.....	333

11.2.1 Javascript & JQUERY	334
11.2.2 Javascript 예제	334
11.3 JQuery 예제	336

Chapter 12 cURL 339

12.1 설치	339
12.1.1 모듈 설정	340
12.1.2 모듈 확인	340
12.1.3 모듈 체크	341
12.2 기본 동작	342
12.2.1 초기화	342
12.2.2 전송 옵션	343
12.2.3 전송 실행	344
12.2.4 세션 종료	344
12.3 공유 핸들	345
12.4 멀티 핸들	346
12.5 cURL 오류 처리	352
12.6 cURL 그 외 함수	354
12.7 POST 접속 응용	360
12.8 파일 업로드	361

Chapter 13 외부 처리 363

13.1 시스템 함수	363
13.2 프로세스	365

Chapter 14 메일 369

14.1 SMTP	369
14.2 메일 작성	370

Chapter 15 오류 처리 함수 371

15.1 오류	371
15.2 오류 출력	373
15.3 역추적	375
15.4 오류 핸들	377

Chapter 16 함수 379

16.1 함수 목록	379
16.2 함수 인자	381
16.3 콜백 호출	383
16.4 메서드 호출	385
16.5 틱 실행	387

Chapter 17 HTML & FORM 389

17.1 FORM	389
17.1.1 FORM 태그	389
17.1.2 Action	390
17.1.3 서밋	393
17.1.4 리셋	394
17.2 요소	394
17.2.1 input 텍스트	395
17.2.2 hidden	398
17.2.3 textarea	398
17.2.4 select	400
17.2.5 checkbox	403
17.2.6 radio box	406
17.2.7 file	407
17.3 다운로드	410

Chapter 18 스크립트 417

18.1 종료	417
18.2 코드 실행	419
18.3 접속 상태	420
18.3.1 connection_status	420
18.3.2 ignore_user_abort	421
18.3.3 connection_aborted	422
18.4 지연 함수, 실행 시간 설정	423
18.4.1 지연	423
18.4.2 실행 시간	424
18.5 스크립트 정보	426

Chapter 19 정보 429

19.1 시스템	429
19.2 정보 함수	432
19.3 ini	434
19.4 프로세스	438
19.5 리소스	441
19.6 include	443
19.7 extention	445
19.8 그 외	447

Chapter 20 오토로드 451

20.1 클래스 의존성	452
20.2 클래스 의존성 체크 함수	453
20.3 클래스 파일 삽입	453
20.4 PSR-4 Autoloading	456
20.5 컴포저	457
20.5.1 컴포저 설치	457

20.5.2 구성 설정.....	457
20.5.3 컴포넌트.....	458
20.5.4 패키지스트.....	459
20.5.5 패키지 다운로드.....	459
20.5.6 사설 저장소.....	460
20.5.7 패키지 사용.....	461

찾아보기	462
------------	-----

01

필터

안정적인 프로그램의 동작을 위해서는 정확한 값의 데이터가 중요합니다. 잘못된 데이터들을 프로그램의 예상치 않은 오동작을 발생시킬 수 있습니다.

프로그램 코드를 작성하다 보면 데이터의 정확한 유효성을 체크하는 작업을 자주 합니다. 하지만 데이터를 일정한 패턴으로 검사하는 것은 그렇게 간단하지 않습니다. PHP는 데이터의 유효한 형식을 분석하고 패턴을 검사할 수 있는 다양한 필터 함수들을 제공합니다.

또한 웹 서비스 용도로 많이 사용하는 PHP 언어의 특성상 데이터는 외부 HTML 폼 입력을 통하여도 많이 입력됩니다. 불특정한 다수의 사람들로부터 데이터를 입력받을 때 이러한 데이터의 유효성과 패턴은 더욱더 중요합니다.

일부 악의적인 사용자가 잘못된 입력으로 프로그램의 오류를 발생시키거나, SQL 쿼리 명령들을 섞어 입력함으로써 데이터베이스의 내용을 해킹할 수도 있습니다.

PHP는 제공되는 필터 함수를 통하여 간단하게 데이터의 패턴을 검사하고 유효성을 확인할 수 있습니다. 필터 함수 기능들은 PHP 5.2.0부터는 기본적으로 설치되어 있습니다.

하지만 필터 기능과 동작은 php.ini의 환경 설정에 의해 약간의 영향을 받습니다. ini 설정은 다음과 같습니다.

filter.default

filter.default 환경 설정은 모든 \$_GET, \$_POST, \$_COOKIE, \$_REQUEST 및 \$_SERVER 데이터를 필터링합니다. 필터 이름을 기본적으로 사용합니다.

filter.default_flags

filter.default_flags 환경 설정은 기본 필터로 설정된 경우 적용할 필터를 지정합니다. 이전 버전과의 호환성을 위해 기본적으로 FILTER_FLAG_NO_ENCODE_QUOTES로 설정됩니다.

01.1 필터의 유용성

필터는 데이터의 값을 패턴 형태로 유효성 검사를 합니다. 필터 기능을 자주 적용하는 분야로는 웹에서 데이터 값을 입력받는 분야일 것입니다. 웹 서비스와 같은 응용 서비스는 다양한 타입의 수많은 외부 데이터를 입력받아 처리합니다.

- HTML 폼 입력값
- 쿠키 값
- API 데이터
- 서버 변수들
- 데이터베이스 결과값

위의 같은 값들은 외부로 입력받을 수 있는 데이터의 유형입니다. 이러한 외부 입력과 연관되는 데이터들을 항상 유효성 검사를 하여 사용하는 것을 권장합니다. 유효성 검사 없이 데이터를 바로 사용할 경우에는 악의적인 사용자로부터 입력받은 데이터 값들이 프로그램의 문제를 발생시키거나 해킹에 노출됩니다.

01.1.1 필터 목록

PHP의 필터들은 다양한 형태의 패턴들을 검사할 수 있습니다. 패턴 검사 유형은 필터 이름으로 구분합니다. 내장 함수 `filter_list()`는 PHP에서 지원하는 내장 필터 목록을 출력합니다.

| 내장 함수 |

```
array filter_list ( void )
```

내장 함수 `filter_list()`를 사용할 때는 전달되는 입력 매개변수는 없습니다. 내장된 필터의 목록들을 배열 형태로 반환합니다.

또한 내장된 필터명은 개발자들이 알기 쉽게 정의한 이름입니다. 이름들은 내부적으로 처리하는 고유의 ID 값들로 다시 매칭되어 있습니다. 내장 함수 `filter_id()`는 입력된 필터명에 대해서 고유의 상수 ID 값을 확인할 수 있습니다.

| 내장 함수 |

```
int filter_id ( string $filtername )
```

예제 파일 | **filter-01.php**

```
1  <?php
2      foreach (filter_list() as $id =>$filter) {
3          echo "필터명 =" . $filter . ', ID=' . filter_id($filter) . '<br>';
4      }
5
6  ?>
```

화면 출력

```
필터명 =int, ID=257
필터명 =boolean, ID=258
필터명 =float, ID=259
필터명 =validate_regexp, ID=272
```

```

필터명 =validate_domain, ID=277
필터명 =validate_url, ID=273
필터명 =validate_email, ID=274
필터명 =validate_ip, ID=275
필터명 =validate_mac, ID=276
필터명 =string, ID=513
필터명 =stripped, ID=513
필터명 =encoded, ID=514
필터명 =special_chars, ID=515
필터명 =full_special_chars, ID=522
필터명 =unsafe_raw, ID=516
필터명 =email, ID=517
필터명 =url, ID=518
필터명 =number_int, ID=519
필터명 =number_float, ID=520
필터명 =magic_quotes, ID=521
필터명 =callback, ID=1024

```

위의 실습 코드는 내장된 필터의 목록과 필터명에 대한 고유한 ID 값을 확인하는 예제입니다. 이처럼 다양한 형태의 데이터 유효성을 검사할 수 있는 필터들을 제공합니다.

01.1.2 유효성 검사

내장 함수 `filter_var()`는 입력된 데이터를 지정한 필터 형태의 패턴으로 데이터 유효성을 검사합니다.

| 내장 함수 |

```
mixed filter_var ( mixed $variable [, int $filter = FILTER_DEFAULT [, mixed $options ]]
```

함수의 매개변수 입력값으로 2개의 값을 전달받습니다. 첫 번째 데이터 값은 검사 유형입니다. `filter_var()` 함수는 지정된 필터로 단일 변수만 필터링합니다.

01.2 필터 상수

PHP는 `filter_var()` 함수를 통하여 필터링하기 위한 패턴 유형을 상수명으로 제공합니다. 알기 쉬운 패턴 코드들을 상수 이름으로 정의하여 제공하는 것이 코드를 작성하는 측면에서 매우 유용합니다.

01.2.1 정수 확인: `FILTER_VALIDATE_INT`

내부 필터 함수 `filter_var()`는 입력되는 데이터 값이 정수형 타입인지를 확인합니다. 함수를 호출할 때 두 번째 인자값으로 필터 유형 `FILTER_VALIDATE_INT`를 사용합니다. 값의 정수 여부 판단은 논리값으로 반환합니다.

예제 파일 | **filter-02.php**

```
1  <?php
2      $int = 100;
3
4      if (!filter_var($int, FILTER_VALIDATE_INT) === false) {
5          echo("정수가 유효합니다");
6      } else {
7          echo("정수가 아닙니다");
8      }
9
10 ?>
```

화면 출력

정수가 유효합니다

위의 실험은 입력 변수 `$int`의 데이터 값이 정수값인지를 필터를 통하여 확인합니다. 만일 정수의 값이면 `true` 반환값을 출력합니다.

수학에서 정수는 양수, 0, 음수 세 가지로 구분합니다. `filter_var()`는 양수와 음수만 정수 값을 판별할 수 있습니다. 즉, 0의 값은 정수로 인식하지 못합니다. 따라서 이와 관련된 부가적인 코드를 함께 작성해야 합니다. 정수값 필터 사용 시 `=== 0` 조건과 `||` 연결을 결합하여 사용하는 것을 권장합니다.

예제 파일 | **filter-03.php**

```
1  <?php
2      $int = 0;
3
4      if (filter_var($int, FILTER_VALIDATE_INT) === 0 ||
5          !filter_var($int, FILTER_VALIDATE_INT) === false) {
6          echo("Integer is valid");
7      } else {
8          echo("Integer is not valid");
9      }
10
11  ?>
```

화면 출력

Integer is valid

위 실험은 0의 정수값을 같이 판별을 하는 예제 코드입니다.

01.2.2 정수 확인 및 범위

컴퓨터에서 정수값을 사용할 때 표현할 수 있는 최대치의 범위가 있습니다. 또한 프로그램에서 입력의 임계치 등의 범위가 필요할 경우가 있습니다. 이러한 경우에는 `FILTER_VALIDATE_INT` 값과 같이 세 번째 매개변수를 통하여 범위 값을 함께 전달할 수 있습니다.

배열값을 응용하여 정수값 판별과 값의 범위 여부 내에 있는지 확인할 수 있습니다.

예제 파일 | **filter-04.php**

```
1  <?php
2      $int = 122;
3      $min = 1;
4      $max = 200;
5
6      $range = array("options" => array("min_range"=>$min, "max_
7          range"=>$max);
```

```

8     if (filter_var($int, FILTER_VALIDATE_INT, $range)) === false) {
9         echo("변수 값이 유효한 범위 내에 있지 않습니다!");
10    } else {
11        echo("변수 값이 유효한 범위 내에 있습니다!");
12    }
13
14    ?>

```

화면 출력

변수 값이 유효한 범위 내에 있습니다!

위의 실험은 필터 함수 `filter_var()`를 사용하여 입력되는 데이터가 정수 유형인지와 1에서 200 사이의 값을 갖는지를 확인합니다.

그 외 참고하여 같이 사용할 수 있는 필터 상수는 다음과 같습니다.

- `FILTER_VALIDATE_BOOLEAN`: 논리값을 검사합니다.
- `FILTER_VALIDATE_FLOAT`: float의 유효성을 검사합니다.
- `FILTER_SANITIZE_NUMBER_FLOAT`: 숫자, + - 및 .e를 제외한 모든 문자를 제거합니다.
- `FILTER_SANITIZE_NUMBER_INT`: 숫자와 + -를 제외한 모든 문자를 제거합니다.

01.2.3 IP 주소 유효성: `FILTER_VALIDATE_IP`

필터 함수 `filter_var()`는 인터넷 IP 주소의 유형을 검사할 수 있습니다. 필터 상수명으로 `FILTER_VALIDATE_IP`를 사용하게 되면 입력한 데이터가 IP 주소인지를 검사할 수 있습니다.

예제 파일 | `filter-05.php`

```

1  <?php
2      $ip = "127.0.0.1";
3
4      if (!filter_var($ip, FILTER_VALIDATE_IP) === false) {
5          echo("$ip 는 유효한 IP 주소입니다.");
6      }
7  }

```

```

6     } else {
7         echo("$ip 는 유효하지 않은 IP 주소입니다.");
8     }
9
10  ?>

```

화면 출력

127.0.0.1는 유효한 IP 주소입니다.

01.2.4 IPv6 주소 유효성

IP에 대해서 IPv6 주소 타입 및 유효성을 검사할 수 있습니다. IPv6 주소를 검사할 때는 FILTER_VALIDATE_IP와 FILTER_FLAG_IPV6 2개의 값을 함께 이용합니다.

예제 파일 | filter-06.php

```

1  <?php
2      $ip = "2001:0db8:85a3:08d3:1319:8a2e:0370:7334";
3
4      if (!filter_var($ip, FILTER_VALIDATE_IP, FILTER_FLAG_IPV6) === false) {
5          echo("$ip 는 유효한 IPv6 address");
6      } else {
7          echo("$ip 는 유효하지 않은 IPv6 address");
8      }
9
10  ?>

```

화면 출력

2001:0db8:85a3:08d3:1319:8a2e:0370:7334는 유효한 IPv6 address

01.2.5 쿼리스트링과 URL 유효성 검사: FILTER_VALIDATE_URL

검사 유형 FILTER_VALIDATE_URL과 FILTER_FLAG_QUERY_REQUIRED 2개의 값을 통하여 URL 유효성과 이후 입력되는 GET 쿼리스트링 데이터 값의 유효성까지 함께 확인할 수 있습니다.

예제 파일 | filter-07.php

```
1 <?php
2     $url = "http://www.hojin.io";
3
4     if (!filter_var($url, FILTER_VALIDATE_URL, FILTER_FLAG_QUERY_
5         REQUIRED) === false) {
6         echo("$url 유효한 URL");
7     } else {
8         echo("$url 유효하지 않은 URL");
9     }
10 ?>
```

화면 출력

http://www.hojin.io 유효하지 않은 URL

다음 실험은 filter_var() 함수를 사용하여 문자열을 삭제합니다. 다음과 같이 문자열에서 모든 HTML 태그와 ASCII 값이 127보다 큰 모든 문자를 제거합니다.

예제 파일 | filter-08.php

```
1 <?php
2     $str = "<h1>Hello World!</h1>";
3
4     $aaa = filter_var($str, FILTER_SANITIZE_STRING, FILTER_FLAG_STRIP_
5         HIGH);
6     echo $aaa;
7 ?>
```

화면 출력

Hello World!

01.2.6 이메일 주소 검사

내장 함수 filter_var()를 통하여 이메일의 유효성을 검사할 수 있습니다. 이메일의 유효성을 검사할 때는 필터 상수 FILTER_VALIDATE_EMAIL을 사용합니다.

예제 파일 | filter-09.php

```
1 <?php
2     $email = "infohojin@naver.com";
3
4     // 이메일에 포함된 부정한 글자들은 제거합니다.
5     $email = filter_var($email, FILTER_SANITIZE_EMAIL);
6
7     // e-mail 유효성
8     if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
9         echo("$email is a valid email address");
10    } else {
11        echo("$email is not a valid email address");
12    }
13
14 ?>
```

화면 출력

james.lee@example.com is a valid email address

그 외 참고하여 같이 사용할 수 있는 필터 상수는 다음과 같습니다.

- FILTER_SANITIZE_EMAIL: 부정한 글자를 제거
- FILTER_VALIDATE_EMAIL: 전자메일 주소 유효성 검사

01.2.7 문자열 체크: FILTER_SANITIZE_STRING

입력한 값을 재출력하는 기능에서 잘못 입력되는 html 코드들은 오동작을 발생하게 됩니다. 내장 함수 filter_var()를 통하여 입력된 문자열에서 HTML 코드를 제거할 수 있습니다.

예제 파일 | filter-10.php

```
1 <?php
2     $str = "<h1>Hello World!</h1>";
3     $newstr = filter_var($str, FILTER_SANITIZE_STRING);
4     echo $newstr;
5 ?>
```

화면 출력

Hello World!

필터 상수 `FILTER_SANITIZE_STRIPPED`은 `FILTER_SANITIZE_STRING`의 별칭입니다.

01.2.8 URL 유효성 검사

필터 상수 `FILTER_SANITIZE_URL`은 입력값의 URL 유효성 검사를 확인할 수 있습니다.

예제 파일 | `filter-11.php`

```
1  <?php
2      $url = "https://www.hojin.io";
3
4      // Remove all illegal characters from a url
5      $url = filter_var($url, FILTER_SANITIZE_URL);
6
7      // Validate url
8      if (!filter_var($url, FILTER_VALIDATE_URL) === false) {
9          echo("$url is a valid URL");
10     } else {
11         echo("$url is not a valid URL");
12     }
13
14  ?>
```

화면 출력

`https://www. hojin.io` is a valid URL

01.2.9 정규표현

필터 상수 `FILTER_VALIDATE_REGEXP`는 입력된 데이터의 정규표현식 유효성을 검사합니다.

01.2.10 특수문자

다음과 같은 필터 상수를 통하여 입력되는 데이터의 특수문자들을 처리할 수 있습니다.

- FILTER_SANITIZE_ENCODED: 특수문자를 제거/인코딩합니다.
- FILTER_SANITIZE_SPECIAL_CHARS: 특수문자를 제거합니다.
- FILTER_UNSAFE_RAW: 별도의 작업 없이 선택적으로 특수문자만 제거/인코딩합니다.
- FILTER_SANITIZE_FULL_SPECIAL_CHARS

표현 문자	숫자 표현	문자 표현	설명
-	�-	-	사용하지 않음
space			-	수평 탭
space	
	-	줄 삽입
-	-	-	사용하지 않음
space	 	-	여백
!	!	-	느낌표
"	"	";	따옴표
#	#	-	숫자 기호
\$	$	-	달러
%	%	-	백분율 기호
&	&	&;	Ampersand
'	'	-	작은따옴표
((-	왼쪽 괄호
))	-	오른쪽 괄호
*	*	-	아스트릭
+	+	-	더하기 기호
,	,	-	쉼표
-	-	-	Hyphen
.	.	-	마침표
/	/	-	Solidus (slash)

표현 문자	숫자 표현	문자 표현	설명
0 - 9	0-9	—	0부터 9까지
:	:	—	콜론
;	;	—	세미콜론
<	<	<	보다 작은
=	=	—	등호
>	>	>	보다 큰
?	?	—	물음표
@	@	—	Commercial at
A - Z	A-Z	—	A부터 Z까지
[[—	왼쪽 대괄호
₩	\	—	백슬래시
]]	—	오른쪽 대괄호
^	^	—	탈자 부호
_	_	—	수평선
'	`	—	Acute accent
a - z	a-z	—	a부터 z까지
{	{	—	왼쪽 중괄호
	|	—	수직선
}	}	—	오른쪽 중괄호
~	~	—	꼬리표
—	-Ÿ	—	사용하지 않음
	 	 	Non-breaking space
¡	¡	¡	거꾸로된 느낌표
¢	¢	¢	센트 기호
£	£	£	파운드
¤	¤	¤	현재 환율
¥	¥	¥	엔
	¦	¦	끊어진 수직선
§	§	§	섹션 기호

표현 문자	숫자 표현	문자 표현	설명
¨	¨	¨	움라우트
©	©	©	저작권
a	ª	ª	Feminine ordinal
«	«	«	왼쪽 꺾인 괄호
¬	¬	¬	부정
	­	­	Soft hyphen
®	®	®	등록상표
̄	¯	¯	Macron accent
°	°	°	Degree sign
±	±	±	Plus or minus
²	²	²	Superscript two
³	³	³	Superscript three
´	´	´	Acute accent
μ	µ	µ	Micro sign (Mu)
¶	¶	¶	문단 기호
•	·	·	Middle dot
¸	¸	¸	Cedilla
¹	¹	¹	Superscript one
o	º	º	Masculine ordinal
»	»	»	오른쪽 꺾인 괄호
¼	¼	¼	4분의 1
½	½	½	2분의 1
¾	¾	¾	4분의 3
¿	¿	¿	거꾸로된 물음표
À	À	À	Capital A, grave accent
Á	Á	Á	Capital A, acute accent
Â	Â	Â	Capital A, circumflex accent
Ã	Ã	Ã	Capital A, tilde
Ä	Ä	Ä	Capital A, dieresis or umlaut mark

표현 문자	숫자 표현	문자 표현	설명
Å	Å	Å	Capital A, ring (Angstrom)
Æ	Æ	Æ	Capital AE diphthong (ligature)
Ç	Ç	Ç	Capital C, cedilla
É	È	È	Capital E, grave accent
Ê	É	É	Capital E, acute accent
Ê	Ê	Ê	Capital E, circumflex accent
Ë	Ë	Ü	Capital E, dieresis or umlaut mark
Ì	Ì	Ì	Capital I, grave accent
Í	Í	Í	Capital I, acute accent
Î	Î	Î	Capital I, circumflex accent
Ï	Ï	Ï	Capital I, dieresis or umlaut mark
Ð	Ð	Ð	Capital Eth, Icelandic
Ñ	Ñ	Ñ	Capital N, tilde
Ó	Ò	Ò	Capital O, grave accent
Ô	Ó	Ó	Capital O, acute accent
Õ	Ô	Ô	Capital O, circumflex accent
Ö	Õ	Õ	Capital O, tilde
Ö	Ö	Ö	Capital O, dieresis or umlaut mark
×	×	×	Multiply sign
Ø	Ø	Ø	width="130"Capital O, slash
Ù	Ù	Ù	Capital U, grave accent
Ú	Ú	Ú	Capital U, acute accent
Û	Û	Û	Capital U, circumflex accent
Ü	Ü	Ü	Capital U, dieresis or umlaut mark
Ý	Ý	Ý	Capital Y, acute accent
Þ	Þ	Þ	Capital Thorn, Icelandic
ß	ß	ß	Small sharp s, German (sz ligature)
À	à	à	Small a, grave accent
Á	á	á	Small a, acute accent

표현 문자	숫자 표현	문자 표현	설명
a	â	â	Small a, circumflex accent
a	ã	ã	Small a, tilde
a	ä	ä	Small a, dieresis or umlaut mark
a	å	å	Small a, ring
æ	æ	æ	Small æ diphthong (ligature)
c	ç	ç	Small c, cedilla
e	è	è	Small e, grave accent
e	é	é	Small e, acute accent
e	ê	ê	Small e, circumflex accent
e	ë	ë	Small e, dieresis or umlaut mark
i	ì	ì	Small i, grave accent
i	í	í	Small i, acute accent
i	î	î	Small i, circumflex accent
i	ï	ï	Small i, dieresis or umlaut mark
ð	ð	ð	Small eth, Icelandic
n	ñ	ñ	Small n, tilde
o	ò	ò	Small o, grave accent
o	ó	ó	Small o, acute accent
o	ô	ô	Small o, circumflex accent
o	õ	õ	Small o, tilde
o	ö	ö	Small o, dieresis or umlaut mark
÷	÷	÷	Division sign
ø	ø	ø	Small o, slash
u	ù	ù	Small u, grave accent
u	ú	ú	Small u, acute accent
u	û	û	Small u, circumflex accent
u	ü	ü	Small u, dieresis or umlaut mark
y	ý	ý	Small y, acute accent
þ	þ	þ	Small thorn, Icelandic

표현 문자	숫자 표현	문자 표현	설명
y	ÿ	ÿ	Small y, dieresis or umlaut mark

01.2.11 함수 적용

다음과 같은 필터 상수들은 함수를 별도의 함수를 적용하거나 호출할 수 있습니다.

- FILTER_SANITIZE_MAGIC_QUOTES: addslashes() 함수를 적용합니다.
- FILTER_CALLBACK: 데이터 필터링을 위한 사용자 정의 함수를 호출합니다.

01.3 필터 함수

PHP는 필터 기능을 좀 더 다양하게 처리하기 위해서 추가로 몇 가지 내장 함수들을 제공합니다. 추가 함수들을 이용하면 복잡한 필터 처리를 보다 쉽게 구현할 수 있습니다.

| 내장 함수 |

```
bool filter_has_var ( int $type , string $variable_name )
```

내장 함수 filter_has_var()는 지정된 유형의 변수가 있는지 확인합니다. 타입의 종류는 다음과 같습니다.

- INPUT_GET
- INPUT_POST
- INPUT_COOKIE
- INPUT_SERVER 또는 INPUT_ENV

예제 파일 | [filter-12.php](#)

```
1 <?php
```

```

2     echo "INPUT_GET = ";
3     echo filter_has_var(INPUT_GET, 'aaa') ? 'Yes' : 'No';
4
5     ?>

```

화면 출력

INPUT_GET = Yes

위의 실험은 타입 유형의 변수 존재 여부를 확인합니다. filter-12.php?aaa=test 형태로 GET 데이터를 추가하여 스크립트를 호출합니다. GET 방식으로 데이터를 전달과 변수명이 aaa이기 때문에 filter_has_var() 함수는 결과를 '참'으로 출력합니다.

| 내장 함수 |

```

mixed filter_input ( int $type , string $variable_name [, int $filter = FILTER_DEFAULT [,
mixed $options ] ] )

```

내장 함수 filter_input()은 양식 입력 등 외부 변수를 읽어와서 선택적으로 필터링합니다.

예제 파일 | filter-13.php

```

1  <?php
2      $search_html = filter_input(INPUT_GET, 'search', FILTER_SANITIZE_
    SPECIAL_CHARS);
3      $search_url = filter_input(INPUT_GET, 'search', FILTER_SANITIZE_
    ENCODED);
4
5      echo "입력한 검색 키워드는 $search_html 입니다.<br>";
6      echo "<a href='?search=$search_url'>재검색</a>";
7
8      ?>

```

위의 실험은 filter-13.php?search=jiny 형태로 GET 데이터를 추가하여 스크립트를 호출합니다.

| 내장 함수 |

```
mixed filter_input_array ( int $type [, mixed $definition [, bool $add_empty = true ]])
```

내장 함수 `filter_input_array()`는 필터를 처리하기 위한 배열 정보를 이용하여 다수의 외부 변수를 일괄적으로 필터링합니다. 이 함수는 `filter_input()`을 반복적으로 호출하지 않고도 많은 값을 검색할 때 유용합니다.

예제 파일 | **filter-14.php**

```
1  <?php
2      error_reporting(E_ALL | E_STRICT);
3      $args = array(
4          'name'    => FILTER_SANITIZE_ENCODED,
5          'age'     => FILTER_VALIDATE_INT,
6          'country' => FILTER_SANITIZE_ENCODED
7      );
8      $myinputs = filter_input_array(INPUT_GET, $args);
9      var_dump($myinputs);
10
11  ?>
```

화면 출력

```
array(3) { ["name"]=> string(4) "jiny" ["age"]=> int(18) ["country"]=>
string(5) "korea" }
```

위의 실험은 `filter-14.php?name=jiny&age=18&country=korea` 형태로 GET 데이터를 추가하여 스크립트를 호출합니다.

| 내장 함수 |

```
mixed filter_var_array ( array $data [, mixed $definition [, bool $add_empty = true ]])
```

내부 변수 `filter_var_array()`는 여러 개의 변수를 필터링합니다. 이 함수는 `filter_var()`를 반복적으로 호출하지 않고도 많은 값을 검색하는 데 유용합니다. 다음 예제는 공식 사이

트에 있는 예제입니다.

예제 파일 | **filter-15.php**

```
1  <?php
2      error_reporting(E_ALL | E_STRICT);
3      $data = array(
4          'product_id'  => 'libgd<script>',
5          'component'   => '10',
6          'versions'    => '2.0.33',
7          'testscalar'  => array('2', '23', '10', '12'),
8          'testarray'   => '2',
9      );
10
11     $args = array(
12         'product_id'  => FILTER_SANITIZE_ENCODED,
13         'component'   => array('filter'  => FILTER_VALIDATE_INT,
14                                'flags'    => FILTER_FORCE_ARRAY,
15                                'options'  => array('min_range' => 1, 'max_
16                                                    range' => 10)
17                                ),
18         'versions'    => FILTER_SANITIZE_ENCODED,
19         'doesnotexist' => FILTER_VALIDATE_INT,
20         'testscalar'  => array(
21             'filter' => FILTER_VALIDATE_INT,
22             'flags'  => FILTER_REQUIRE_SCALAR,
23         ),
24         'testarray'   => array(
25             'filter' => FILTER_VALIDATE_INT,
26             'flags'  => FILTER_FORCE_ARRAY,
27         )
28     );
29
30     $myinputs = filter_var_array($data, $args);
31
32     var_dump($myinputs);
33     echo "\n";
34
35  ?>
```

화면 출력

```
array(6) { ["product_id"]=> string(17) "libgd%3Cscript%3E" ["component"]=>
array(1) { [0]=> int(10) } ["versions"]=> string(6) "2.0.33" ["doesnotexist"]=>
NULL ["testscalar"]=> bool(false) ["testarray"]=> array(1) { [0]=> int(2) } }
```

02

배열

배열은 여러 개의 값을 가지는 유형의 데이터 타입입니다. 여러 개의 값을 가지고 있는 배열의 특성상 값의 정렬, 분리, 결합 등의 추가적인 작업이 가능합니다.

PHP는 배열 데이터에 대해서 추가 작업을 할 수 있도록 내장 함수를 제공합니다.

02.1 배열 생성

배열 변수를 생성할 수 있습니다. 또한 배열을 여러 개의 변수로 분할하여 처리할 수도 있습니다. 배열을 변수화하여 데이터를 처리하는 것은 배열을 활용하는 데 매우 유용합니다.

02.1.1 변수 할당

배열 변수를 생성하기 위해서는 데이터의 값을 배열 타입으로 저장하면 됩니다. 배열 타입의 값을 초기에 생성하기 위해서는 `array()` 함수를 이용합니다.

| 내장 함수 | 배열 생성

```
array array ([ mixed $... ] )
```

내장 함수 `array()`는 입력된 값에 대한 배열 데이터를 반환합니다. 각각의 데이터는 콤마 (,)로 구분하며 키를 삽입할 때는 => 기호를 사용합니다.

예제 파일 | `array-01.php`

```
1  <?php
2      // 배열 변수를 생성합니다.
3      // 입력한 값에 대한 배열값을 반환합니다.
4      $arr1 = array("hojin","jiny","james","eric");
5
6      // 배열값을 foreach를 통하여 하나씩 출력합니다.
7      foreach ($arr1 as $value) {
8          echo "$value <br>";
9      }
10
11     echo "==== <br>";
12
13     // 키 형태의 배열을 지정합니다.
14     $arr2 = array('firstname' => "hojin", 'lastname' => "lee" );
15     foreach ($arr2 as $key => $value) {
16         echo "{$key} => {$value} ";
17         print_r($arr);
18     }
19
20  ?>
```

화면 출력

```
hojin
jiny
james
eric
====
firstname => hojin lastname => lee
```

배열은 값의 묶음 처리입니다. 묶음 형태의 배열을 처리하는 것은 유사한 데이터를 처리하는 데 있어서 많은 변수의 이름을 통하여 자원을 낭비하는 것을 줄이기 위해서입니다.

| 내장 함수 | 배열의 변수화

```
array list ( mixed $var1 [, mixed $... ] )
```

하지만 반대로 배열을 여러 개의 변수로 분할할 수도 있습니다. 내장 함수 `list()`는 배열의 값을 지정한 각각의 변수에 값을 할당할 수 있습니다. 배열의 데이터를 여러 개의 변수에 분배하여 저장할 때 편리합니다.

예제 파일 | array-02.php

```
1  <?php
2      // 배열을 생성합니다.
3      $arr = array('orange', 'apple', 'grape');
4
5      // 배열의 값을 각각의 변수에 값을 매칭하여 대입합니다.
6      list($orange, $apple, $grape) = $arr;
7
8      echo "변수 orange = $orange <br>";
9      echo "변수 apple = $apple <br>";
10     echo "변수 grape = $grape <br>";
11
12     echo "<br> ===== <br><br>";
13
14     while (list($key,$val) = each($arr)) {
15         echo $key. "==" . $val. "<br>";
16     }
17
18  ?>
```

화면 출력

```
변수 orange = orange
변수 apple = apple
변수 grape = grape
=====
```



```
0==>orange
1==>apple
2==>grape
```

| 내장 함수 |

```
array array_fill ( int $start_index , int $num , mixed $value )
```

내장 함수 `array_fill()`은 지정한 값으로 배열을 채웁니다. 지정한 값을 배열의 시작 위치 (`start_index`)부터 반복 횟수(`num`)만큼 반복하여 배열을 생성합니다.

예제 파일 | **array-03.php**

```
1 <?php
2     $a = array_fill(5, 6, 'aaa');
3     $b = array_fill(-2, 4, 'bbb');
4     print_r($a);
5     print_r($b);
6
7 ?>
```

화면 출력

```
Array ( [5] => aaa [6] => aaa [7] => aaa [8] => aaa [9] => aaa [10] => aaa )
Array ( [-2] => bbb [0] => bbb [1] => bbb [2] => bbb )
```

| 내장 함수 |

```
array array_fill_keys ( array $keys , mixed $value )
```

내장 함수 `array_fill_keys()`는 입력받은 배열의 값을 키 값으로 사용하여 배열의 데이터를 채웁니다.

예제 파일 | **array-04.php**

```
1 <?php
2     $keys = array('foo', 5, 10, 'bar');
```

```

3     $a = array_fill_keys($keys, 'banana');
4     print_r($a);
5
6     ?>

```

화면 출력

Array ([foo] => banana [5] => banana [10] => banana [bar] => banana)

02.1.2 값과 키

PHP는 다양한 형태의 최신 배열 표기 및 처리를 지원합니다. 일반 배열과 키 값을 가지고 있는 연상 배열 모두 사용할 수 있습니다. 내장 함수 `each()` 함수는 배열의 키와 값을 한 쌍으로 반환합니다. `list()` 함수와 결합하여 배열의 키와 값을 추출할 수 있습니다.

| 내장 함수 | 배열의 쌍 값

`array each (array &$array)`

배열이 담긴 `each()` 함수를 호출할 때마다 다음 배열 위치로 전달됩니다.

예제 파일 | array-05.php

```

1  <?php
2      // 배열을 생성합니다.
3      $arr = array('orange', 'apple', 'grape');
4
5      $test = each($arr);
6      echo "원소 0= ".$test[0]."<br>";
7      echo "원소 1= ".$test[1]."<br>";
8      echo "원소 key= ".$test[key]."<br>";
9      echo "원소 value= ".$test[value]."<br>";
10
11     echo "<br><br>";
12
13     while (list($key,$val) = each($arr)) {
14         echo $key. "=="> . $val. "<br>";

```

```

15     }
16
17  ?>

```

화면 출력

```

원소 0= 0
원소 1= orage
원소 key= 0
원소 value= orage

```

```

1==>apple
2==>grape

```

위의 실습에서 배열의 키와 값을 추출해 봅니다. 첫 번째 명령에서 1번, 반복문에서 2번 실행되어 총 3개의 배열 정보가 출력됩니다. `each()` 함수는 네 가지의 키 값으로 데이터를 반환하는데, 0, key, 1, value입니다.

연상 배열에서 값만 추출한 다른 배열을 생성할 수 있습니다. 내장 함수 `array_values()`는 배열의 값을 반환합니다.

| 내장 함수 | 배열값

```
array array_values ( array $array )
```

예제 파일 | array-06.php

```

1  <?php
2      // 배열을 생성합니다.
3      $arr = array('firstname' => 'hojin', 'lastname' => 'lee' );
4      while (list($key,$val) = each($arr)) {
5          echo $key. "==" . $val. ", ";
6      }
7      echo "<br>====<br>";
8
9      $arr2 = array_values($arr);
10     while (list($key,$val) = each($arr2)) {
11         echo $key. "==" . $val. ", ";

```

```

12     }
13
14     ?>

```

화면 출력

```

firstname==>hojin, lastname==>lee,
====
0==>hojin, 1==>lee,

```

| 내장 함수 | 배열 인덱스 키

```
array array_keys ( array $array [, mixed $search_value = null [, bool $strict = false ] ] )
```

내장 함수 `array_keys()`는 배열의 인덱스 키 이름만을 반환받을 수 있습니다.

예제 파일 | array-07.php

```

1  <?php
2      $arr = array('firstname' => "hojin", 'lastname' => "lee" );
3      $index = array_keys($arr);
4
5      while (list($key,$val) = each($index)) {
6          echo $key. "==" . $val. "<br>";
7      }
8
9      ?>

```

화면 출력

```

0==>firstname
1==>lastname

```

| 내장 함수 |

```
array array_column ( array $input , mixed $column_key [, mixed $index_key = null ] )
```

내장 함수 `array_column()`은 입력된 다중 배열에서 단일 열에서 값만을 추출하여 배열로 반환합니다.

예제 파일 | **array-08.php**

```
1  <?php
2      $records = array(
3          array(
4              'id' => 2135,
5              'first_name' => '홍',
6              'last_name' => '길동',
7          ),
8          array(
9              'id' => 3245,
10             'first_name' => '이',
11             'last_name' => '호진',
12          ),
13          array(
14              'id' => 5342,
15              'first_name' => '장',
16              'last_name' => '승빈',
17          )
18      );
19
20      $first_names = array_column($records, 'last_name');
21      print_r($first_names);
22
23  ?>
```

화면 출력

Array ([0] => 길동 [1] => 호진 [2] => 승빈)

| 내장 함수 |

array array_flip (array \$array)

내장 함수 `array_flip()`은 배열의 키와 값을 서로 교환합니다. `array_flip()`은 배열의 키가 값이 됩니다. 값은 다시 배열의 키로 변경하여 배열로 처리합니다.

예제 파일 | [array-09.php](#)

```
1 <?php
2     $input = array("a"=>"oranges", "b"=>"apples", "c"=>"pears");
3     $flipped = array_flip($input);
4
5     print_r($flipped);
6
7 ?>
```

화면 출력

Array ([oranges] => a [apples] => b [pears] => c)

02.1.3 변수 결합과 분리

내장 함수 `compact()`는 외부의 변수명을 응용하여 배열을 생성할 수 있습니다. 변수명은 키 값으로 사용됩니다.

| 내장 함수 | [변수 배열 생성](#)

`array compact (mixed $varname1 [, mixed $...])`

예제 파일 | [array-10.php](#)

```
1 <?php
2
3     $one = "1개";
4     $two = "2개";
5     $three = "3개";
6     $four = "4개";
7
8     // 배열에 변수의 값을 넣기 위해서는 ""로 변수명을 입력하면 됩니다.
9     $num = array("three", "four");
10
11    // 배열을 넣을 때는 배열명만 인수로 전달합니다.
12    $arr = compact("one", "two", $num);
13
```

```

14     while (list($key,$val) = each($arr)) {
15         echo $key. "==>" . $val. "<br>";
16     }
17
18     ?>

```

화면 출력

```

one==>1개
two==>2개
three==>3개
four==>4개

```

| 내장 함수 | 배열 변수화

```
int extract ( array &$array [, int $flags = EXTR_OVERWRITE [, string $prefix = NULL ]])
```

내장 함수 `extract()`는 배열의 키 이름으로 각각 변수화합니다. `compact()` 함수의 반대 기능입니다.

예제 파일 | array-11.php

```

1  <?php
2      $arr = array("blue"=>"파랑색","red"=>"빨강색","black"=>"검정색");
3
4      // 배열을 변수로 생성합니다.
5      extract($arr);
6
7      echo "blue = ".$blue."<br>";
8      echo "red = ".$red."<br>";
9      echo "black = ".$black."<br>";
10
11  ?>

```

화면 출력

```

blue = 파랑색
red = 빨강색
black = 검정색

```

02.1.4 결합 및 분리

내장 함수 `array_merge()`는 입력되는 배열을 하나의 배열 형태로 결합하여 반환합니다.

| 내장 함수 | 배열 결합

```
array array_merge ( array $array1 [, array $... ] )
```

예제 파일 | array-12.php

```
1  <?php
2      $arr1 = array('지니', '호진');
3      $arr2 = array('철수', '영수');
4      $arr3 = array('제니퍼', '홍길동');
5
6      $arr = array_merge($arr1, $arr2, $arr3);
7
8      while (list($key,$val) = each($arr)) {
9          echo $key. "==>" . $val. "<br>";
10     }
11
12  ?>
```

화면 출력

```
0==>지니
1==>호진
2==>철수
3==>영수
4==>제니퍼
5==>홍길동
```

| 내장 함수 | 재귀적 배열 결합

```
array array_merge_recursive ( array $array1 [, array $... ] )
```

2개 이상의 배열을 재귀적으로 병합합니다. 내장 함수 `array_merge_recursive()`는 다수

의 배열 요소를 병합하여 배열의 끝에 추가합니다.

병합될 배열에 동일한 문자열 키가 있는 경우 이러한 키의 값은 배열로 병합됩니다. 이는 재귀적으로 수행되므로 값 중 하나가 배열 자체이면 함수는 해당 항목과 병합합니다. 그리고 배열에 동일한 숫자 키가 있으면 나중에 값이 원래 값을 덮어쓰지 않고 뒤에 추가됩니다.

예제 파일 | array-13.php

```
1 <?php
2 $ar1 = array("color" => array("favorite" => "red"), 5);
3 $ar2 = array(10, "color" => array("favorite" => "green", "blue"));
4 $result = array_merge_recursive($ar1, $ar2);
5
6 print_r($result);
7
8 ?>
```

화면 출력

```
Array ( [color] => Array ( [favorite] => Array ( [0] => red [1] => green ) [0]
=> blue ) [0] => 5 [1] => 10 )
```

| 내장 함수 | 배열 결합 (키/값)

array **array_combine** (array \$keys , array \$values)

내장 함수 array_combine()은 첫 번째 배열을 키 값으로, 두 번째 배열을 값 형태로 배열을 결합합니다.

예제 파일 | array-14.php

```
1 <?php
2 $a = array('green', 'red', 'yellow');
3 $b = array('avocado', 'apple', 'banana');
4
5 // $a 배열은 키명으로 사용하고, $b는 데이터로 사용합니다.
6 // 2개의 배열을 연상 배열 형태로 결합합니다.
```

```

7     $arr = array_combine($a, $b);
8     print_r($arr);
9
10    ?>

```

화면 출력

Array ([green] => avocado [red] => apple [yellow] => banana)

| 내장 함수 | 일부 배열 분리

```
array array_slice ( array $array , int $offset [, int $length = NULL [, bool $preserve_
keys = false ]])
```

내장 함수 array_slice()는 배열의 일부분의 요소만을 추출하여 반환합니다.

예제 파일 | array-15.php

```

1  <?php
2      // 배열을 생성합니다.
3      $arr = array('하나', '둘', '셋', '넷', '다섯', '여섯', '일곱', '여덟', '아홉',
        '열');
4
5      // 첫 번째 인자: 추출 시작 값
6      // 처음 2부터 끝까지.
7      $slice = array_slice($arr, 2);
8      while (list($key,$val) = each($slice)) {
9          echo $key. "==>" . $val. " , ";
10     }
11
12     echo "<br>";
13
14     // 두 번째 인자: 추출 데이터 개수
15     // 처음 1부터, 3개의 데이터를 추출
16     $slice = array_slice($arr, 1,3);
17     while (list($key,$val) = each($slice)) {
18         echo $key. "==>" . $val. " , ";
19     }
20

```

```

21     echo "<br>";
22
23     // 처음 인자가 음수이면 끝에서 -인자부터 시작하여, 두 번째 인자 개수를 추출
24     // 마지막 -5 위치부터(여섯), 3개의 데이터를 추출
25     $slice = array_slice($arr, -5, 3);
26     while (list($key, $val) = each($slice)) {
27         echo $key. "==" . $val. ", ";
28     }
29
30     ?>

```

화면 출력

```

0==>셋, 1==>넷, 2==>다섯, 3==>여섯, 4==>일곱, 5==>여덟, 6==>아홉, 7==>열,
0==>둘, 1==>셋, 2==>넷,
0==>여섯, 1==>일곱, 2==>여덟,

```

| 내장 함수 |

```
array array_chunk ( array $array , int $size [, bool $preserve_keys = false ] )
```

내장 함수 `array_chunk()`는 배열을 크기 요소가 있는 배열로 분할합니다. 마지막 chunk는 크기 요소보다 작은 요소를 포함할 수 있습니다.

예제 파일 | [array-16.php](#)

```

1  <?php
2      $array = array('a', 'b', 'c', 'd', 'e');
3      print_r(array_chunk($array, 2));
4      echo "<br>";
5
6      print_r(array_chunk($array, 3));
7      echo "<br>";
8
9      ?>

```

화면 출력

```

Array ( [0] => Array ( [0] => a [1] => b ) [1] => Array ( [0] => c [1] => d ) [2]
=> Array ( [0] => e ) )

```

```
Array ( [0] => Array ( [0] => a [1] => b [2] => c ) [1] => Array ( [0] => d [1]
=> e ) )
```

02.1.5 출력

내장 함수 `array_reverse()`는 배열의 내용을 역순으로 출력합니다.

| 내장 함수 | 역순 배열

```
array array_reverse ( array $array [, bool $preserve_keys = false ] )
```

예제 파일 | [array-17.php](#)

```
1  <?php
2      $arr = array("hojin","jiny","james","eric");
3      $reverse = array_reverse($arr);
4
5      while (list($key,$val) = each($reverse)) {
6          echo $key. "==" . $val. "<br>";
7      }
8
9  ?>
```

화면 출력

```
0==>eric
1==>james
2==>jiny
3==>hojin
```

02.1.6 추가 및 삭제

내장 함수 `array_push()`는 배열의 마지막 위치에 새로운 원소 데이터를 추가합니다.

| 내장 함수 | 배열 원소 추가(마지막)

```
int array_push ( array &$array , mixed $value1 [, mixed $... ] )
```

예제 파일 | **array-18.php**

```
1  <?php
2  // 배열을 생성합니다.
3  $arr = array('orange', 'apple', 'grape');
4
5  // $arr 배열에 2개의 원소를 추가합니다.
6  array_push($arr, 'banana', 'tomato');
7
8  while (list($key,$val) = each($arr)) {
9      echo $key. "==" . $val. "<br>";
10 }
11
12 ?>
```

화면 출력

```
0==>orange
1==>apple
2==>grape
3==>banana
4==>tomato
```

| 내장 함수 | 배열 원소 제거

```
mixed array_pop ( array &$array )
```

내장 함수 `array_pop()`은 배열의 마지막 요소 하나를 제거합니다. 이는 데이터 처리 알고리즘 `pop`과 유사합니다.

예제 파일 | **array-19.php**

```
1  <?php
```

```

2 // 배열을 생성합니다.
3 $arr = array('orange', 'apple', 'grape');
4
5 // 마지막 배열 원소를 삭제합니다.
6 $temp = array_pop($arr);
7 echo "배열에서 마지막 $temp 요소를 제거합니다.<br> ";
8
9 while (list($key,$val) = each($arr)) {
10     echo $key. "=>" . $val. "<br>";
11 }
12
13 ?>

```

화면 출력

배열에서 마지막 grape 요소를 제거합니다.

0=>orange

1=>apple

| 내장 함수 | 배열 원소 추가(처음)

```
int array_unshift ( array &$array , mixed $value1 [, mixed $... ] )
```

내장 함수 array_unshift()는 배열 앞쪽에 새로운 원소를 추가합니다.

예제 파일 | array-20.php

```

1 <?php
2 // 배열을 생성합니다.
3 $arr = array('orange', 'apple', 'grape');
4
5 // $arr 배열에 2개의 원소를 추가합니다.
6 array_unshift($arr, 'banana', 'tomato');
7
8 while (list($key,$val) = each($arr)) {
9     echo $key. "=>" . $val. "<br>";
10 }
11
12 ?>

```

화면 출력

```
0==>banana
1==>tomato
2==>orange
3==>apple
4==>grape
```

| 내장 함수 | 배열 원소 제거(처음)

```
mixed array_shift ( array &$array )
```

내장 함수 `array_shift()`는 배열의 맨 처음 요소 하나를 제거합니다. 배열을 좌측으로 하나 밀어서 제거하는 형태입니다.

예제 파일 | **array-21.php**

```
1  <?php
2      // 배열을 생성합니다.
3      $arr = array('orange', 'apple', 'grape');
4
5      // 배열 원소를 삭제합니다.
6      $temp = array_shift($arr);
7      echo "배열에서 처음 $temp 요소 하나를 제거합니다.<br> ";
8
9      while (list($key,$val) = each($arr)) {
10         echo $key. "==" . $val. "<br>";
11     }
12
13  ?>
```

화면 출력

배열에서 처음 orange 요소 하나를 제거합니다.

```
0==>apple
1==>grape
```

| 내장 함수 | 일부 배열 삭제

```
array array_splice ( array &$input , int $offset [, int $length = count($input) [, mixed  
$replacement = array() ] ] )
```

내장 함수 `array_splice()`는 일부분의 데이터를 삭제한 후에 값을 반환합니다.

예제 파일 | array-22.php

```
1  <?php
2      // 배열을 생성합니다.
3      $arr = array('하나', '둘', '셋', '넷', '다섯', '여섯', '일곱', '여덟', '아홉',
4                  '열');
5      echo "원본<br>";
6      while (list($key,$val) = each($arr)) {
7          echo $key. "==">". $val. " , ";
8      }
9      echo "<br><br>";
10
11     // 첫 번째 인자: 삭제 시작 값
12     // 2까지의 원소를 삭제하고 반환
13     $slice = array_splice($arr, 7);
14     while (list($key,$val) = each($slice)) {
15         echo $key. "==">". $val. " , ";
16     }
17     echo "를 삭제하였습니다.<br>";
18     echo "<br>";
19
20     $slice = array_splice($arr, 3,3);
21     while (list($key,$val) = each($slice)) {
22         echo $key. "==">". $val. " , ";
23     }
24     echo "를 삭제하였습니다.<br>";
25     echo "<br>";
26  ?>
```

화면 출력

원본

0==>하나, 1==>둘, 2==>셋, 3==>넷, 4==>다섯, 5==>여섯, 6==>일곱, 7==>여덟, 8==>아홉,
9==>열,

0==>여덟, 1==>아홉, 2==>열, 를 삭제하였습니다.

0==>넷, 1==>다섯, 2==>여섯, 를 삭제하였습니다.

02.2 배열 검사

배열은 여러 개의 데이터를 포함하고 있습니다. 같은 유형의 데이터의 묶음을 처리하는 데 있어서 값을 검사하고 처리하는 데 매우 유용합니다.

PHP는 배열의 데이터를 검색하고 처리하는 몇 가지 내장 함수를 지원합니다.

02.2.1 검사

내장 함수 `in_array()`는 배열에 값이 존재하는지 검사할 수 있습니다. 값이 있는 경우에는 `true` 값을 반환합니다.

| 내장 함수 | 배열 검색

```
bool in_array ( mixed $needle , array $haystack [, bool $strict = FALSE ] )
```

예제 파일 | array-23.php

```
1  <?php
2      $arr = array("blue"=>"파랑색", "red"=>"빨강색", "black"=>"검정색");
3
4      if (in_array("파랑색", $arr)) {
5          echo "값이 존재합니다.";
6      } else {
7          echo "배열값이 없습니다.";
8      }
9
10  ?>
```

화면 출력

값이 존재합니다.

| 내장 함수 | 배열 인덱스 키 확인

```
bool array_key_exists ( mixed $key , array $array )
```

내장 함수 `array_key_exists()`는 연상 배열 안에 인덱스 키 값이 존재하는지를 확인합니다.

예제 파일 | array-24.php

```
1  <?php
2      $array = array('first' => null, 'second' => 4);
3
4      // returns true
5      if (array_key_exists('first', $array)) {
6          echo "배열에 키 값이 존재합니다.";
7      } else {
8          echo "키 값이 없습니다.";
9      }
10
11  ?>
```

화면 출력

배열에 키 값이 존재합니다.

| 내장 함수 |

```
mixed array_search ( mixed $needle , array $haystack [, bool $strict = false ] )
```

내장 함수 `array_search()`는 지정한 배열에서 값을 검색하여 일치하는 키 값을 반환합니다.

예제 파일 | **array-25.php**

```
1  <?php
2      $array = array(0 => 'blue', 1 => 'red', 2 => 'green', 3 => 'red');
3
4      $key = array_search('green', $array); // $key = 2;
5      print_r($key);
6      echo "<br>";
7
8      $key = array_search('red', $array);    // $key = 1;
9      print_r($key);
10
11  ?>
```

화면 출력

```
2
1
```

02.2.2 중복

내장 함수 `array_count_values()`는 배열 안에 들어 있는 동일한 값의 개수를 반환합니다.

| 내장 함수 | **중복값 체크**

array **array_count_values** (array \$array)

예제 파일 | **array-26.php**

```
1  <?php
2      $arr = array('orange', 'apple', 'grape', 'orange', 'apple', 'grape',
3                  'orange');
4
5      $arr_count = array_count_values($arr);
6      while (list($key,$val) = each($arr_count)) {
7          echo $key. "==" . $val. "<br>";
8      }
```

```
8
9  ?>
```

화면 출력

```
orange==>3
apple==>2
grape==>2
```

| 내장 함수 | 중복값 제거

```
array array_unique ( array $array [, int $sort_flags = SORT_STRING ] )
```

내장 함수 `array_unique()`는 입력된 배열값 중에서 중복된 내용은 제거합니다.

예제 파일 | array-27.php

```
1  <?php
2      $arr = array("a" => "green", "red", "b" => "green", "blue", "red");
3      $result = array_unique($arr);
4      print_r($result);
5
6  ?>
```

화면 출력

```
Array ( [a] => green [0] => red [1] => blue )
```

02.3 배열 위치

배열은 여러 개의 데이터의 값을 가지고 있습니다. 인덱스 또는 키를 통하여 데이터에 접근할 수 있습니다.

그 외에 배열의 위치 값을 이용하여 데이터에 접근 및 값을 읽어올 수 있습니다. PHP는 배열의 위치 값을 제어할 수 있는 몇 가지 내장 함수를 지원하고 있습니다.

02.3.1 위치 이동

| 내장 함수 | 배열 포인트 초기화

```
mixed reset ( array &$array )
```

내장 함수 `reset()`은 배열의 포인트를 초기화합니다. 배열의 처음 위치를 가리키게 됩니다.

| 내장 함수 | 배열 포인트 다음

```
mixed next ( array &$array )
```

내장 함수 `next()`는 배열의 다음 포인트로 이동합니다. 함수를 한 번 호출할 때마다 배열의 포인트를 다음으로 이동합니다.

| 내장 함수 | 배열 포인트 마지막

```
mixed end ( array &$array )
```

내장 함수 `end()`는 배열의 마지막 포인트로 이동합니다.

| 내장 함수 | 배열 포인트 이전

```
mixed prev ( array &$array )
```

내장 함수 `prev()`는 배열의 이전 포인트로 이동합니다. 함수를 한 번 호출할 때마다 배열의 포인트를 이전으로 이동합니다.

예제 파일 | [array-28.php](#)

```
1 <?php
```

```

2 // 배열을 생성합니다.
3 $arr = array('하나', '둘', '셋', '넷', '다섯', '여섯', '일곱', '여덟', '아홉',
  '열');
4 echo "원본<br>";
5 while (list($key,$val) = each($arr)) {
6     echo $key. "==" . $val. " , ";
7 }
8 echo "<br> ===== <br>";
9
10 // 배열의 포인터를 처음으로 이동합니다.
11 reset($arr);
12
13 // 다음 배열의 포인터를 이동합니다.
14 next($arr);
15 echo "현재 포인트 = " . current($arr) . "<br>";
16
17 // 다음 배열의 포인터를 이동합니다.
18 next($arr);
19 echo "현재 포인트 = " . pos($arr) . "<br>";
20
21 // 마지막 배열의 포인터를 이동합니다.
22 end($arr);
23 echo "현재 포인트 = " . current($arr) . "<br>";
24
25 // 이전 배열의 포인터를 이동합니다.
26 prev($arr);
27 echo "현재 포인트 = " . pos($arr) . "<br>";
28
29 ?>

```

화면 출력

원본

0==>하나, 1==>둘, 2==>셋, 3==>넷, 4==>다섯, 5==>여섯, 6==>일곱, 7==>여덟, 8==>아홉,
9==>열,

=====

현재 포인트 = 둘

현재 포인트 = 셋

현재 포인트 = 열

현재 포인트 = 아홉

위의 예제는 배열의 위치를 지정하는 함수들을 이용하여 배열의 데이터를 출력하는 예제입니다.

02.3.2 위치 확인

배열의 위치 함수를 이용하면 배열의 데이터를 가리키는 포인터를 변경하게 됩니다. 현재 가리키고 있는 값과 키를 알 수 있는 함수들도 함께 제공합니다.

| 내장 함수 | 배열 현재 위치

```
mixed current ( array &$array )
```

내장 함수 `current()`는 배열의 현재 위치의 값을 반환합니다. `pos()` 함수는 `current()` 함수의 또 다른 이름의 별칭입니다.

| 내장 함수 | 배열 포인트 키 값

```
mixed key ( array &$array )
```

내장 함수 `key()`는 현재의 배열 포인트의 키 값을 출력합니다.

예제 파일 | array-29.php

```
1  <?php
2      $arr = array( 'fruit1' => 'apple', 'fruit2' => 'orange', 'fruit3' =>
3          'grape', 'fruit4' => 'apple', 'fruit5' => 'apple');
4      while ($name = current($arr)) {
5          if ($name == 'apple') {
6              echo key($arr). '<br>';
7          }
8      }
9      next($arr);
10 }
11 ?>
```

화면 출력

```
fruit1  
fruit4  
fruit5
```

02.4 배열 정렬

배열을 이용하면 배열의 값을 알파벳 또는 숫자를 기준으로 오름차순, 내림차순으로 정렬할 수 있습니다.

02.4.1 오름차순

| 내장 함수 | 오름차순 정렬

```
bool sort ( array &$array [, int $sort_flags = SORT_REGULAR ] )
```

내장 함수 `sort()`를 이용하면 배열의 값을 오름차순으로 정렬합니다.

| 내장 함수 | 이중 오름차순 정렬

```
bool asort ( array &$array [, int $sort_flags = SORT_REGULAR ] )
```

내장 함수 `asort()`는 연상 배열의 값일 경우 네임 키를 우선적으로 오름차순으로 정렬된 상태에서 서버 데이터 값을 오름차순으로 정렬합니다.

| 내장 함수 | 이름 키 오름차순 정렬

```
bool ksort ( array &$array [, int $sort_flags = SORT_REGULAR ] )
```

내장 함수 `ksort()`는 연상 배열일 경우 이름 키에 대해서 오름차순으로 정렬합니다.

02.4.2 내림차순

| 내장 함수 | 이름 키 오름차순 정렬

```
bool rsort ( array &$array [, int $sort_flags = SORT_REGULAR ] )
```

내장 함수 `rsort()`는 오름차순의 반대인 내림차순으로 배열의 값을 정렬합니다.

| 내장 함수 | 이름 키 내림차순 정렬

```
bool krsort ( array &$array [, int $sort_flags = SORT_REGULAR ] )
```

내장 함수 `krsort()`는 연상 배열일 경우 이름 키에 대해서 내림차순으로 정렬합니다.

| 내장 함수 | 내림차순 배열

```
bool arsort ( array &$array [, int $sort_flags = SORT_REGULAR ] )
```

내장 함수 `arsort()`는 인덱스 값을 유지한 상태에서 배열 요소의 데이터 값만 내림차순으로 재정렬합니다.

02.4.3 정렬 예제

내림차순 및 오름차순 정렬을 예제를 통하여 확인하도록 하겠습니다.

예제 파일 | [array-30.php](#)

```
1  <?php
2
3      $arr = array('one'=>'하나', 'two'=>'둘', 'three'=>'셋', 'four'=>'넷',
4                  'five'=>'다섯', 'six'=>'여섯', 'seven'=>'일곱');
5
6      echo "오름차순 정렬<br>";
```

```

6      sort($arr);
7      while (list($key,$val) = each($arr)) {
8          echo $key. "==" . $val. ", ";
9      }
10
11     echo "<br>";
12     echo "내림차순 정렬<br>";
13     rsort($arr);
14     while (list($key,$val) = each($arr)) {
15         echo $key. "==" . $val. ", ";
16     }
17
18     echo "<br>";
19     echo "이중 오름차순 정렬<br>";
20     asort($arr);
21     while (list($key,$val) = each($arr)) {
22         echo $key. "==" . $val. ", ";
23     }
24
25     echo "<br>";
26     echo "이름 키 오름차순 정렬<br>";
27     ksort($arr);
28     while (list($key,$val) = each($arr)) {
29         echo $key. "==" . $val. ", ";
30     }
31
32     echo "<br>";
33     echo "이름 키 내림차순 정렬<br>";
34     krsort($arr);
35     while (list($key,$val) = each($arr)) {
36         echo $key. "==" . $val. ", ";
37     }
38
39     ?>

```

화면 출력

오름차순 정렬

0==>넷, 1==>다섯, 2==>둘, 3==>셋, 4==>여섯, 5==>일곱, 6==>하나,

내림차순 정렬

0==>하나, 1==>일곱, 2==>여섯, 3==>셋, 4==>둘, 5==>다섯, 6==>넷,

이중 오름차순 정렬

6==>넷, 5==>다섯, 4==>둘, 3==>셋, 2==>여섯, 1==>일곱, 0==>하나,

이름 키 오름차순 정렬

0==>하나, 1==>일곱, 2==>여섯, 3==>셋, 4==>둘, 5==>다섯, 6==>넷,

이름 키 내림차순 정렬

6==>넷, 5==>다섯, 4==>둘, 3==>셋, 2==>여섯, 1==>일곱, 0==>하나,

02.4.4 자연 순서 정렬

내장 함수 `natsort()`는 '자연 순서' 알고리즘을 사용해 배열을 정렬합니다

| 내장 함수 |

```
bool natsort ( array &$array )
```

키와 값을 유지하면서 영숫자 문자열을 사람이 직접 분류하는 것과 같은 정렬 알고리즘을 구현합니다.

예제 파일 | [array-31.php](#)

```
1  <?php
2      $array1 = $array2 = array("img12.png", "img10.png", "img2.png",
3                                "img1.png");
4
5      echo "기본 정렬<br>";
6      asort($array1);
7      print_r($array1);
8
9      echo "<br>";
10
11     echo "자연 순서<br>";
12     natsort($array2);
13     print_r($array2);
14
15  ?>
```

화면 출력

기본 정렬

```
Array ( [3] => img1.png [1] => img10.png [0] => img12.png [2] => img2.png )
```

자연 순서

```
Array ( [3] => img1.png [2] => img2.png [1] => img10.png [0] => img12.png )
```

| 내장 함수 |

```
bool natcasesort ( array &$array )
```

내장 함수 natcasesort()는 natsort()의 대소문자를 구별하지 않는 버전입니다.

netcasesort()는 대문자와 소문자를 구별하지 않는 '자연 순서' 알고리즘을 사용해 배열을 정렬합니다.

예제 파일 | array-32.php

```
1  <?php
2      $array1 = $array2 = array('IMG0.png', 'img12.png', 'img10.png',
3      'img2.png', 'img1.png', 'IMG3.png');
4
5      echo "기본 정렬<br>";
6      sort($array1);
7      print_r($array1);
8
9      echo "<br>";
10
11     echo "자연 순서<br>";
12     natcasesort($array2);
13     print_r($array2);
14 ?>
```

화면 출력

기본 정렬

```
Array ( [0] => IMG0.png [1] => IMG3.png [2] => img1.png [3] => img10.png [4]
=> img12.png [5] => img2.png )
```

자연 순서

```
Array ( [0] => IMG0.png [4] => img1.png [3] => img2.png [5] => IMG3.png [2] =>
img10.png [1] => img12.png )
```

02.4.5 다차원 정렬

내장 함수 `array_multisort()`는 다중 또는 다차원 배열을 정렬합니다. `array_multisort()`는 한 번에 여러 배열을 정렬하거나, 하나 이상의 차원(다차원) 배열을 정렬하는 데 사용할 수 있습니다.

| 내장 함수 |

```
bool array_multisort ( array &$array1 [, mixed $array1_sort_order = SORT_ASC [,
mixed $array1_sort_flags = SORT_REGULAR [, mixed $... ]]] )
```

예제 파일 | [array-33.php](#)

```
1  <?php
2      $ar1 = array(10, 100, 100, 0);
3      $ar2 = array(1, 3, 2, 4);
4      array_multisort($ar1, $ar2);
5
6      var_dump($ar1);
7      var_dump($ar2);
8
9  ?>
```

화면 출력

```
array(4) { [0]=> int(0) [1]=> int(10) [2]=> int(100) [3]=> int(100) }
array(4) { [0]=> int(4) [1]=> int(1) [2]=> int(2) [3]=> int(3) }
```

위 예제에서는 다중 정렬합니다. 첫 번째 배열은 0, 10, 100, 100 형태로 정렬됩니다. 첫 번째 정렬 순서를 따라서 두 번째 배열은 4, 1, 2, 3 형태로 정렬됩니다.

02.5 배열 외부 함수

배열의 데이터를 처리할 때 처리하고자 하는 방식으로 별도의 외부 함수를 생성하여 작업을 위임할 수 있습니다. 생성한 외부 함수에 배열의 데이터를 하나씩 전달하여 함수를 처리하게 됩니다.

02.5.1 외부 함수 호출

내장 함수 `array_walk()`는 배열의 인자값을 응용한 사용자 정의 함수를 호출할 수 있습니다.

| 내장 함수 | 배열 사용자 함수 호출

```
bool array_walk ( array &$amp;array , callable $callback [, mixed $userdata = NULL ] )
```

예제 파일 | `array-34.php`

```
1  <?php
2      $arr = array('하나'=>'orange', '둘'=>'apple', '셋'=>'grape');
3
4      function arr_print($arr1, $key) {
5          // 호출된 배열의 값을 출력합니다.
6          echo $key . " = ". $arr1."<br>";
7      }
8
9      // 배열값과 사용자 정의 함수 arr_print를 호출합니다.
10     array_walk($arr,'arr_print');
11
12     // 배열의 포인터를 처음 위치로 초기화합니다.
13     reset($arr);
14
15     echo "<br>";
16
17     function arrKey_print(&$arr1, $key, $value) {
18         echo $value . $key . " = ". $arr1 . "<br>";
19     }
```

```

20     }
21     // 배열값과 사용자 정의 함수 arrKey_print를 호출합니다.
22     array_walk($arr, 'arrKey_print', "테스트2: ");
23
24     // 배열의 포인터를 처음 위치로 초기화합니다.
25     reset($arr);
26
27     ?>

```

화면 출력

```

하나 = orage
둘 = apple
셋 = grape

```

```

테스트2: 하나 = orage
테스트2: 둘 = apple
테스트2: 셋 = grape

```

| 내장 함수 |

```

bool array_walk_recursive ( array &$array , callable $callback [, mixed $userdata =
NULL ] )

```

내장 함수 `array_walk_recursive()`는 지정한 콜백 함수를 배열 각각의 모든 요소에게 반복적으로 함수를 적용합니다.

예제 파일 | [array-35.php](#)

```

1  <?php
2      $sweet = array('a' => 'apple', 'b' => 'banana');
3      $fruits = array('sweet' => $sweet, 'sour' => 'lemon');
4
5      print_r($fruits);
6      echo "<br>";
7
8      function test_print($item, $key)
9      {
10         echo "$key ==> $item <br>";

```

```

11     }
12
13     array_walk_recursive($fruits, 'test_print');
14
15     ?>

```

화면 출력

```

Array ( [sweet] => Array ( [a] => apple [b] => banana ) [sour] => lemon )
a ==> apple
b ==> banana
sour ==> lemon

```

위의 예제는 다중 배열로 설정되어 있습니다. 다중 배열의 각각의 요소에 사용자 지정 test_print() 함수를 적용합니다.

02.5.2 외부 함수 정렬

내부 정렬 알고리즘 이외에 별도의 함수를 생성하여 정렬 처리를 위임할 수 있습니다.

| 내장 함수 | 외부 함수 데이터 정렬

```
bool uasort ( array &$amp;array , callable $value_compare_func )
```

내장 함수 uasort()는 배열 정보와 외부 함수를 통하여 정렬합니다. 이때 인덱스 키 값은 유지합니다.

예제 파일 | array-36.php

```

1  <?php
2      // 비교 함수
3      function cmp($a, $b) {
4          if ($a == $b) return 0;
5          else return ($a < $b) ? -1 : 1;
6      }
7
8      // 배열

```



```

9   $array = array('a' => 5, 'b' => 7, 'c' => -1, 'd' => -6, 'e' => 3, 'f'
    => 2, 'g' => 8, 'h' => -3);
10  print_r($array);
11
12  echo "<br>";
13
14  // 외부 비교 함수를 통하여 정렬
15  uasort($array, 'cmp');
16  print_r($array);
17
18  ?>

```

화면 출력

```

Array ( [a] => 5 [b] => 7 [c] => -1 [d] => -6 [e] => 3 [f] => 2 [g] => 8 [h] =>
-3 )
Array ( [d] => -6 [h] => -3 [c] => -1 [f] => 2 [e] => 3 [a] => 5 [b] => 7 [g]
=> 8 )

```

| 내장 함수 | 외부 함수 키 정렬

```
bool uksort ( array &$array , callable $key_compare_func )
```

내장 함수 uksort()는 배열 정보와 외부 함수를 통하여 정렬합니다.

예제 파일 | array-37.php

```

1  <?php
2      // 비교 함수
3      function cmp($a, $b) {
4          if ($a == $b) return 0;
5          else return ($a < $b) ? -1 : 1;
6      }
7
8      // 배열
9      $array = array('a' => 5, 'b' => 7, 'c' => -1, 'd' => -6, 'e' => 3, 'f'
    => 2, 'g' => 8, 'h' => -3);
10  print_r($array);
11
12  echo "<br>";

```

```

13
14     // 외부 비교 함수를 통하여 정렬
15     uksort($array, 'cmp');
16     print_r($array);
17
18     ?>

```

화면 출력

```

Array ( [a] => 5 [b] => 7 [c] => -1 [d] => -6 [e] => 3 [f] => 2 [g] => 8 [h] =>
-3 )
Array ( [a] => 5 [b] => 7 [c] => -1 [d] => -6 [e] => 3 [f] => 2 [g] => 8 [h] =>
-3 )

```

| 내장 함수 | 외부 함수 값 정렬

```
bool usort ( array &$amp;array , callable $value_compare_func )
```

내장 함수 `usort()`는 배열 정보와 외부 함수를 통하여 정렬합니다.

예제 파일 | [array-38.php](#)

```

1  <?php
2      // 비교 함수
3      function cmp($a, $b) {
4          if ($a == $b) return 0;
5          else return ($a < $b) ? -1 : 1;
6      }
7
8      // 배열
9      $array = array('a' => 5, 'b' => 7, 'c' => -1, 'd' => -6, 'e' => 3, 'f'
=> 2, 'g' => 8, 'h' => -3);
10     print_r($array);
11
12     echo "<br>";
13
14     // 외부 비교 함수를 통하여 정렬
15     usort($array, 'cmp');
16     print_r($array);

```

```
17
18 ?>
```

화면 출력

```
Array ( [a] => 5 [b] => 7 [c] => -1 [d] => -6 [e] => 3 [f] => 2 [g] => 8 [h] =>
-3 )
Array ( [0] => -6 [1] => -3 [2] => -1 [3] => 2 [4] => 3 [5] => 5 [6] => 7 [7]
=> 8 )
```

02.5.3 콜백

내장 함수 `array_filter()`는 콜백 함수를 이용하여 배열을 필터링합니다.

| 내장 함수 |

```
array array_filter ( array $array [, callable $callback [, int $flag = 0 ]] )
```

배열의 각 값을 반복하여 콜백 함수에 전달합니다. 콜백 함수가 true를 반환하면 array의 현재 값이 결과 배열로 반환됩니다. 배열 키는 보존됩니다.

예제 파일 | [array-39.php](#)

```
1 <?php
2 function odd($var)
3 {
4     // returns whether the input integer is odd
5     return($var & 1);
6 }
7
8 function even($var)
9 {
10    // returns whether the input integer is even
11    return(!($var & 1));
12 }
13
14 $array1 = array("a"=>1, "b"=>2, "c"=>3, "d"=>4, "e"=>5);
15 $array2 = array(6, 7, 8, 9, 10, 11, 12);
```

```

16
17     echo "Odd :\n";
18     print_r(array_filter($array1, "odd"));
19
20     echo "Even:\n";
21     print_r(array_filter($array2, "even"));
22
23  ?>

```

화면 출력

```

Odd : Array ( [a] => 1 [c] => 3 [e] => 5 )
Even: Array ( [0] => 6 [2] => 8 [4] => 10 [6] => 12 )

```

| 내장 함수 |

array **array_map** (callable \$callback , array \$array1 [, array \$...])

내장 함수 array_map()은 지정된 배열 요소 각각에 콜백 함수를 적용합니다. array_map()은 배열 각각에 콜백 함수를 적용한 후 array1의 모든 요소를 포함하는 배열을 반환합니다. 콜백 함수로 전달되는 매개변수의 개수는 array_map()에 전달된 배열의 개수와 일치해야 합니다.

예제 파일 | array-40.php

```

1  <?php
2      function cube($n)
3      {
4          // 3제곱 값을 반환
5          return($n * $n * $n);
6      }
7
8      $a = array(1, 2, 3, 4, 5);
9      $b = array_map("cube", $a);
10     print_r($b);
11
12  ?>

```

화면 출력

Array ([0] => 1 [1] => 8 [2] => 27 [3] => 64 [4] => 125)

| 내장 함수 |

mixed **array_reduce** (array \$array , callable \$callback [, mixed \$initial = NULL])

내장 함수 `array_reduce()`는 콜백 함수를 반복적으로 배열 요소에 적용하여 배열을 단일 값으로 줄입니다.

예제 파일 | **array-41.php**

```
1  <?php
2      function sum($carry, $item)
3      {
4          $carry += $item;
5          return $carry;
6      }
7
8      function product($carry, $item)
9      {
10         $carry *= $item;
11         return $carry;
12     }
13
14     $a = array(1, 2, 3, 4, 5);
15     $x = array();
16
17     var_dump(array_reduce($a, "sum"));
18     // int(15)
19
20     var_dump(array_reduce($a, "product", 10));
21     // int(1200), because: 10*1*2*3*4*5
22     // 입력한 10도 같이 포함하여 동작함
23
24     var_dump(array_reduce($x, "sum", "No data to reduce"));
25     // string(17) "No data to reduce"
26     // 문자열의 개수만큼 반복된 17번의 $carry 값이 출력됨
27  ?>
```

화면 출력

```
int(15)
int(1200)
string(17)
"No data to reduce"
```

02.6 배열 연산

배열은 여러 개의 유사한 값들을 가지고 있습니다. 이러한 특성을 위해서 배열의 상태나 연산 등의 특수 처리를 할 수도 있습니다.

02.6.1 개수

배열을 처리할 때는 반복문을 사용하여 처리하는 경우가 대부분입니다. 이런 경우 배열을 처리하기 위해서 반복 횟수를 알아야 합니다.

PHP는 배열의 크기를 통하여 요소의 개수를 확인할 수 있는 특별한 함수를 제공합니다.

| 내장 함수 |

```
int count ( mixed $array_or_countable [, int $mode = COUNT_NORMAL ] )
```

내장 함수 `count()`는 배열의 데이터의 개수가 몇 개가 있는지 확인할 수 있는 간단한 함수입니다.

`count()` 함수를 이용하면 배열에 들어 있는 데이터 요소의 개수를 확인할 수 있습니다. `sizeof()`는 `count()`의 또 다른 별칭입니다.

예제 파일 | [array-42.php](#)

```
1 <?php
2     $arr = array("hojin","jiny","james","eric");
3     echo "배열의 개수는 = ". count($arr) . " 입니다.<br>";
```

```

4
5     echo "배열의 개수는 = ". sizeof($arr) . " 입니다.<br>";
6
7     ?>

```

화면 출력

배열의 개수는 = 4 입니다.
배열의 개수는 = 4 입니다.

| 내장 함수 |

```
array array_pad ( array $array , int $size , mixed $value )
```

내장 함수 `array_pad()`는 입력된 배열과 함께 지정된 개수(`size`)의 배열을 반환합니다.

지정한 배열의 개수의 값이 양수면, 입력 배열 뒤쪽에 `$value` 값으로 채워진 배열을 생성하여 반환합니다. 또는 지정한 배열의 개수의 값이 음수면, 입력 배열 앞쪽에 `$value` 값으로 채워진 배열을 생성합니다.

만일 지정한 배열의 개수가 입력된 배열의 개수보다 작은 경우에는 패딩 처리를 하지 않습니다. 한 번에 최대 생성할 수 있는 배열의 개수는 1048576개입니다.

예제 파일 | [array-43.php](#)

```

1  <?php
2      $input = array(12, 10, 9);
3
4      $result = array_pad($input, 5, 0);
5      // result is array(12, 10, 9, 0, 0)
6      print_r($result);
7      echo "<br>";
8
9      $result = array_pad($input, -7, -1);
10     // result is array(-1, -1, -1, -1, 12, 10, 9)
11     print_r($result);
12     echo "<br>";
13

```

```

14     $result = array_pad($input, 2, "noop");
15     // not padded
16     print_r($result); echo "<br>";
17
18     ?>

```

화면 출력

```

Array ( [0] => 12 [1] => 10 [2] => 9 [3] => 0 [4] => 0 )
Array ( [0] => -1 [1] => -1 [2] => -1 [3] => -1 [4] => 12 [5] => 10 [6] => 9 )
Array ( [0] => 12 [1] => 10 [2] => 9 )

```

02.6.2 범위

내장 함수 `range()`는 정수값을 갖는 배열이 있을 때 값의 범위를 구하여 데이터를 추출할 수 있습니다.

| 내장 함수 | 정수값의 배열

```
array range ( mixed $start , mixed $end [, number $step = 1 ] )
```

예제 파일 | `array-44.php`

```

1  <?php
2      // 배열
3      $array = array('a' => 5, 'b' => 7, 'c' => -1, 'd' => -6, 'e' => 3, 'f'
        => 2, 'g' => 8, 'h' => -3);
4
5      $a = range(3,10);
6      print_r($a);
7
8      ?>

```

화면 출력

```

Array ( [0] => 3 [1] => 4 [2] => 5 [3] => 6 [4] => 7 [5] => 8 [6] => 9 [7] =>
10 )

```


02.6.3 연산 처리

내장 함수 `array_sum()`은 숫자값의 배열이 있을 때 배열값의 총합을 출력합니다.

| 내장 함수 | 배열의 합

number **array_sum** (array \$array)

예제 파일 | **array-45.php**

```
1  <?php
2      // 배열
3      $array = array('a' => 5, 'b' => 7, 'c' => -1, 'd' => -6, 'e' => 3, 'f'
=> 2, 'g' => 8, 'h' => -3);
4
5      $a = array_sum($array);
6      echo "배열의 합 = ".$a;
7
8  ?>
```

화면 출력

배열의 합 = 15

| 내장 함수 | 배열의 곱

number **array_product** (array \$array)

내장 함수 `array_product()`은 입력된 배열의 값을 모두 곱하여 출력합니다.

예제 파일 | **array-46.php**

```
1  <?php
2
3      $a = array(1, 2, 3, 4, 5);
4      echo "배열의 곱 = " . array_product($a);
5
6
7  ?>
```

화면 출력

배열의 곱 = 120

02.6.4 비정렬

배열을 처리하는 데 있어서 정렬 처리하여 사용하는 경우가 많이 있습니다. 이와 반대로 배열의 데이터를 비정렬화하여 처리할 수 있습니다.

PHP는 데이터의 비정렬화를 할 수 있는 몇 가지 함수를 제공합니다.

| 내장 함수 | 배열을 뒤섞음

```
bool shuffle ( array &$array )
```

내장 함수 `shuffle()`은 입력된 배열의 순서를 뒤섞어서 출력합니다.

예제 파일 | array-47.php

```
1  <?php
2      // 배열
3      $array = array('a' => 5, 'b' => 7, 'c' => -1, 'd' => -6, 'e' => 3, 'f'
4          => 2, 'g' => 8, 'h' => -3);
5      print_r($array);
6
7      echo "<br> === 배열 뒤섞음 === <br>";
8
9      shuffle($array);
10     print_r($array);
11 ?>
```

화면 출력

```
Array ( [a] => 5 [b] => 7 [c] => -1 [d] => -6 [e] => 3 [f] => 2 [g] => 8 [h] =>
-3 )
=== 배열 뒤섞음 ===
Array ( [0] => 7 [1] => -6 [2] => 8 [3] => 5 [4] => 3 [5] => 2 [6] => -3 [7] =>
-1 )
```

| 내장 함수 | 배열 난수

```
mixed array_rand ( array $array [, int $num = 1 ] )
```

내장 함수 `array_rand()`는 입력된 배열에서 지정한 수만큼의 임의의 키를 반환합니다.

예제 파일 | **array-48.php**

```
1 <?php
2     $country = array("korea", "usa", "japan", "china", "frach", "canada");
3
4     // 배열에서 임의 값 2개를 추출합니다.
5     $keys = array_rand($country, 2);
6
7     echo $country[$keys[0]] . "\n";
8     echo $country[$keys[1]] . "\n";
9
10 ?>
```

화면 출력

japan china

| 내장 함수 |

```
array array_replace ( array $array1 , array $array2 [, array $... ] )
```

내장 함수 `array_replace()`는 배열 `array1`의 값을 다음 입력되는 배열의 동일한 키를 기준으로 값을 바꿉니다. 또한 세 번째 배열에서 동일한 키 값이 있을 경우 계속 대체됩니다. 배열은 순서대로 이전 값을 덮어씁니다.

예제 파일 | **array-49.php**

```
1 <?php
2     $base = array("orange", "banana", "apple", "raspberry");
3     $replacements = array(0 => "pineapple", 4 => "cherry");
4
```

```

5     $basket = array_replace($base, $replacements);
6     print_r($basket);
7
8     ?>

```

화면 출력

```

Array ( [0] => pineapple [1] => banana [2] => apple [3] => raspberry [4] =>
cherry )

```

| 내장 함수 |

```

array array_replace_recursive ( array $array1 , array $array2 [, array $... ] )

```

내장 함수 `array_replace_recursive()`는 배열의 요소를 재귀적으로 대체합니다. `array_replace_recursive()`는 `array1`의 값을 다음 배열의 값과 동일하게 대체합니다.

첫 번째 배열의 키가 두 번째 배열에 존재하면 기존 값은 두 번째 배열의 값으로 대체됩니다. 키가 첫 번째 배열이 아닌 두 번째 배열에 존재하면 첫 번째 배열에 키가 만들어집니다. 키 값이 첫 번째 배열에만 있는 경우 그대로 유지합니다.

예제 파일 | array-50.php

```

1  <?php
2      $base = array('citrus' => array( "orange" ) ,
3                  'berries' =>array("blackberry", "raspberry"), );
4      $replacements = array('citrus' => array('pineapple'),
5                          'berries' => array('blueberry'));
6
7      $basket = array_replace_recursive($base, $replacements);
8      print_r($basket);
9      echo "<br>";
10
11     $basket = array_replace($base, $replacements);
12     print_r($basket);
13
14     ?>

```

화면 출력

```
Array ( [citrus] => Array ( [0] => pineapple ) [berries] => Array ( [0] =>
blueberry [1] => raspberry ) )
Array ( [citrus] => Array ( [0] => pineapple ) [berries] => Array ( [0] =>
blueberry ) )
```

02.6.5 변환

내장 함수 `array_change_key_case()`는 인덱스 키의 대문자 또는 소문자로 변환합니다.

| 내장 함수 | 인덱스 키 대소문자 변환

```
array array_change_key_case ( array $array [, int $case = CASE_LOWER ] )
```

예제 파일 | array-51.php

```
1  <?php
2      $arr = array("FirSt" => 1, "Sec0nd" => 4);
3
4      // 대문자
5      $UPPER = array_change_key_case($arr, CASE_UPPER);
6      print_r($UPPER);
7
8      echo "<br> ===== <br>";
9      // 소문자
10     $LOWER = array_change_key_case($arr, CASE_LOWER);
11     print_r($LOWER);
12
13  ?>
```

화면 출력

```
Array ( [FIRST] => 1 [SECOND] => 4 )
=====
Array ( [first] => 1 [second] => 4 )
```

02.6.6 집합

내장 함수 `array_intersect()`는 키 값은 유지하면서 2개의 배열의 교차 부분(교집합)을 계산하여 배열로 반환합니다. 값을 기준으로 비교합니다.

| 내장 함수 |

```
array array_intersect ( array $array1 , array $array2 [, array $... ] )
```

예제 파일 | [array-52.php](#)

```
1 <?php
2     $array1 = array("a" => "green", "red", "blue");
3     $array2 = array("b" => "green", "yellow", "red");
4     $result = array_intersect($array1, $array2);
5     print_r($result);
6
7 ?>
```

화면 출력

```
Array ( [a] => green [0] => red )
```

| 내장 함수 |

```
array array_intersect_assoc ( array $array1 , array $array2 [, array $... ] )
```

내장 함수 `array_intersect_assoc()`는 `array_intersect()`와 달리 키 값도 비교하는 데 사용됩니다. 2개의 배열을 비교하여 교차 부분(교집합)만 배열로 반환합니다.

예제 파일 | [array-53.php](#)

```
1 <?php
2     $array1 = array("a" => "green", "b" => "brown", "c" => "blue", "red");
3     $array2 = array("a" => "green", "b" => "yellow", "blue", "red");
4     $result_array = array_intersect_assoc($array1, $array2);
```

```

5     print_r($result_array);
6
7     ?>

```

화면 출력

Array ([a] => green)

| 내장 함수 |

array **array_intersect_key** (array \$array1 , array \$array2 [, array \$...])

내장 함수 array_intersect_key()는 키 값을 사용하여 배열의 교차 부분을 계산합니다. array_intersect_key()는 array1의 키를 기준으로 교차되는 모든 항목의 배열을 반환합니다.

예제 파일 | array-54.php

```

1  <?php
2      $array1 = array('blue' => 1, 'red' => 2, 'green' => 3,
3                      'purple' => 4);
4      $array2 = array('green' => 5, 'blue' => 6, 'yellow' => 7,
5                      'cyan' => 8);
6
7      var_dump(array_intersect_key($array1, $array2));
8
9      ?>

```

화면 출력

array(2) { ["blue"]=> int(1) ["green"]=> int(3) }

| 내장 함수 |

array **array_intersect_uassoc** (array \$array1 , array \$array2 [, array \$...], callable \$key_compare_func)

내장 함수 `array_intersect_uassoc()`는 콜백 함수를 사용하여 인덱스를 비교합니다. `array_intersect_uassoc()`는 `array1`을 기준으로 배열의 교차 부분을 비교합니다.

예제 파일 | `array-55.php`

```
1 <?php
2     $array1 = array("a" => "green", "b" => "brown", "c" => "blue", "red");
3     $array2 = array("a" => "GREEN", "B" => "brown", "yellow", "red");
4
5     print_r(array_intersect_uassoc($array1, $array2, "strcasecmp"));
6
7 ?>
```

화면 출력

Array ([b] => brown)

| 내장 함수 |

`array array_intersect_ukey` (array \$array1 , array \$array2 [, array \$...], callable \$key_compare_func)

내장 함수 `array_intersect_ukey()`는 외부 콜백 함수를 통하여 두 배열의 교차 부분(교집합)을 계산합니다. `array_intersect_ukey()`는 일치하는 키를 가진 `array1`을 기준으로 배열을 반환합니다.

예제 파일 | `array-56.php`

```
1 <?php
2     function key_compare_func($key1, $key2)
3     {
4         if ($key1 == $key2)
5             return 0;
6         else if ($key1 > $key2)
7             return 1;
8         else
9             return -1;
10    }
```



```

11
12     $array1 = array('blue' => 1, 'red' => 2, 'green' => 3,
13                     'purple' => 4);
14     $array2 = array('green' => 5, 'blue' => 6, 'yellow' => 7,
15                     'cyan' => 8);
16
17     var_dump(array_intersect_ukey($array1, $array2, 'key_compare_func'));
18
19     ?>

```

화면 출력

```
array(2) { ["blue"]=> int(1) ["green"]=> int(3) }
```

| 내장 함수 |

```
array array_uintersect ( array $array1 , array $array2 [, array $... ], callable $value_
compare_func )
```

내장 함수 `array_uintersect()`는 콜백 함수를 통하여 배열의 교차점을 비교합니다.

예제 파일 | [array-57.php](#)

```

1  <?php
2      $array1 = array("a" => "green", "b" => "brown", "c" => "blue", "red");
3      $array2 = array("a" => "GREEN", "B" => "brown", "yellow", "red");
4
5      print_r(array_uintersect($array1, $array2, "strcasecmp"));
6
7      ?>

```

화면 출력

```
Array ( [a] => green [b] => brown [0] => red )
```

| 내장 함수 |

```
array array_uintersect_assoc ( array $array1 , array $array2 [, array $... ], callable  
$value_compare_func )
```

내장 함수 `array_uintersect_assoc()`는 인덱스 검사를 통해 배열의 교차점을 계산 후에 콜백 함수를 통하여 데이터를 비교합니다.

예제 파일 | [array-58.php](#)

```
1 <?php
2     $array1 = array("a" => "green", "b" => "brown", "c" => "blue", "red");
3     $array2 = array("a" => "GREEN", "B" => "brown", "yellow", "red");
4
5     print_r(array_uintersect_assoc($array1, $array2, "strcasecmp"));
6
7 ?>
```

화면 출력

```
Array ( [a] => green )
```

| 내장 함수 |

```
array array_uintersect_uassoc ( array $array1 , array $array2 [, array $... ], callable  
$value_compare_func , callable $key_compare_func )
```

내장 함수 `array_uintersect_uassoc()`는 인덱스 검사를 통해 배열의 교차점을 찾은 후에 별도의 콜백 함수를 통하여 데이터와 인덱스를 비교합니다.

예제 파일 | [array-59.php](#)

```
1 <?php
2     $array1 = array("a" => "green", "b" => "brown", "c" => "blue", "red");
3     $array2 = array("a" => "GREEN", "B" => "brown", "yellow", "red");
4
5     print_r(array_uintersect_uassoc($array1, $array2, "strcasecmp",
```

```
"strcasecmp"));
```

```
6    ?>
```

화면 출력

```
Array ( [a] => green [b] => brown )
```

02.7 비교

배열의 데이터 값, 키를 비교하여 처리할 수 있습니다.

02.7.1 값의 비교

| 내장 함수 |

```
array array_diff ( array $array1 , array $array2 [, array $... ] )
```

내장 함수 `array_diff()`는 배열의 차이를 계산합니다. `array1`을 다른 배열과 비교합니다. 비교 후에 `array1`에는 있지만 다른 배열에는 없는 값들을 반환합니다.

예제 파일 | **array-60.php**

```
1  <?php
2      $array1 = array("a" => "green", "red", "blue", "red");
3      $array2 = array("b" => "green", "yellow", "red");
4      $result = array_diff($array1, $array2);
5
6      print_r($result);
7
8  ?>
```

화면 출력

```
Array ( [1] => blue )
```

| 내장 함수 |

```
array array_udiff ( array $array1 , array $array2 [, array $... ], callable $value_
compare_func )
```

내장 함수 `array_udiff()`는 콜백 함수를 사용하여 배열의 차이를 계산합니다. 데이터 비교를 위한 내부 함수를 사용하는 `array_diff()`와는 차이가 있습니다.

예제 파일 | `array-61.php`

```
1  <?php
2      // Arrays to compare
3      $array1 = array(new stdClass, new stdClass,
4                      new stdClass, new stdClass,
5                      );
6
7      $array2 = array(
8          new stdClass, new stdClass,
9          );
10
11     // Set some properties for each object
12     $array1[0]->width = 11; $array1[0]->height = 3;
13     $array1[1]->width = 7;  $array1[1]->height = 1;
14     $array1[2]->width = 2;  $array1[2]->height = 9;
15     $array1[3]->width = 5;  $array1[3]->height = 7;
16
17     $array2[0]->width = 7;  $array2[0]->height = 5;
18     $array2[1]->width = 9;  $array2[1]->height = 2;
19
20     function compare_by_area($a, $b) {
21         $areaA = $a->width * $a->height;
22         $areaB = $b->width * $b->height;
23
24         if ($areaA < $areaB) {
25             return -1;
26         } elseif ($areaA > $areaB) {
27             return 1;
28         } else {
```

```

29         return 0;
30     }
31 }
32
33 print_r(array_udiff($array1, $array2, 'compare_by_area'));
34
35 ?>

```

화면 출력

```

Array ( [0] => stdClass Object ( [width] => 11 [height] => 3 ) [1] => stdClass
Object ( [width] => 7 [height] => 1 ) )

```

| 내장 함수 |

```
array array_diff_assoc ( array $array1 , array $array2 [, array $... ] )
```

내장 함수 array_diff_assoc()는 인덱스를 기준으로 배열의 차이를 계산합니다.

array1과 array2를 비교하여 차이를 반환합니다. 비교 후에 array1에 있고 다른 배열에는 없는 값들을 반환합니다. array_diff()와 달리 배열 키도 비교에 사용합니다.

예제 파일 | array-62.php

```

1  <?php
2      $array1 = array("a" => "green", "b" => "brown", "c" => "blue", "red");
3      $array2 = array("a" => "green", "yellow", "red");
4      $result = array_diff_assoc($array1, $array2);
5      print_r($result);
6
7  ?>

```

화면 출력

```

Array ( [b] => brown [c] => blue [0] => red )

```

| 내장 함수 |

```
array array_udiff_assoc ( array $array1 , array $array2 [, array $... ], callable $value_
compare_func )
```

내장 함수 `array_udiff_assoc()`는 콜백 함수를 통하여 인덱스 검사하고 배열의 데이터를 비교합니다.

예제 파일 | array-63.php

```
1  <?php
2      class cr {
3          private $priv_member;
4          function cr($val)
5          {
6              $this->priv_member = $val;
7          }
8
9          static function comp_func_cr($a, $b)
10         {
11             if ($a->priv_member === $b->priv_member) return 0;
12             return ($a->priv_member > $b->priv_member)? 1:-1;
13         }
14     }
15
16     $a = array("0.1" => new cr(9), "0.5" => new cr(12),
17               0 => new cr(23), 1=> new cr(4), 2 => new cr(-15),);
18     $b = array("0.2" => new cr(9), "0.5" => new cr(22),
19               0 => new cr(3), 1=> new cr(4), 2 => new cr(-15),);
20
21     $result = array_udiff_assoc($a, $b, array("cr", "comp_func_cr"));
22     print_r($result);
23  ?>
```

화면 출력

```
Array ( [0.1] => cr Object ( [priv_member:cr:private] => 9 ) [0.5] => cr Object (
[priv_member:cr:private] => 12 ) [0] => cr Object ( [priv_member:cr:private]
=> 23 ) )
```

| 내장 함수 |

```
array array_udiff_uassoc ( array $array1 , array $array2 [, array $... ],  
    callable $value_compare_func , callable $key_compare_func )
```

내장 함수 `array_udiff_uassoc()`는 콜백 함수로 데이터와 인덱스를 비교합니다.

먼저 인덱스 검사를 통해 배열의 차이점을 계산합니다. 계산된 데이터 및 인덱스를 콜백 함수를 통하여 비교합니다.

예제 파일 | **array-64.php**

```
1  <?php  
2  
3  class cr {  
4      private $priv_member;  
5      function cr($val)  
6      {  
7          $this->priv_member = $val;  
8      }  
9  
10     static function comp_func_cr($a, $b)  
11     {  
12         if ($a->priv_member === $b->priv_member) return 0;  
13         return ($a->priv_member > $b->priv_member)? 1:-1;  
14     }  
15  
16     static function comp_func_key($a, $b)  
17     {  
18         if ($a === $b) return 0;  
19         return ($a > $b)? 1:-1;  
20     }  
21 }  
22  
23 $a = array("0.1" => new cr(9), "0.5" => new cr(12),  
24           0 => new cr(23), 1=> new cr(4), 2 => new cr(-15),);  
25 $b = array("0.2" => new cr(9), "0.5" => new cr(22),  
26           0 => new cr(3), 1=> new cr(4), 2 => new cr(-15),);  
27
```

```

28     $result = array_udiff_uassoc($a, $b, array("cr", "comp_func_cr"),
    array("cr", "comp_func_key"));
29     print_r($result);
30
31     ?>

```

화면 출력

```

Array ( [0.1] => cr Object ( [priv_member:cr:private] => 9 ) [0.5] => cr Object (
[priv_member:cr:private] => 12 ) [0] => cr Object ( [priv_member:cr:private]
=> 23 ) )

```

| 내장 함수 |

```

array array_diff_uassoc ( array $array1 , array $array2 [, array $... ], callable $key_
compare_func )

```

내장 함수 `array_diff_uassoc()`는 사용자 지정 콜백 함수를 통하여 배열을 검사합니다.
`array1`과 `array2`를 비교하여 차이를 반환합니다.

사용자가 제공하는 콜백 함수는 인덱스 비교에 사용됩니다.

예제 파일 | [array-65.php](#)

```

1  <?php
2      function key_compare_func($a, $b)
3      {
4          if ($a === $b) {
5              return 0;
6          }
7          return ($a > $b)? 1:-1;
8      }
9
10     $array1 = array("a" => "green", "b" => "brown", "c" => "blue", "red");
11     $array2 = array("a" => "green", "yellow", "red");
12     $result = array_diff_uassoc($array1, $array2, "key_compare_func");
13     print_r($result);
14     ?>

```


화면 출력

```
Array ( [b] => brown [c] => blue [0] => red )
```

02.7.2 키의 비교

| 내장 함수 |

```
array array_diff_key ( array $array1 , array $array2 [, array $... ] )
```

내장 함수 `array_diff_key()`는 키를 사용하여 배열의 차이점을 계산합니다.

`array1`의 키와 `array2`의 키를 비교하여 차이를 배열로 반환합니다. 이 함수는 `array_diff()`와 비슷하지만 비교 값이 키에서 수행된다는 점이 다릅니다.

예제 파일 | [array-66.php](#)

```
1 <?php
2     $array1 = array('blue' => 1, 'red' => 2, 'green' => 3, 'purple' => 4);
3     $array2 = array('green' => 5, 'yellow' => 7, 'cyan' => 8);
4
5     var_dump(array_diff_key($array1, $array2));
6
7 ?>
```

화면 출력

```
array(3) { ["blue"]=> int(1) ["red"]=> int(2) ["purple"]=> int(4) }
```

| 내장 함수 |

```
array array_diff_ukey ( array $array1 , array $array2 [, array $... ], callable $key_
compare_func )
```

내장 함수 `array_diff_ukey()`는 사용자 콜백 함수를 통하여 배열의 키를 비교합니다.

array1의 키와 array2의 키를 비교하여 차이를 반환합니다. 이 함수는 array_diff()와 비슷합니다만 비교 값 대신 키에서 수행된다는 점에서 차이가 있습니다.

예제 파일 | **array-67.php**

```
1  <?php
2      function key_compare_func($key1, $key2)
3      {
4          if ($key1 == $key2)
5              return 0;
6          else if ($key1 > $key2)
7              return 1;
8          else
9              return -1;
10     }
11
12     $array1 = array('blue' => 1, 'red' => 2,
13                    'green' => 3, 'purple' => 4);
14     $array2 = array('green' => 5, 'blue' => 6,
15                    'yellow' => 7, 'cyan' => 8);
16
17     var_dump(array_diff_ukey($array1, $array2, 'key_compare_func'));
18
19  ?>
```

화면 출력

```
array(2) { ["red"]=> int(2) ["purple"]=> int(4) }
```

03

문자열

문자열은 응용프로그램을 개발하면서 가장 많이 사용되는 기능입니다. 프로그램의 복잡한 동작과 수치적인 연산을 하지만 최종적으로 사용자에게 보여지는 모든 것은 글자화된 문자열입니다.

PHP는 문자들을 다루고 처리할 수 있도록 다양한 문자열 함수를 제공하고 있습니다. 일부 내장 함수들은 추가 모듈 설치가 필요할 수도 있습니다.

03.1 문자열 변수

PHP는 C 언어 스타일의 인터프리터 언어입니다. 따라서 기본적인 문자열의 처리와 개념 또한 C 언어의 속성과 스타일을 따르는 경우가 많습니다.

PHP에서 문자열을 조작하고 처리하는 것은 생각보다 매우 쉽습니다. C 언어처럼 변수의 타입과 엄격한 문자와 문자열을 처리할 수 있는 규칙을 적용하지는 않습니다. 또한 PHP는 변수에 문자열을 입력함으로써 자동적으로 문자열이 변수형 타입으로 설정됩니다.

03.1.1 문자열 길이

문자열 데이터를 처리할 때 가장 많이 사용하는 기능은 문자열의 길이를 측정하는 것입니다. 문자열은 여러 개의 문자들의 집합이기 때문에 총 몇 개의 글자로 구성되어 있는지 확인하는 것이 중요합니다.

문자열의 길이를 기반으로 문자열을 처리해야 하는 패턴의 처리량이 결정됩니다. C 언어와 같은 언어에서는 문자열을 처리할 때 주로 배열을 통하여 저장합니다. 실제로 PHP 내부에서도 변수에 문자열을 저장하면 배열 형태로 처리할 수 있습니다.

```
$msg = "hello world!";
```

PHP에서 위와 같이 변수의 생성과 문자열을 입력합니다. 변수 `$msg`는 문자열의 데이터를 가지고 있기 때문에 문자열 변수라고 할 수 있습니다. 그리고 `$msg` 변수는 배열의 접근 방식으로 한 글자 한 글자를 지정할 수 있습니다.

배열과 같은 특성을 가진 문자열 변수는 길이를 측정할 수 있습니다. 한 글자씩 문자를 포함하고 있는 메모리의 크기를 계산하면 되기 때문입니다. PHP에서는 문자열의 길이를 측정할 수 있는 내장 함수를 제공하고 있습니다.

| 내장 함수 |

```
int strlen ( string $string )
```

내장 함수 `strlen()`은 변수에 들어 있는 문자열의 길이를 확인할 수 있습니다. 함수의 매개변수 값으로 길이를 측정할 변수명을 입력하면 됩니다. 문자열의 길이는 정수 형태의 값으로 반환합니다. 문자열은 문자들의 연결된 배열의 크기를 측정하는 것과 같습니다.

예제 파일 | **strlen-01.php**

```
1  <?php
2      // 문자열 변수를 생성합니다.
3      $name = "Hello world!";
4
5      // 문자의 개수를 출력합니다.
```

```

6      echo "문자의 개수는:".strlen($name);
7
8      ?>

```

화면 출력

문자의 개수는: 12

문자열의 변수가 배열 형태를 띠고 있고, strlen() 함수를 통하여 문자열의 길이를 측정할 수 있었습니다. 이러한 특성을 이용하여 문자열 변수에서 한 글자씩 문자 값을 불러서 개별적으로 처리도 가능합니다.

예제 파일 | strlen-02.php

```

1  <?php
2      // 문자열 변수를 생성합니다.
3      $name = "Hello world!";
4
5      // 한 줄에 문자 한 글자씩 출력합니다.
6      for ($i=0;$i<strlen($name);$i++) {
7          echo $name[$i]."<br>";
8      }
9
10     ?>

```

화면 출력

H
e
l
l
o

w
o
r
l
d
!

위의 실험은 문자열의 변수를 배열로 접근하여 한 글자씩 출력합니다. 문자열을 배열 형

태로 접근할 때는 문자열 변수명 뒤에 대괄호를 통하여 접근합니다. 대괄호 안에 접근 값으로 0부터 시작되는 인덱스 값을 넣어주면 됩니다.

03.1.2 문자열 카운트

문자열 안에 또 다른 문자열을 포함하고 있을 경우 포함되고 있는 문자의 반복 개수를 측정할 수 있습니다.

| 내장 함수 |

```
int substr_count ( string $haystack , string $needle [, int $offset = 0 [, int $length ] ] )
```

내장 함수 `substr_count()`은 원본의 `$haystack` 변수에서 검색하고자 하는 문자열 `$needle` 값이 몇 개를 포함하고 있는지를 확인합니다. 반환값은 정수입니다. 검색할 때는 대소문자를 구분하여 처리합니다.

예제 파일 | `substr_count.php`

```
1  <?php
2      $text = 'This is a test';
3      echo $text . "<br>";
4
5      echo "원본 문자열 길이 = ". strlen($text) . "<br>";
6      // 출력: 14
7
8      // 'This is a test' 안에 is 두 번 존재
9      echo "needle substring이 발생하는 횟수 = ". substr_count($text,
10         'is') . "<br>";
11     // 출력 :2
12
13     // 오프셋 3 적용하여 문자열은 's is a test' 형태로 앞에 3개의 문자열은 건너뜁니다.
14     echo "오프셋 적용 = ".substr_count($text, 'is', 3) . "<br>";
15
16     // 오프셋 3 적용 및 문자열의 길이는 3으로 제한합니다.
17     // 따라서 문자열은 's i'로 적용됩니다.
18     echo "오프셋 적용, 길이 제한 = ".substr_count($text, 'is', 3, 3) . "<br>";
```

```

18
19 // 오류 발생
20 // 오프셋 5 + 길이 10 = 총길이 15로 원본 $text 문자열 14보다 크기 때문에 오류 발생
21 echo substr_count($text, 'is', 5, 10) . "<br>";
22
23 // prints only 1, because it doesn't count overlapped substrings
24 $text2 = 'gcdgcdgcd';
25 echo substr_count($text2, 'gcdgcd') . "<br>";
26
27 ?>

```

화면 출력

```

This is a test
원본 문자열 길이 = 14
needle substring이 발생하는 횟수 =2
오프셋 적용 = 1
오프셋 적용, 길이 제한 = 0

```

위의 실험은 \$text 문자열에서 is의 문자열을 포함하고 있는 문자열의 개수를 찾아서 처리합니다.

03.1.3 단어 개수 측정

문자열은 하나의 단어일 수도 있고 문장일 수도 있습니다. 만일 문자열이 여러 개의 단어로 구성되어 있는 문장일 경우 포함하고 있는 단어의 개수를 측정할 수 있습니다.

| 내장 함수 |

```
mixed str_word_count ( string $string [, int $format = 0 [, string $charlist ] ] )
```

내장 함수 str_word_count()은 입력된 문자열에서 단어의 개수를 찾아서 반환합니다.

예제 파일 | str_word_count.php

```

1 <?php
2 $str = "I love you";

```

```

3     echo str_word_count($str);
4
5     ?>

```

화면 출력

3

03.2 문자열 자르기

문자열을 처리하는 데 있어서 많이 사용하는 기능은 문자열을 분리하는 것입니다. 문자열은 다양한 형태로 존재하기 때문에 관련 함수들이 많이 지원됩니다.

03.2.1 문자열 제거

내장 함수 `substr()`은 입력한 문자열의 일부분을 잘라낼 수 있습니다. 문자열을 잘라낼 때는 문자열의 시작 위치와 끝 위치를 지정하여 해당 부분을 추출합니다.

| 내장 함수 |

```
string substr ( string $string , int $start [ , int $length ] )
```

예제 파일 | **substr.php**

```

1  <?php
2      $string = "abcdefghijklmnopqrstuvwxy";
3
4      // 1 위치 이후부터 표시함
5      echo substr($string,1)."<br>";
6
7      // 3 위치 이후부터 5개 표시함
8      echo substr($string,3,5)."<br>";
9
10     ?>

```


화면 출력

```
bcdefghijklmnopqrstuvwxyz  
defgh
```

위의 예제는 문자열 \$string 변수에서 특정 부분의 문자열을 추출합니다. 문자열을 추출할 때 세 번째 끝 값은 생략이 가능합니다. 끝 값을 생략할 때는 시작부터 끝까지를 모두 출력합니다.

03.2.2 공백 제거

문자열 변수는 공백도 문자로 취급을 합니다. 문자열을 가공하다가 보면 문자열 앞 또는 뒤에 쓸모가 없는 공백 문자들이 있는 경우가 있습니다. 이러한 경우 공백을 자동적으로 제거할 수 있는 내장 함수를 제공하고 있습니다.

PHP는 공백을 제거하고 처리할 수 있는 3개의 함수를 제공합니다. trim(), ltrim(), rtrim() 함수는 문자열의 공백을 제거합니다.

| 내장 함수 | 양쪽 공백 제거

```
string trim ( string $str [, string $character_mask = "\t\n\r\0\x0B" ] )
```

내장 함수 trim()은 문자열의 앞뒤에 존재하는 공백 문자를 제거합니다.

| 내장 함수 | 우측 공백 제거

```
string rtrim ( string $str [, string $character_mask ] )  
string chop (string str);
```

내장 함수 rtrim(), chop()은 문자열 오른쪽에 있는 공백 문자를 제거합니다. chop() 함수는 rtrim() 함수의 별칭입니다.

| 내장 함수 | 좌측 공백 제거

```
string ltrim ( string $str [, string $character_mask ] )
```

내장 함수 ltrim()은 문자열의 왼쪽에 있는 공백 문자를 제거합니다.

예제 파일 | trim.php

```
1  <?php
2      $string = "  jiny  ";
3      echo "안녕하세요!".$string."입니다.<br>";
4
5      // 앞뒤 공백 문자열을 삭제합니다.
6      echo "안녕하세요!".trim($string)."입니다.<br>";
7
8      // 오른쪽 공백을 제거합니다.
9      echo "안녕하세요!".chop($string)."입니다.<br>";
10     echo "안녕하세요!".rtrim($string)."입니다.<br>";
11
12     // 왼쪽 공백을 제거합니다.
13     echo "안녕하세요!".ltrim($string)."입니다.<br>";
14
15  ?>
```

화면 출력

안녕하세요! jiny 입니다.

안녕하세요!jiny입니다.

안녕하세요! jiny입니다.

안녕하세요! jiny입니다.

안녕하세요!jiny 입니다.

03.3 문자열 검색

문자열을 다양하게 처리하기 위해서는 문자열에서 특정 문자들의 위치를 찾아 처리하는 것입니다. 문자열의 내용을 검색하고 값을 반환하거나 위치 값을 구할 수 있습니다.

03.3.1 첫 번째 검색

내장 함수 `strstr()`은 문자열 안에서 찾을 문자의 첫 번째 위치를 구하고, 이후의 문자열을 데이터를 반환합니다. 문자를 검색할 때는 대소문자를 구별을 하는 점을 주의해야 합니다.

| 내장 함수 |

```
string strstr ( string $haystack , mixed $needle [, bool $before_needle = false ] )
```

`strchr()` 함수는 `strstr()` 함수의 별칭으로 동일한 작업을 수행합니다.

예제 파일 | `strstr.php`

```
1  <?php
2      $str = "abcdeghijklm-abcdeghijklm-1234567";
3      echo "원본 : " . $str . "<br>";
4
5      // 문자열에서 em-으로 시작하는 위치를 찾아
6      // 이후의 문자열을 반환합니다.
7      $a = strstr($str,"em-");
8      echo $a;
9
10  ?>
```

화면 출력

```
원본 : abcdeghijklm-abcdeghijklm-1234567
em-abcdeghijklm-1234567
```

| 내장 함수 |

```
string stristr ( string $haystack , mixed $needle [, bool $before_needle = false ] )
```

내장 함수 `stristr()`은 `strstr()` 함수와 동일한 동작을 하는 함수입니다. 하지만 차이점으로는 대소문자를 구분하지 않습니다.

03.3.2 마지막 검색

내장 함수 `strrchr()`은 `strstr()` 함수와 반대로 마지막 위치를 찾아서 문자열 데이터를 처리합니다. 검색하고자 하는 문자열에 찾을 문자들이 여러 개가 있는 경우 마지막 부분을 반환합니다.

| 내장 함수 |

```
string strrchr ( string $haystack , mixed $needle )
```

예제 파일 | `strchr.php`

```
1 <?php
2     $str = "abcdefghijklem-abcdefghijklem-1234567";
3     echo strrchr($str,"em");
4
5 ?>
```

화면 출력

em-1234567

위 실험에서 원본의 문자열에서 `em` 글자를 찾습니다. `em` 글자는 원본 문자열에 두 번 포함되어 있습니다. 마지막의 `em` 위치를 찾아서 이후의 문자열의 데이터를 반환합니다.

03.3.3 첫 번째 위치

내장 함수 `strpos()`는 문자열에서 검색하고자 하는 글자를 찾아 첫 번째 위치를 반환합니다. 만일 글자를 찾게 되면 문자열의 시작 위치를 정수값으로 반환합니다.

| 내장 함수 |

```
mixed strpos ( string $haystack , mixed $needle [, int $offset = 0 ] )
```

예제 파일 | **strupos.php**

```
1  <?php
2      $string = "abcdefg";
3      $keyword = "cde";
4      if (($pos = strupos($string, $keyword)) === false) {
5          echo "Err] 찾는 문자열이 없습니다.";
6      }
7      } else {
8          echo "문자열 시작 위치 $pos 존재<br>";
9      }
10
11  ?>
```

화면 출력

문자열 시작 위치 2 존재

| 내장 함수 |

```
mixed strupos ( string $haystack , string $needle [, int $offset = 0 ] )
```

내장 함수 `strupos()`는 `strpos()`와 동일한 동작을 합니다. 차이점으로는 검색 시 대소문자를 구분하지 않고 처리합니다.

03.3.4 마지막 위치

내장 함수 `strrpos()`는 문자열에서 검색하고자 하는 글자의 **마지막 위치의 값**을 정수값으로 반환합니다.

| 내장 함수 |

```
int strrpos ( string $haystack , string $needle [, int $offset = 0 ] )
```

예제 파일 | **strrpos.php**

```
1 <?php
2     $str = "abcdefghijklem-abcdefghijklem-1234567";
3     echo "원본 : " . $str . "<br>";
4     $a = strrpos($str,"em-");
5     echo $a;
6
7 ?>
```

화면 출력

원본 : abcdefghijklem-abcdefghijklem-1234567
25

| 내장 함수 |

int **stripos** (string \$haystack , string \$needle [, int \$offset = 0])

내장 함수 stripos()는 strrpos() 함수와 동일하게 동작합니다. 차이점으로는 문자열에서 대소문자를 구분하지 않고 문자열의 마지막 위치를 찾습니다.

03.3.5 마스크 필터

내장 함수 strspn()은 문자열 \$object의 첫 번째 문장에 대해서 마스크 필터링된 길이를 출력합니다.

| 내장 함수 |

int **strspn** (string \$subject , string \$mask [, int \$start [, int \$length]])

예제 파일 | **strspn.php**

```
1 <?php
2     // mask에 맞는 initial segment의 길이를 반환합니다.
3     $object = "423336 is the answer to the 128th question.";
```

```

4     $mask = "1234567890abcdefghijklmnopqrstuvwxyz";
5     $var = strspn($object, $mask);
6
7     echo $var;
8
9     ?>

```

화면 출력

6

내장 함수 `strcspn()`은 `$subject` 문자열에서 `$mask`에 포함되어 있지 않은 문자의 초기 세그먼트 값의 길이를 출력합니다.

| 내장 함수 |

```
int strcspn ( string $subject , string $mask [, int $start [, int $length ]] )
```

예제 파일 | `strcspn.php`

```

1  <?php
2
3     $a = strcspn('abcd', 'apple');
4     var_dump($a); //0
5
6     $b = strcspn('abcd', 'banana');
7     var_dump($b); // 0
8
9     $c = strcspn('hello', 'l');
10    var_dump($c); // 2
11
12    $d = strcspn('hello', 'world');
13    var_dump($d); // 2
14
15    $e = strcspn('abcdhelloabcd', 'abcd', -9);
16    var_dump($e); //5
17
18    $f = strcspn('abcdhelloabcd', 'abcd', -9, -5);
19    var_dump($f); //4
20    ?>

```

화면 출력

```
int(0)
int(0)
int(2)
int(2)
int(5)
int(4)
```

03.3.6 매칭 검색

내장 함수 `strpbrk()`는 임의의 문자 집합에 대한 문자열을 검색합니다. 매칭 검색된 이후의 문자열을 반환합니다.

| 내장 함수 |

```
string strpbrk ( string $haystack , string $char_list )
```

예제 파일 | `strpbrk.php`

```
1  <?php
2
3      $text = 'This is a Simple text.';
4
5      // "is is a Simple text."를 출력합니다.
6      // 처음에 'i'가 먼저 매칭되었기 때문입니다.
7      echo strpbrk($text, 'mi');
8      echo "<br>";
9
10     // "Simple text."를 출력합니다.
11     echo strpbrk($text, 'S');
12
13  ?>
```

화면 출력

```
is is a Simple text.
Simple text.
```


위의 예제는 임의의 문자 \$char_list에 대해서 매칭 검색을 하여 처리합니다. 매칭되는 문자들은 1개 또는 여러 개일 수 있습니다.

03.4 문자열 비교

문자열을 비교하여 처리하는 기능은 문자열 처리 중에서도 비중이 높은 활용도가 있는 부분입니다. PHP는 다양한 문자열 비교 함수를 통하여 보다 쉽게 문자열을 처리할 수 있습니다.

03.4.1 문자열 비교

내장 함수 strcmp()는 바이너리 형태로 2개의 문자열을 비교합니다. 2개의 문자열이 같을 경우 0보다 큰 값을 반환합니다. 문자열을 비교할 때는 대소문자를 구별합니다.

| 내장 함수 |

```
int strcmp ( string $str1 , string $str2 )
```

예제 파일 | **strcmp.php**

```
1  <?php
2      $str1 = "hello";
3      $str2 = "hello";
4      $str3 = "word";
5
6      if (strcmp($str1, $str2) == 0) {
7          echo $str1 . "==" . $str2 . "<br>";
8      } else {
9          echo $str1 . "!=" . $str2 . "<br>";
10     }
11
12     if (strcmp($str2, $str3) == 0 ) {
13         echo $str2 . "==" . $str3 . "<br>";
```

```

14     } else {
15         echo $str2 . "!= " . $str3 . "<br>";
16     }
17
18     ?>

```

화면 출력

```

hello== hello
hello!= word

```

| 내장 함수 |

```
int strcoll ( string $str1 , string $str2 )
```

내장 함수 `strcoll()`은 현재 로케일 기반 문자열을 비교합니다. 현재 로케일이 C 또는 POSIX면 이 함수는 `strcmp()`와 같습니다.

예제 파일 | **strcoll.php**

```

1  <?php
2
3      $a = 'a';
4      $b = 'A';
5
6      print strcmp ($a, $b) . "<br>";
7      // prints 1
8
9      setlocale (LC_COLLATE, 'C');
10     print "Locale based C: " . strcoll ($a, $b) . "<br>";
11     // prints 1
12
13     setlocale (LC_COLLATE, 'de_DE');
14     print "de_DE: " . strcoll ($a, $b) . "<br>";
15
16     setlocale (LC_COLLATE, 'de_CH');
17     print "de_CH: " . strcoll ($a, $b) . "<br>";
18
19     setlocale (LC_COLLATE, 'en_US');

```

```

20     print "en_US: " . strcoll ($a, $b) . "<br>";
21
22     ?>

```

화면 출력

```

1
Locale based C: 1
de_DE: 1
de_CH: 1
en_US: 1

```

03.4.2 문자열 Binary safe

C 언어와 같이 저수준의 언어의 경우 문자열은 메모리 세그먼트를 포인터로 표현합니다. 문자열의 마지막 종료 기호로는 특수 마크로 0바이트 또는 null 바이트를 사용합니다. 따라서 0과 같이 이러한 특수 마크 기호는 문자열에 포함할 수 없습니다.

이러한 점으로 문자열을 저장하는 또 다른 방법으로는 포인터와 문자열의 길이를 함께 저장하는 것입니다. 하지만 이러한 방법은 문자열을 관리하는 데 2개의 값을 사용해야 하기 때문에 문자열을 처리하는 것은 매우 복잡합니다.

| 내장 함수 |

```
int strncmp ( string $str1 , string $str2 , int $len )
```

내장 함수 `strncmp()`는 첫 번째 n 문자의 Binary safe 문자열을 비교합니다. 대소문자 구분합니다. `str1`가 `str2`보다 작은 경우 0보다 작은 값을 반환합니다. `str2`가 `str1`보다 큰 경우에는 >0 값을 반환합니다. 2개의 값이 같은 경우에는 0을 반환합니다.

예제 파일 | `strncmp.php`

```

1  <?php
2      echo strncmp("xybc","a3234",0);
3      // 0
4

```

```

5     echo "<br>";
6
7     echo strcmp("blah123","hohoho", 0);
8     //0
9
10    ?>

```

화면 출력

```

0
0

```

| 내장 함수 |

```
int strcasecmp ( string $str1 , string $str2 )
```

내장 함수 `strcasecmp()`는 Binary safe 유형으로 대소문자를 구분하지 않고 문자열을 비교합니다.

예제 파일 | **strcasecmp.php**

```

1  <?php
2      $var1 = "Hello";
3      $var2 = "hello";
4      if (strcasecmp($var1, $var2) == 0) {
5          echo '$var1과 $var2는 대소문자를 구분하지 않고 동일한 문자열입니다.';
6      }
7
8  ?>

```

화면 출력

`$var1` 와 `$var2` 은 대소문자를 구분하지 않고 동일한 문자열입니다.

| 내장 함수 |

```
int substr_compare ( string $main_str , string $str , int $offset [, int $length [, bool  
$case_insensitivity = false ]])
```

내장 함수 `substr_compare()`는 바이너리 세이프 형태로 오프셋의 두 문자열 비교, 최대 길이 문자를 비교합니다. `substr_compare()`는 오프셋 위치에서 `main_str`을 `str`과 최대 길이 문자를 비교합니다.

예제 파일 | `substr_compare.php`

```
1  <?php
2      echo substr_compare("abcde", "bc", 1, 2); // 0
3      echo substr_compare("abcde", "de", -2, 2); // 0
4      echo substr_compare("abcde", "bcg", 1, 2); // 0
5      echo substr_compare("abcde", "BC", 1, 2, true); // 0
6      echo substr_compare("abcde", "bc", 1, 3); // 1
7      echo substr_compare("abcde", "cd", 1, 2); // -1
8      echo substr_compare("abcde", "abc", 5, 1); // warning
9
10  ?>
```

화면 출력

```
0
0
0
0
1
-1
```

03.4.3 자연 순서

자연 순서 알고리즘을 통하여 문자열을 비교하고 처리할 수 있습니다.

| 내장 함수 |

```
int strnatcmp ( string $str1 , string $str2 )
```

내장 함수 `strnatcmp()`는 '자연 순서' 알고리즘을 사용하여 문자열을 비교합니다.

예제 파일 | `strnatcmp.php`

```
1  <?php
2      $arr1 = $arr2 = array("img12.png", "img10.png", "img2.png", "img1.
    png");
3      echo "Standard string comparison\n";
4      usort($arr1, "strcmp");
5      print_r($arr1);
6
7      echo "\nNatural order string comparison\n";
8      usort($arr2, "strnatcmp");
9      print_r($arr2);
10  ?>
```

화면 출력

```
Standard string comparison Array ( [0] => img1.png [1] => img10.png [2] =>
img12.png [3] => img2.png )
Natural order string comparison Array ( [0] => img1.png [1] => img2.png [2] =>
img10.png [3] => img12.png )
```

| 내장 함수 |

```
int strnatcasecmp ( string $str1 , string $str2 )
```

내장 함수 `strnatcasecmp()`는 '자연 순서' 알고리즘을 적용합니다. 대소문자를 구분하지 않는 문자열 비교합니다.

03.4.4 유사성

문자열의 유사성을 비교하고 검사합니다.

| 내장 함수 |

```
int similar_text ( string $first , string $second [, float &$percent ] )
```

내장 함수 `similar_text()`는 2개의 문자의 비슷한 정도를 계산합니다.

예제 파일 | **similar_text.php**

```
1  <?php
2      $str1 = "안녕하세요! 지니 입니다.";
3      $str2 = "안녕하세요! jiny 입니다.";
4
5
6      similar_text($str1, $str2, $percent);
7      echo "두 문장의 유사도는 $percent % 입니다.<br>";
8
9      similar_text($str2, $str1, $percent);
10     echo "두 문장의 유사도는 $percent % 입니다.<br>";
11
12  ?>
```

화면 출력

두 문장의 유사도는 84.848484848485 % 입니다.

두 문장의 유사도는 84.848484848485 % 입니다.

| 내장 함수 |

```
string soundex ( string $str )
```

내장 함수 `soundex()`는 문자열의 `soundex` 키 값을 반환합니다.

예제 파일 | **soundex.php**

```
1  <?php
2      echo "Euler ".soundex("Euler") . " == " .
3          "Ellery ". soundex("Ellery") . "<br>";
4
5      echo "Gauss ".soundex("Gauss") . " == " .
6          "Ghosh ". soundex("Ghosh") . "<br>";
7
8      echo "Hilbert ".soundex("Hilbert") . " == " .
9          "Heilbronn ". soundex("Heilbronn") . "<br>";
10
11  ?>
```

화면 출력

```
Euler E460 == Ellery E460
Gauss G200 == Ghosh G200
Hilbert H416 == Heilbronn H416
```

내장 함수 `levenshtein()`은 두 문자열 사이의 Levenshtein 거리를 반환합니다. Levenshtein 거리는 문자열 `str1`을 `str2`로 변환하기 위해 교체하거나 삽입 또는 삭제해야 하는 최소 문자 수를 말합니다.

| 내장 함수 |

```
int levenshtein ( string $str1 , string $str2 )
```

예제 파일 | **levenshtein.php**

```
1  <?php
2      // 당근 : carrot의 스펠링을 잘못 입력합니다.
3      $input = 'carrrot';
4
5      // 단어 데이터들
6      $words = array('apple','pineapple','banana','orange',
7                     'radish','carrot','pea','bean','potato');
8
```



```

9      $shortest = -1;
10
11      // 입력한 단어를 비교합니다.
12      foreach ($words as $word) {
13
14          $lev = levenshtein($input, $word);
15
16          // 입력한 단어가 일치할 때
17          if ($lev == 0) {
18
19              // 일치한 단어 설정
20              $closest = $word;
21              $shortest = 0;
22
23              // 단어가 일치하기 때문에, 반복문 종료
24              break;
25          }
26
27          if ($lev <= $shortest || $shortest < 0) {
28              $closest = $word;
29              $shortest = $lev;
30          }
31      }
32
33      echo "입력 단어: $input <br>";
34      if ($shortest == 0) {
35          echo "정확한 단어: $closest <br>";
36      } else {
37          echo "혹시 원하는 단어가 $closest 입니까? <br>";
38      }
39
40      ?>

```

화면 출력

입력 단어: carrrot

혹시 원하는 단어가 carrot 입니까?

| 내장 함수 |

```
string metaphone ( string $str [, int $phonemes = 0 ] )
```

내장 함수 `metaphone()`은 문자열의 메타폰 키를 계산합니다.

예제 파일 | `metaphone.php`

```
1  <?php
2      var_dump(metaphone('programming'));
3      echo "<br>";
4      var_dump(metaphone('programmer'));
5      echo "<br>";
6      var_dump(metaphone('programming', 5));
7      echo "<br>";
8      var_dump(metaphone('programmer', 5));
9      echo "<br>";
10
11  ?>
```

화면 출력

```
string(7) "PRKRMNK"
string(6) "PRKRMR"
string(5) "PRKRM"
string(5) "PRKRM"
```

03.5 문자열 치환

치환이란 특정 문자열의 내용을 다른 문자열로 대체한다는 것입니다. 문자열 치환은 다양한 문자열 출력 결과물을 만들어 처리하는 데 매우 유용한 기능입니다.

문자열의 내용을 치환하는 알고리즘은 복잡합니다. PHP는 문자열 치환 함수들을 통하여 보다 간단하게 처리할 수 있습니다.

| 내장 함수 |

```
string strtr ( string $str , string $from , string $to )
```

내장 함수 `strtr()`은 특정 문자열을 대체합니다. 참고로 문자열을 대체할 때는 문자열의 길이가 같아야 합니다. 만일 대체하고자 하는 문자열의 길이가 더 클 때는 이후 문자열은 무시됩니다.

예제 파일 | `strtr.php`

```
1  <?php
2      $str = "안녕하세요. jiny 입니다.!!";
3      echo $str."<br>";
4
5      $src = "jiny";
6      $dst = "hojinlee";
7
8      echo strtr($str, $src, $dst);
9
10  ?>
```

화면 출력

```
안녕하세요. jiny 입니다.!!
안녕하세요. hoji 입니다.!!
```

| 내장 함수 |

```
mixed str_replace ( mixed $search , mixed $replace , mixed $subject [, int &$amp;count ] )
```

내장 함수 `str_replace()`는 문자열 내의 특정 문자열을 검색하여 다른 문자열로 대체합니다.

예제 파일 | `str_replace.php`

```
1  <?php
2      $string = "abcdefg";
```

```

3     $keyword = "cde";
4
5     $body = str_replace($keyword, " 11111 ", $string);
6     echo $body;
7
8     ?>

```

화면 출력

ab 11111 fg

| 내장 함수 |

mixed **str_ireplace** (mixed \$search , mixed \$replace , mixed \$subject [, int &\$count])

내장 함수 str_ireplace()는 str_replace()의 대소문자를 구별하지 않는 버전입니다.

| 내장 함수 |

mixed **substr_replace** (mixed \$string , mixed \$replacement , mixed \$start [, mixed \$length])

내장 함수 substr_replace()는 문자열의 일부분 내에서 텍스트를 바꿉니다.

예제 파일 | [substr_replace.php](#)

```

1  <?php
2      $var = 'ABCDEFGH:/MNRPQR/';
3      echo "원본: $var<hr/>\n";
4
5      /* 전제 문자열을 'bob'으로 변경합니다. */
6      echo substr_replace($var, 'bob', 0) . "<br />\n";
7      echo substr_replace($var, 'bob', 0, strlen($var)) . "<br />\n";
8
9      /* 문자열 앞에 'bob'을 추가합니다. */
10     echo substr_replace($var, 'bob', 0, 0) . "<br />\n";
11
12     /* 'MNRPQR' 부분을 'bob'으로 바꾸기 합니다. */

```

```

13     echo substr_replace($var, 'bob', 10, -1) . "<br />\n";
14     echo substr_replace($var, 'bob', -7, -1) . "<br />\n";
15
16     /* 'MNRPQR' 부분을 삭제합니다. */
17     echo substr_replace($var, '', 10, -1) . "<br />\n";
18
19     ?>

```

화면 출력

```

원본: ABCDEFGH:/MNRPQR/_____bob
bob
bobABCDEFGH:/MNRPQR/
ABCDEFGH:/bob/
ABCDEFGH:/bob/
ABCDEFGH://

```

03.6 문자

문자는 문자열을 구성하는 요소입니다. PHP는 이러한 문자들을 표현하고 처리할 수 있는 다양한 함수를 제공합니다.

| 내장 함수 |

```
string chr ( int $ascii )
```

내장 함수 `chr()`은 아스키 코드값에 대한 문자를 출력합니다.

예제 파일 | `chr.php`

```

1  <?php
2      // 아스키 코드
3      echo "27 = ".chr(27). "<br>";
4      echo "65 = ".chr(65). "<br>";
5      echo "92 = ".chr(92). "<br>";
6

```

```
7  ?>
```

화면 출력

```
27 =  
65 = A  
92 = \
```

| 내장 함수 |

```
int ord ( string $string )
```

내장 함수 ord()는 입력한 문자의 아스키 코드값을 출력합니다.

예제 파일 | ord.php

```
1  <?php  
2      // 아스키 코드  
3      echo "27 = ".chr(27)."<br>";  
4      echo "65 = ".chr(65)."<br>";  
5      echo "92 = ".chr(92)."<br>";  
6  
7      echo "<br>";  
8  
9      echo "A = ".ord('A')."<br>";  
10     echo "+ = ".ord('+')."<br>";  
11     echo "% = ".ord('%')."<br>";  
12  
13  ?>
```

화면 출력

```
27 =  
65 = A  
92 = \
```

```
A = 65  
+ = 43  
% = 37
```

| 내장 함수 |

```
string str_repeat ( string $input , int $multiplier )
```

내장 함수 `str_repeat()`는 문자열을 지정한 횟수만큼 반복합니다.

예제 파일 | `str_repeat.php`

```
1  <?php
2      echo str_repeat("=", 10);
3
4  ?>
```

화면 출력

=====

| 내장 함수 |

```
string str_pad ( string $input , int $pad_length [, string $pad_string = " " [, int $pad_
type = STR_PAD_RIGHT ] ] )
```

내장 함수 `str_pad()`는 문자열을 다른 문자열로 특정 길이를 채웁니다.

예제 파일 | `str_pad.php`

```
1  <?php
2      $input = "hojin";
3      echo str_pad($input, 10) . "<br>";
4      echo str_pad($input, 10, "=", STR_PAD_LEFT) . "<br>";
5      echo str_pad($input, 10, "_", STR_PAD_BOTH) . "<br>";
6      echo str_pad($input, 6, "___") . "<br>";
7      echo str_pad($input, 3, "*") . "<br>";
8
9  ?>
```

화면 출력

```
hojin
----hojin
__hojin__
hojin_
hojin
```

03.7 구분화

구분화는 문자열을 특정한 규칙을 통하여 분리하는 기능입니다. 여러 개의 문자열 데이터를 특정 키를 기준으로 연결되어 있을 경우 이를 처리하여 구분할 수 있습니다.

03.7.1 토큰

내장 함수 `strtok()`는 문자열을 주어진 키로 토큰화합니다.

| 내장 함수 |

```
string strtok ( string $str , string $token )
```

예제 파일 | `strtok.php`

```
1  <?php
2      $str = "안녕하세요! 지니 PHP 코딩입니다.";
3
4      // 공백 문자로 문자열을 토큰화합니다.
5      $aaa = strtok($str, " ");
6
7      $i=0;
8
9      while ($aaa) {
10         echo $i++ . " = " . $aaa . "<br>";
11         $aaa = strtok(" ");
12     }
13
14  ?>
```


화면 출력

0= 안녕하세요!
1= 지니
2= PHP
3= 코딩
4= 입니다.

| 내장 함수 |

```
array explode ( string $delimiter , string $string [, int $limit = PHP_INT_MAX ] )
```

내장 함수 `explode()`는 주어진 문자열을 구분자로 구분하여 배열로 변환합니다.

| 내장 함수 |

```
string implode ( string $glue , array $pieces )
```

내장 함수 `implode()`는 반대로 배열을 연결하여 문자열로 반환합니다.

예제 파일 | `implode.php`

```
1  <?php
2
3      $string = "aaa;bbb;ccc;ddd;eee";
4      $arr = explode(";", $string);
5
6      foreach ($arr as $key => $value) {
7          echo $key."=", $value."<br>";
8      }
9
10     $msg = implode(", ", $arr);
11     echo $msg;
12
13  ?>
```

화면 출력

```
0=aaa
1=bbb
2=ccc
3=ddd
4=eee
aaa,bbb,ccc,ddd,eee
```

03.7.2 문자열 분리

내장 함수 `chunk_split()`는 문자열을 비슷한 작은 크기로 분리합니다. `base64_encode()` 출력을 RFC 2045에 맞게 변환할 때 유용하게 사용할 수 있습니다. 모든 문자마다 종료 시퀀스 `"\r\n"`을 삽입합니다.

| 내장 함수 |

```
string chunk_split ( string $body [, int $chunklen = 76 [, string $end = "\r\n" ] ] )
```

예제 파일 | `chunk_split.php`

```
1 <?php
2     $str = "abcdefghijklem";
3     echo chunk_split($str, 4) . "<br>";
4
5 ?>
```

화면 출력

```
abcd efgh ijkl em
```

| 내장 함수 |

```
array str_split ( string $string [, int $split_length = 1 ] )
```

내장 함수 `str_split()`는 문자열을 배열로 변환합니다.

예제 파일 | **str_split.php**

```
1  <?php
2
3      $str = "Hello Friend";
4
5      // 한 글자씩 배열로 변환합니다.
6      $arr1 = str_split($str);
7      print_r($arr1);
8      echo "<br>";
9
10     // 세 글자씩 배열로 변환합니다.
11     $arr2 = str_split($str, 3);
12     print_r($arr2);
13
14  ?>
```

화면 출력

```
Array ( [0] => H [1] => e [2] => l [3] => l [4] => o [5] => [6] => F [7] => r [8]
=> i [9] => e [10] => n [11] => d )
Array ( [0] => Hel [1] => lo [2] => Fri [3] => end )
```

03.7.3 래핑

내장 함수 `wordwrap()`은 문자열을 주어진 문자 수로 래핑합니다.

| 내장 함수 |

```
string wordwrap ( string $str [, int $width = 75 [, string $break = "\n" [, bool $cut =
false ]]] )
```

예제 파일 | **wordwrap.php**

```
1  <?php
2      $text = "The quick brown fox jumped over the lazy dog.";
3      $newtext = wordwrap($text, 20, "<br />\n");
4
```

```

5      echo $newtext;
6
7      ?>

```

화면 출력

The quick brown fox
jumped over the lazy
dog.

03.7.4 변수 해석

내장 함수 `parse_str()`은 입력된 하나의 문자열을 변수로 해석합니다. 각각의 변수는 &로 구분하며, 변수명=값 형태로 지정할 수 있습니다.

| 내장 함수 |

```
void parse_str ( string $encoded_string [, array &$result ] )
```

예제 파일 | `parse_str.php`

```

1  <?php
2      $str = "name[]=jiny&name[]=lee&country=korea";
3      parse_str($str);
4
5      echo $country . "<br>";
6
7      echo $name[0]."<br>";
8      echo $name[1]."<br>";
9
10  ?>

```

화면 출력

korea
jiny
lee

03.8 문자열 조작

컴퓨터와 프로그램 언어들은 초기 영문권을 중심으로 개발이 되면서 대부분의 언어 처리는 알파벳으로 구성되어 있습니다. 알파벳은 특성상 대문자와 소문자로 구분됩니다.

PHP는 이러한 알파벳 문자의 특징을 처리할 수 있는 함수들을 지원합니다.

03.8.1 문자열 순서

내장 함수 `strrev()`는 입력된 문자열의 순서를 반대로 바꿉니다.

| 내장 함수 |

```
string strrev ( string $string )
```

예제 파일 | `strrev.php`

```
1  <?php
2      $str = "abcdefghijklem-abcdefghijklem-1234567";
3      echo "원본 : " . $str . "<br>";
4      $a = strrev($str);
5      echo $a;
6
7  ?>
```

화면 출력

```
원본 : abcdeghijklem-abcdeghijklem-1234567
7654321-melkjihgedcba-melkjihgedcba
```

03.8.2 대소문자

알파벳의 대소문자들을 변경할 수 있습니다.

| 내장 함수 |

```
string strtolower ( string $string )
```

내장 함수 `strtolower()`, `mb_strlen()`는 알파벳 문자열을 소문자로 변경합니다.

| 내장 함수 |

```
string strtoupper ( string $string )
```

내장 함수 `strtoupper()`, `mb_strtoupper()`는 알파벳 문자열을 대문자로 변경합니다.

예제 파일 | [strtolower.php](#)

```
1  <?php
2      $lower = "ABCD";
3      echo $lower. "=" . strtolower($lower). "<br>";
4
5      $upper = "abcd";
6      echo $upper. "=" . strtoupper("abcd"). "<br>";
7
8  ?>
```

화면 출력

```
ABCD=abcd
abcd=ABCD
```

03.8.3 낙타 표기

알파벳의 단어나 문장들을 단어의 첫 글자를 대문자로 표기하는 낙타 표기법을 사용합니다. PHP는 단어의 낙타 표기법을 변환할 수 있는 함수들을 제공합니다.

| 내장 함수 |

```
string ucwords ( string $str [, string $delimiters = "\t\r\n\f\v" ] )
```

내장 함수 `ucwords()`는 단어를 낙타 표기법처럼 각각의 단어를 대문자로 변환합니다.

| 내장 함수 |

```
string ucfirst ( string $str )
```

내장 함수 `ucfirst()`는 전체 문자열 중에서 첫 글자만 대문자로 변환합니다.

예제 파일 | `ucwords.php`

```
1  <?php
2      $string = "abcd ergh ijk l mnop";
3
4      // 첫 단어를 대문자로 변경합니다.
5      echo ucwords($string) . "<br>";
6
7      // 문자열 전체에서 첫 단어만 대문자로 변경합니다.
8      echo ucfirst($string) . "<br>";
9
10 ?>
```

화면 출력

```
Abcd Ergh Ijk l Mnop
Abcd ergh ijk l mnop
```

| 내장 함수 |

```
string lcfirst ( string $str )
```

내장 함수 `lcfirst()`는 문자열의 첫 번째 문자를 소문자로 변환합니다.

예제 파일 | **lcfirst.php**

```
1 <?php
2     $str = 'HelloWorld';
3     $str = lcfirst($str);
4     echo $str;
5
6 ?>
```

화면 출력

helloWorld

03.9 변환

정수형 자료의 경우 숫자의 값을 가지고 있습니다. 실수형의 자료의 경우도 실수의 숫자 값을 가지고 있습니다. 하지만 정수형, 실수형의 표현은 문자로도 출력이 가능합니다.

정수값, 실수값을 문자로 정확하게 표기하기 위해서는 내부 숫자 변환 함수를 이용하면 편리합니다.

03.9.1 숫자 표기

내장 함수 `number_format()`은 입력된 문자열을 기준으로 그룹화된 숫자 서식 형태로 변경할 수 있습니다. 함수의 입력 매개변수는 1개, 2개, 4개로 전달합니다.

| 내장 함수 |

```
string number_format ( float $number [, int $decimals = 0 ] )
```

기본적으로 매개변수 1개만 입력되는 경우 천 단위 표기로 변경된 포맷을 출력합니다. 두 번째 인자값은 소수점 자리수를 의미합니다.

세 번째 인자와 네 번째 인자는 같이 한 쌍으로 입력해야 합니다. 세 번째는 소수점 표기

기호, 네 번째는 천 단위 표시 기호입니다.

예제 파일 | `number_format.php`

```
1  <?php
2
3      $number = 1234.56;
4
5      // 기본
6      // 매개인자 1개만 전달할 경우 천 단위 구분자 쉼표(,)로 포맷 변경됩니다.
7      echo number_format($number) . "<br>";
8      // 1,235
9
10     // 두 번째 매개변수는 소수점 자리수
11     echo number_format($number,5) . "<br>";
12     // 1,234.56000
13
14     // 세 번째 인자는 = 소수점 표기 기호
15     // 네 번째 인자는 = 천 단위 표기 기호
16     echo number_format($number, 2, ',', ' ') . "<br>";
17     // 1 234,56
18
19     $number = 1234.5678;
20     echo number_format($number, 2, '.', ',');
21     // 1234.57
22
23  ?>
```

화면 출력

```
1,235
1,234.56000
1 234,56
1,234.57
```

03.9.2 통화 표시

경제학, 금융 쪽에서 사용되는 숫자는 통화로 사용되는 경우가 많습니다. 숫자를 통화로 표기하는 것은 각각의 나라마다 표현하고 처리하는 방법이 다릅니다.

내장 함수 `money_format()`은 통화 형식의 문자열을 반환합니다. 참고로 윈도우 환경에서는 `money_format()`을 사용할 수 없습니다.

| 내장 함수 |

```
string money_format ( string $format , float $number )
```

예제 파일 | `money_format.php`

```
1  <?php
2
3      $number = 1234.56;
4
5      // 로컬 설정 en_US
6      setlocale(LC_MONETARY, 'en_US');
7      echo "en_US". money_format(' %i', $number) . "<br>";
8      // USD 1,234.56
9
10     // 이탈리아 포맷 2 decimals`
11     setlocale(LC_MONETARY, 'it_IT');
12     echo money_format(' %.2n', $number) . "\n";
13     // Eu 1.234,56
14
15     // 음수 값
16     $number = -1234.525234;
17
18     // US 포맷
19     // 왼쪽 정밀도의 경우 10 자리
20     setlocale(LC_MONETARY, 'en_US');
21     echo money_format(' %(#10n', $number) . "\n";
22     // ($      1,234.57)
23
24     // 위 형식과 비슷한 형식으로 2 자리의 오른쪽 자리 정밀도와
25     // '*'를 채우기 문자로 추가합니다.
26     echo money_format(' %=(#10.2n', $number) . "\n";
27     // ($*****1,234.57)
28
29
30     // 왼쪽에서 14 자리의 너비, 8 자리의 왼쪽 자릿수, 2 자리의 오른쪽 자릿수
```

```

31 // without 그룹화 문자 및 de_DE 로케일의 국제 형식을 사용합니다.
32 setlocale(LC_MONETARY, 'de_DE');
33 echo money_format('%*^~14#8.2i', 1234.56) . "\n";
34 // Eu 1234,56****
35
36
37 // 전환 지정 전후에 몇 가지 문안을 추가하겠습니다.
38 setlocale(LC_MONETARY, 'en_GB');
39 $fmt = 'The final value is %i (after a 10%% discount)';
40 echo money_format($fmt, 1234.56) . "\n";
41 // The final value is GBP 1,234.56 (after a 10% discount)
42
43 ?>

```

03.10 인코딩

글로벌화된 소프트웨어의 개발로 인하여 다국어 처리와 다양한 문자 언어셋을 지원하는 것은 중요합니다. 문자열은 다양한 문자 언어셋으로 처리됩니다.

PHP는 문자열의 언어 인코딩을 변경할 수 있는 몇 가지 함수를 제공하고 있습니다.

| 내장 함수 |

```
string iconv ( string $in_charset , string $out_charset , string $str )
```

내장 함수 `iconv()`는 문자 인코딩을 변환합니다.

예제 파일 | `iconv.php`

```

1 <?php
2 $text = "This is the Euro symbol '€'.";
3
4 echo 'Original : ', $text, "<br>";
5 echo 'TRANSLIT : ', iconv("UTF-8", "ISO-8859-1//TRANSLIT", $text),
  "<br>";
6 echo 'IGNORE : ', iconv("UTF-8", "ISO-8859-1//IGNORE", $text),

```

```
"<br>";
```

```
7 ?>
```

화면 출력

Original : This is the Euro symbol '€'.
TRANSLIT : This is the Euro symbol 'EUR'.
IGNORE : This is the Euro symbol ''.

| 내장 함수 |

```
string convert_uuencode ( string $data )
```

내장 함수 `convert_uuencode()`는 문자열을 uuencode 알고리즘으로 인코딩합니다.

예제 파일 | `convert_uuencode.php`

```
1 <?php
2     $string = "test\ntext text\r\n";
3     echo convert_uuencode($string);
4
5 ?>
```

화면 출력

0=&5S=`IT97AT('1E>'0-"@`` `

| 내장 함수 |

```
string convert_uudecode ( string $data )
```

내장 함수 `convert_uudecode()`는 uuencode된 문자열을 디코딩합니다.

예제 파일 | `convert_uudecode.php`

```
1 <?php
2     echo convert_uudecode("+22!L;w9E(%!(4\"$`\\n`");
```

```
3
4  ?>
```

화면 출력

I love PHP!

| 내장 함수 |

```
string quoted_printable_encode ( string $str )
```

내장 함수 `quoted_printable_encode()`는 8비트 문자열, 따옴표 붙은 인쇄 가능한 문자열로 변환합니다.

| 내장 함수 |

```
string quoted_printable_decode ( string $str )
```

내장 함수 `quoted_printable_decode()`는 `quoted-printable` 문자열을 8비트 문자열로 변경합니다.

| 내장 함수 |

```
string convert_cyr_string ( string $str , string $from , string $to )
```

내장 함수 `convert_cyr_string()`은 하나의 Cyrillic 문자 집합을 다른 집합으로 변환합니다.

03.11 랜덤

랜덤이란 난수를 발행하는 알고리즘입니다.

내장 함수 `random_bytes()`는 시스템을 통하여 랜덤 바이트의 난수를 생성합니다.

| 내장 함수 |

```
string random_bytes ( int $length )
```

예제 파일 | [random_bytes.php](#)

```
1  <?php
2      $bytes = random_bytes(5);
3      var_dump(bin2hex($bytes));
4
5  ?>
```

화면 출력

```
string(10) "fab26daa1e"
```

| 내장 함수 |

```
string str_shuffle ( string $str )
```

내장 함수 `str_shuffle()`은 입력한 문자열을 무작위로 섞습니다.

예제 파일 | [str_shuffle.php](#)

```
1  <?php
2      $str = 'abcdef';
3      $shuffled = str_shuffle($str);
4
5      echo $shuffled;
6
7  ?>
```

화면 출력

```
efbcda
```

03.12 해시 및 암호화

문자열은 우리가 일반적으로 읽기 쉬운 형태의 문장들입니다. 이러한 문장들은 데이터를 처리하는 데는 매우 편리합니다. 하지만 데이터를 저장하거나 외부로 전송할 때 문장 그대로 처리하다 보면 보안상 문제가 발생할 수 있습니다.

기본적으로 문자열을 보안 처리하는 방법으로는 암호화가 있습니다. PHP는 다양한 해시 및 암호화 모듈이 내장되어 있습니다. PHP 환경 설정 부분을 확인해 보면 설치되어 있는 목록을 볼 수 있습니다.

```
md2 md4 md5 sha1 sha224 sha256 sha384 sha512/224 sha512/256 sha512 sha3-224
sha3-256 sha3-384 sha3-512 ripemd128 ripemd160 ripemd256 ripemd320 whirlpool
tiger128,3 tiger160,3 tiger192,3 tiger128,4 tiger160,4 tiger192,4 snefru snefru256 gost
gost-crypto adler32 crc32 crc32b fnv132 fnv1a32 fnv164 fnv1a64 joaat haval128,3
haval160,3 haval192,3 haval224,3 haval256,3 haval128,4 haval160,4 haval192,4
haval224,4 haval256,4 haval128,5 haval160,5 haval192,5 haval224,5 haval256,5
```

03.12.1 해시

해시는 가장 기초적인 문자열 암호화 방식입니다.

| 내장 함수 |

```
string hash ( string $algo , string $data [, bool $raw_output = false ] )
```

내장 함수 `hash()`는 해시값을 생성합니다. 첫 번째 인자 `$algo`는 알고리즘의 타입을 선택합니다. 예로 'md5', 'sha256', 'haval160,4' 등 두 번째 인자인 `$data`는 해시를 적용할 메시지입니다. 세 번째 인자인 `$raw_output`은 true일 때는 바이너리 형식, false는 소문자 hex로 반환합니다.

예제 파일 | [hash.php](#)

```
1 <?php
```

```

2      echo hash('ripemd160', 'hello world php!');
3
4      ?>

```

화면 출력

271ab7cf2239daa049877e8c6fc73cfc8097d0de

| 내장 함수 |

```
resource hash_init ( string $algo [, int $options = 0 [, string $key = NULL ]] )
```

내장 함수 `hash_init()`는 추가된 문맥에 대해서 해시를 초기화합니다.

- 옵션: `HASH_HMAC`. 지정되면 키를 지정해야 합니다.
- 키: 옵션 `HASH_HMAC`가 지정되면 HMAC 해시 메서드와 함께 사용할 공유 비밀 키를 매개변수로 제공해야 합니다.

예제 파일 | `hash_init.php`

```

1  <?php
2      $ctx = hash_init('md5');
3      hash_update($ctx, 'hello world ');
4      hash_update($ctx, 'jinyPHP. ');
5      echo hash_final($ctx);
6
7      ?>

```

화면 출력

c4009628bfac77c4060e51d14bd417a5

| 내장 함수 |

```
array hash_algos ( void )
```

내장 함수 `hash_algos()`는 등록된 해시 알고리즘 목록을 반환합니다.

예제 파일 | [hash_algos.php](#)

```
1 <?php
2     print_r(hash_algos());
3
4 ?>
```

화면 출력

```
Array ( [0] => md2 [1] => md4 [2] => md5 [3] => sha1 [4] => sha224 [5] =>
sha256 [6] => sha384 [7] => sha512/224 [8] => sha512/256 [9] => sha512 [10] =>
sha3-224 [11] => sha3-256 [12] => sha3-384 [13] => sha3-512 [14] => ripemd128
[15] => ripemd160 [16] => ripemd256 [17] => ripemd320 [18] => whirlpool [19]
=> tiger128,3 [20] => tiger160,3 [21] => tiger192,3 [22] => tiger128,4 [23] =>
tiger160,4 [24] => tiger192,4 [25] => snefru [26] => snefru256 [27] => gost [28]
=> gost-crypto [29] => adler32 [30] => crc32 [31] => crc32b [32] => fnv132 [33]
=> fnv1a32 [34] => fnv164 [35] => fnv1a64 [36] => joaat [37] => haval128,3 [38]
=> haval160,3 [39] => haval192,3 [40] => haval224,3 [41] => haval256,3 [42] =>
haval128,4 [43] => haval160,4 [44] => haval192,4 [45] => haval224,4 [46] =>
haval256,4 [47] => haval128,5 [48] => haval160,5 [49] => haval192,5 [50] =>
haval224,5 [51] => haval256,5 )
```

03.12.2 MD5

내장 함수 md5()는 RFC1321 기준 md5 해시 알고리즘을 제공합니다.

| 내장 함수 |

```
string md5 ( string $str [, bool $raw_output = false ] )
```

예제 파일 | [md5.php](#)

```
1 <?php
2     $password = "abcd1234";
3     echo "password = " . $password . "<br>";
4
5     echo "MD5 = " . md5($password) . "<br>";
6     echo "MD5 = " . md5($password) . "<br>";
7
```

```

8      echo "랜덤생성 예 === <br>";
9      echo "랜덤 MD5 = " . md5(mt_rand()) . "<br>";
10
11  ?>

```

화면 출력

```

password = abcd1234
MD5 = e19d5cd5af0378da05f63f891c7467af
MD5 = e19d5cd5af0378da05f63f891c7467af
랜덤생성 예 ===
랜덤 MD5 = bea084f3108d3fa7ce4f6826097e2cd8

```

| 내장 함수 |

```
string md5_file ( string $filename [, bool $raw_output = false ] )
```

내장 함수 md5_file()은 주어진 파일에 대해서 MD5 해시값을 계산합니다.

예제 파일 | md5_file.php

```

1  <?php
2      $file = 'md5_file.php';
3      echo 'MD5 file hash of ' . $file . ': ' . md5_file($file);
4
5  ?>

```

화면 출력

```
MD5 file hash of md5_file.php: 25adcad58baa2a626dfa53b98dff0995
```

03.12.3 crc32

내장 함수 crc32()는 문자열에 대해서 CRC32 polynomial 계산을 수행합니다.

| 내장 함수 |

```
int crc32 ( string $str )
```

CRC32는 보통 데이터 전송 시 무결성 검증을 위해서 사용합니다. 입력된 문자열 스트링의 32비트 순환 중복 검사에 대한 결과를 출력합니다.

예제 파일 | **crc32.php**

```
1 <?php
2     $checksum = crc32("hello php world");
3     printf("%u\n", $checksum);
4
5 ?>
```

화면 출력

2202403677

03.12.4 sha1

내장 함수 sha1()은 문자열에 대해서 sha1 해시 계산을 처리합니다.

| 내장 함수 |

```
string sha1 ( string $str [, bool $raw_output = false ] )
```

raw_output이 true인 경우에는 길이가 20인 원시 바이너리 형식을 반환합니다. false인 경우에는 40문자 16진수를 반환합니다.

예제 파일 | **sha1.php**

```
1 <?php
2     $str = 'apple';
3
4     // 40문자 16진수
5     echo $str . " = ". sha1($str). "<br>";
6
7     // 길이가 20인 원시 바이너리 형식
8     echo $str . " = ". sha1($str,true). "<br>";
9 ?>
```

화면 출력

```
apple = d0be2dc421be4fcd0172e5afceea3970e2f3d940  
apple = o-◆!◆0◆r◆◆◆◆9p◆◆◆@
```

| 내장 함수 |

```
string sha1_file ( string $filename [, bool $raw_output = false ] )
```

내장 함수 sha1_file()은 주어진 파일에 대해서 SHA1 해시를 계산합니다.

예제 파일 | sha1_file.php

```
1  <?php  
2  
3      foreach(glob('*.exe') as $ent)  
4      {  
5          if (is_dir($ent))  
6          {  
7              continue;  
8          }  
9  
10         echo $ent . ' = (SHA1: ' . sha1_file($ent) . ' )' . "<br>";  
11     }  
12  
13  ?>
```

화면 출력

```
deplister.exe = (SHA1: 5aeb27623d25d042e101bb64ca011308cf2aa785)  
php-cgi.exe = (SHA1: 5cdb18117c91de9db5616c8141456dff63dc4a75)  
php-win.exe = (SHA1: 342aa529b6bf06b34e15ee7d3fef4dc87ee6199c)  
php.exe = (SHA1: aeee36515446efd8ca4fccddc5e7b277f0fb217c)  
phpdbg.exe = (SHA1: b1af4e81d2a146d9e3bd047c2a44b06b65999034)
```

03.12.5 crypt

내장 함수 crypt() 함수는 표준 유닉스 DES 형태로 단방향 암호화된 문자열을 반환합니다.

| 내장 함수 |

```
string crypt ( string $str [, string $salt ] )
```

운영체제별로 암호화 방식은 약간씩 다른데, MD5로 대체하여 처리하기도 합니다. 암호화 작업 시 기본 문자열 이외에 암호키(salt)를 사용할 수 있습니다.

예제 파일 | **crypt.php**

```
1  <?php
2
3      $password = "ABCD1234";
4      echo "암호 = " . $password . "<br>";
5
6      // salt 자동 생성
7      echo "DES 기반 암호 =" . crypt('ABCD1234') . "<br>";
8
9      // 사용자 salt
10     $salt = "복잡한 암호키입니다.";
11     echo "DES 기반 암호 =" . crypt('ABCD1234',$salt) . "<br>";
12
13  ?>
```

화면 출력

암호 = ABCD1234

DES 기반 암호 =\$1\$Lhg1VRxu\$bLQyHdT/Cja/XbSgiNgGq.

DES 기반 암호 =◆◆mEIN4PXgIk.

03.12.6 str_rot13

내장 함수 ROT13(Rotate by 13)은 카이사르 암호 형식입니다. 알파벳에 13을 밀어서 표기를 합니다. 즉 A 문자는 13을 더한 N 문자로 표기합니다. str_rot13() 함수는 입력된 문자열에 대해서 ROT13 암호화 작업을 수행합니다.

| 내장 함수 |

```
string str_rot13 ( string $str )
```

예제 파일 | [str_rot13.php](#)

```
1  <?php
2      echo str_rot13('PHP 4.3.0'); // CUC 4.3.0
3
4  ?>
```

화면 출력

CUC 4.3.0

03.13 문자열 출력

PHP에서 변수의 데이터 값을 출력할 수 있는 방법은 다양합니다. 간단하게 결과를 출력하는 echo 함수뿐만 아니라 다양한 포맷을 통하여 출력할 수 있는 몇 가지 함수를 제공합니다.

03.13.1 출력

내장 함수 echo()는 문자열을 출력합니다. php에서 가장 기본적이고 결과를 출력할 수 있는 방법입니다.

| 내장 함수 |

```
void echo ( string $arg1 [, string $... ] )
```

예제 파일 | [echo.php](#)

```
1  <?php
```

```

2     echo "hello world";
3
4     ?>

```

화면 출력

hello world

| 내장 함수 |

```
int print ( string $arg )
```

내장 함수 print() 함수는 echo() 함수와 다르게 문자열을 화면에 출력 후 성공 여부를 논리값으로 반환합니다. 반환값은 화면의 정상적인 출력 여부를 확인할 때 편리합니다.

예제 파일 | print.php

```

1  <?php
2      if (print("안녕하세요!")) {
3          echo ">true";
4      } else {
5          echo ">false";
6      }
7
8  ?>

```

화면 출력

안녕하세요!>true

03.13.2 포맷 출력

포맷 출력은 문자열을 그대로 출력하는 것이 아니라 포매팅 처리를 하여 결과를 출력하는 방법입니다. 포매팅 출력은 C 언어 등에서 많이 이용하는 방법입니다. PHP에서도 C 언어와 같이 다양한 포매팅 처리 함수를 사용할 수 있습니다.

사용법 또한 매우 유사합니다. 사용되는 포맷 코드는 다음과 같습니다.

- `%b`: 바이너리 출력
- `%c`: 아스키문자 출력
- `%d`: 10진수 출력
- `%f`: 실수 출력
- `%o`: 8진수 출력
- `%s`: 문자열 출력
- `%x`: 16진수 소문자 출력
- `%X`: 16진수 대문자 출력

| 내장 함수 |

```
int printf ( string $format [, mixed $args [, mixed $... ] ] )
```

내장 함수 `printf()`는 문자열을 지정한 포맷 방식으로 출력합니다. 포맷은 출력 문자열에 지정한 타입 형태로 데이터를 삽입하여 출력할 수 있는 기능입니다.

예제 파일 | `printf.php`

```
1 <?php
2     $name = "jiny";
3
4     if (printf("안녕하세요! %s 입니다.", $name)) {
5         echo ">true";
6     } else {
7         echo ">false";
8     }
9
10 ?>
```

화면 출력

안녕하세요! jiny 입니다.>true

| 내장 함수 |

```
int fprintf ( resource $handle , string $format [, mixed $args [, mixed $... ] ] )
```

내장 함수 `fprintf()`는 지정한 포맷을 스트림으로 출력합니다.

예제 파일 | `fprintf.php`

```
1  <?php
2      if (!$fp = fopen('date.txt', 'w')) {
3          return;
4      }
5
6      // 지정한 포맷으로 파일 스트림에 출력합니다.
7      fprintf($fp, "%04d-%02d-%02d", $year, $month, $day);
8
9  ?>
```

| 내장 함수 |

```
int vprintf ( string $format , array $args )
```

내장 함수 `vprintf()`는 형식화된 문자열을 출력합니다.

예제 파일 | `vprintf.php`

```
1  <?php
2      vprintf("%04d-%02d-%02d", explode('-', '2017-8-6'));
3
4  ?>
```

화면 출력

2017-08-06

| 내장 함수 |

```
int fprintf ( resource $handle , string $format , array $args )
```

내장 함수 `fprintf()`는 지정한 포맷을 스트림으로 출력합니다.

예제 파일 | `fprintf.php`

```
1 <?php
2     if (!($fp = fopen('date.txt', 'w'))){
3         return;
4     }
5     fprintf($fp, "%04d-%02d-%02d", array($year, $month, $day));
6
7 ?>
```

| 내장 함수 |

```
string sprintf ( string $format [, mixed $args [, mixed $... ]] )
```

내장 함수 `sprintf()`는 `printf()` 함수와 달리 화면에 출력하지 않고 포맷 형태로 변환하여 문자열을 반환합니다.

예제 파일 | `sprintf.php`

```
1 <?php
2     $name = "jiny";
3     $string = sprintf("안녕하세요! %s 입니다.", $name);
4     echo $string;
5
6 ?>
```

화면 출력

안녕하세요! jiny 입니다.

| 내장 함수 |

```
string vsprintf ( string $format , array $args )
```

내장 함수 `vsprintf()`는 포맷 스트링을 반환합니다.

예제 파일 | `vsprintf.php`

```
1  <?php
2      echo vsprintf("%04d-%02d-%02d", explode('-', '2017-8-6'));
3
4  ?>
```

화면 출력

2017-08-06

03.13.3 포맷 입력

내장 함수 `sscanf()`는 형식에 따라 입력된 문자열의 구문 분석합니다.

| 내장 함수 |

```
mixed sscanf ( string $str , string $format [, mixed &$... ] )
```

예제 파일 | `sscanf.php`

```
1  <?php
2      // 시리얼 넘버를 읽어옵니다.
3      list($serial) = sscanf("SN/123456", "SN/%d");
4
5      $mandate = "july 06 2017";
6      list($month, $day, $year) = sscanf($mandate, "%s %d %d");
7      echo "Item $serial was manufactured on: $year-" . substr($month, 0, 3)
8          . "-$day\n";
9
10 ?>
```

Item 123456 was manufactured on: 2017-jul-6

03.14 html 문자열

PHP는 웹 사이트 개발 및 HTML을 처리하는 데 매우 친화적으로 사용할 수 있는 언어입니다. HTML은 다양한 태그와 기호를 포함하고 있습니다. PHP는 HTML 태그들을 처리하고 출력을 위한 다양한 함수를 지원합니다.

03.14.1 백슬래시

문자열을 처리하는 데 있어서 특수기호 백슬래시(\)는 SQL 처리 및 문자열을 처리하는 데 방해될 수 있습니다. 웹과 DB 연동을 위해서 문자열의 백슬래시 기호는 안전하게 처리해야 합니다. 이를 위해서 PHP는 백슬래시 처리에 관련된 함수를 제공합니다.

| 내장 함수 |

```
string addslashes ( string $str )
```

내장 함수 addslashes()는 주어진 문자열에 백슬래시로 감싸 반환합니다. 백슬래시는 ', ", \ 등 특수기호를 데이터베이스 쿼리 등 작업할 때 많이 사용합니다.

예제 파일 | addslashes.php

```
1 <?php
2     echo addslashes("c:\aaa\bbb\ccc");
3     echo "<br>";
4
5     echo addslashes("안녕하세요! 'jiny'님");
6     echo "<br>";
7
8 ?>
```

화면 출력

```
c:\aaa\bbb\ccc
안녕하세요! \'jiny\'님
```

| 내장 함수 |

```
string addslashes ( string $str , string $charlist )
```

내장 함수 addslashes()는 C 스타일로 주어진 문자열에 백슬래시로 감싸 반환합니다.

| 내장 함수 |

```
string stripslashes ( string $str )
```

내장 함수 stripslashes()는 addslashes() 함수의 반대입니다. 주어진 매개변수 문자열에서 quote 부분을 삭제하여 반환합니다.

예제 파일 | stripslashes.php

```
1  <?php
2      $str = "안녕하세요 \'지니\'입니다.";
3      $temp = addslashes($str);
4      echo $temp."<br>";
5
6      $temp2 = stripslashes($temp);
7      echo $temp2."<br>";
8
9  ?>
```

화면 출력

```
안녕하세요 \'지니\'입니다.
안녕하세요 \'지니\'입니다.
```

| 내장 함수 |

```
string stripslashes ( string $str )
```

내장 함수 stripslashes()는 addslashes()의 반대입니다.

03.14.2 라인 브레이크

콘솔 및 텍스트 기반의 문자열 처리에서 다음 줄을 표시하는 기호는 \n을 사용합니다. 하지만 웹 화면에서는 다음 줄(\n)이 반영되지 않습니다.

출력 결과를 웹으로 처리하기 위해서는 \n 기호를 웹에서의 다음 줄인
 기호로 변경해야 합니다.

| 내장 함수 |

```
string nl2br ( string $string [, bool $is_xhtml = true ] )
```

내장 함수 nl2br()은 HTML 라인 브레이크로
 태그를 사용합니다.

예제 파일 | nl2br.php

```
1 <?php
2     echo nl2br("안녕하세요!\n 지니입니다.");
3
4 ?>
```

화면 출력

안녕하세요!
지니입니다.

03.14.3 태그 제거

내장 함수 strip_tags()는 문자열에서 HTML과 PHP 태그를 제거한 문자열을 반환합니다.

| 내장 함수 |

```
string strip_tags ( string $str [, string $allowable_tags ] )
```

예제 파일 | **strip_tags.php**

```
1  <?php
2      $html = "
3      <h1>안녕하세요!</h1>
4      <br>
5      <?php phpinfo(); ?>
6      ";
7
8      echo $html;
9
10     echo "===== <br>";
11
12     $temp = strip_tags($html);
13     echo $temp;
14
15  ?>
```

화면 출력

안녕하세요!

=====
안녕하세요!

03.14.4 html entities

내장 함수 `htmlentities()`는 변경 가능한 모든 글자들을 HTML entities 코드로 변환합니다.

| 내장 함수 |

```
string htmlentities ( string $string [, int $flags = ENT_COMPAT | ENT_HTML401 [,  
string $encoding = ini_get("default_charset") [, bool $double_encode = true ]]] )
```

- ENT_COMPAT: 큰따옴표만 변환합니다.
- ENT_QUOTES: 큰따옴표와 작은따옴표를 모두 변환합니다.
- ENT_NOQUOTES: 큰따옴표와 작은따옴표를 변환하지 않습니다.
- ENT_IGNORE: 빈 문자열을 반환하는 대신에 잘못된 코드 단위 시퀀스는 자동으로 무시합니다. 이 플래그를 사용하면 보안에 영향을 미칠 수 있으므로 사용하지 않는 것이 좋습니다.
- ENT_SUBSTITUTE: 잘못된 코드 단위 시퀀스를 유니 코드 대체 문자 U + FFFD (UTF-8) 또는 & # FFFD; (그렇지 않으면) 빈 문자열을 반환합니다.
- ENT_DISALLOWED: 주어진 문서 유형에 대한 유효하지 않은 코드 포인트를 유니 코드 대체 문자 U + FFFD (UTF-8) 또는 & # FFFD; (그렇지 않은 경우) 그대로 남겨 둡니다.
- ENT_HTML401: 코드를 HTML 4.01로 처리합니다.
- ENT_XML1: 코드를 XML 1로 처리합니다.
- ENT_XHTML: 코드를 XHTML로 처리합니다.
- ENT_HTML5: 코드를 HTML 5로 처리합니다.

예제 파일 | **htmlentities.php**

```
1  <?php
2      $str = "A 'quote' is <b>bold</b>";
3
4      // 출력: A 'quote' is &lt;b&gt;bold&lt;/b&gt;
5      echo htmlentities($str);
6      echo "<br>";
7
8      // 출력: A &#039;quote&#039; is &lt;b&gt;bold&lt;/b&gt;
9      echo htmlentities($str, ENT_QUOTES);
10     echo "<br>";
```



```

11
12     $str = "\x8F!!!";
13
14     // 출력: an empty string
15     echo htmlentities($str, ENT_QUOTES, "UTF-8");
16     echo "<br>";
17
18     // 출력: "!!!"
19     echo htmlentities($str, ENT_QUOTES | ENT_IGNORE, "UTF-8");
20     echo "<br>";
21
22     ?>

```

화면 출력

```

'quote' is &lt;b>bold&lt;/b>
<br>
A &#039;quote&#039; is &lt;b>bold&lt;/b>
<br>
<br>
!!!
<br>

```

| 내장 함수 |

```

string html_entity_decode ( string $string [, int $flags = ENT_COMPAT | ENT_HTML401 [, string $encoding = ini_get("default_charset") ] ] )

```

내장 함수 `html_entity_decode()`는 HTML 엔티티를 문자로 변환합니다. `html_entity_decode()`는 문자열의 모든 HTML 엔티티를 해당 문자로 변환한다는 점에서 `htmlentities()`와 반대 함수입니다.

예제 파일 | [html_entity_decode.php](#)

```

1  <?php
2      $str = "I'll \"walk\" the <b>cat</b> now";
3      echo $str."<br>";
4

```

```

5     $a = htmlentities($str);
6     echo $a."<br>";
7
8     echo html_entity_decode($a);
9
10  ?>

```

화면 출력

```

I'll "walk" the cat now
I'll "walk" the <b>cat</b> now
I'll "walk" the cat now

```

| 내장 함수 |

```

string htmlspecialchars ( string $string [, int $flags = ENT_COMPAT | ENT_HTML401
[, string $encoding = ini_get("default_charset") [, bool $double_encode = true ]]] )

```

내장 함수 `htmlspecialchars()`는 다음과 같은 HTML 특수 문자를 다른 entity 코드로 변환해 줍니다.

- & `&`;
- " `"`;
- ' `'`;
- < `<`;
- > `>`;

코드를 변환하여 출력하면 브라우저에서 코드를 분석하지 않고 html 코드 자체를 바로 출력할 수 있습니다.

예제 파일 | [htmlspecialchars.php](#)

```

1  <?php
2      $body = "<h1 class=\"aaa\" id='bb'>안녕하세요</h1> <br>";
3      echo $body;
4

```

```

5      echo htmlspecialchars($body);
6
7      ?>

```

화면 출력

```

<h1 class="aaa" id='bb'>안녕하세요</h1> <br>
<h1 class="aaa" id='bb'>안녕하세요</h1> <br>

```

| 내장 함수 |

```

string htmlspecialchars_decode ( string $string [, int $flags = ENT_COMPAT | ENT_HTML401 ])

```

내장 함수 htmlspecialchars_decode()는 특수 HTML 엔티티를 다시 문자로 변환합니다. htmlspecialchars() 함수의 반대 함수입니다.

예제 파일 | htmlspecialchars_decode.php

```

1  <?php
2      $str = "<p>this -&gt; &quot;</p>\n";
3
4      echo htmlspecialchars_decode($str);
5
6      echo htmlspecialchars_decode($str, ENT_NOQUOTES);
7
8      ?>

```

화면 출력

```

<p>this -> "</p>
<p>this -> &quot;</p>

```

| 내장 함수 |

```

array get_html_translation_table ([ int $table = HTML_SPECIALCHARS [, int $flags = ENT_COMPAT | ENT_HTML401 [, string $encoding = "UTF-8" ]]] )

```

내장 함수 `get_html_translation_table()`은 `htmlspecialchars()` 및 `htmlentities()`에서 사용하는 변환 테이블을 반환합니다.

예제 파일 | `get_html_translation_table.php`

```
1  <?php
2      var_dump(get_html_translation_table(HTML_ENTITIES, ENT_QUOTES | ENT_
        HTML5));
3  ?>
```

화면 출력

```
array(1511) {
  [" "]=> string(5) "  "
  [" "]=> string(9) "  "
  ["!"]=> string(6) "!"
  ["'"]=> string(6) "'"
  ["#"]=> string(5) "#"
  .... 중간생략
}
```

03.14.5 메타

내장 함수 `get_meta_tags()`는 html 파일의 내용을 읽어 `<head></head>` 안에 설정되어 있는 메타 태그 값을 추출합니다. 추출된 모든 content를 배열 형태로 반환합니다.

| 내장 함수 |

```
array get_meta_tags ( string $filename [, bool $use_include_path = false ] )
```

예제 파일 | `get_meta_tags.php`

```
1  <?php
2      // html 파일
3      $tags = get_meta_tags("sample.htm");
4
```

```

5     echo "META author = ". $tags['author'] . "<br>";
6     echo "META keywords = ". $tags['keywords'] . "<br>";
7     echo "META description = ". $tags['description'] . "<br>";
8
9     ?>

```

화면 출력

```

META author = jiny
META keywords = php documentation
META description = jinyPHP 샘플 html 파일입니다.

```

| 내장 함수 |

```
string quotemeta ( string $str )
```

내장 함수 `quotemeta()`는 메타 문자들에 대해서 백슬래시가 붙어 있는 형태로 변환합니다.

- 메타 문자: `\ + * ? [^] ($)`

예제 파일 | **quotemeta.php**

```

1  <?php
2      $str = "Hello world. (반가워요!)";
3      echo quotemeta($str);
4
5      ?>

```

화면 출력

```
Hello world\.\ (반가워요\)
```

03.14.6 escape

내장 함수 `escapeshellcmd()`는 이스케이프 셸 메타 문자를 처리합니다.

| 내장 함수 |

```
string escapeshellcmd ( string $command )
```

escapeshellcmd()는 외부 입력값을 통하여 셸 명령을 실행할 때 발생할 수 있는 악성 문자열 등을 이스케이프 처리합니다. 악성 문자열 등은 시스템 명령을 생성하는 과정에 악의적인 명령을 통하여 시스템 보안을 취약하게 만들 수 있습니다.

exec() 또는 system() 함수를 실행 전에 모든 입력 데이터에 대해서 이스케이프 처리를 하는 것이 좋습니다.

다음 문자 앞에는 백슬래시를 추가합니다. & # & | * ? ~ < > ^ () [] { } \$ \, \ x0A 및 \ xFF. 윈도우에서는 이러한 모든 문자와 % 및 !가 대신 공백으로 대체됩니다.

예제 파일 | **escapeshellcmd.php**

```
1  <?php
2      $command = './configure '.$_POST['options'];
3
4      $escaped_command = escapeshellcmd($command);
5
6      system($escaped_command);
7
8  ?>
```

| 내장 함수 |

```
string escapeshellarg ( string $arg )
```

내장 함수 escapeshellarg()는 셸 인수로 사용되는 문자열을 이스케이프 처리합니다.

escapeshellarg()는 문자열 주위에 작은따옴표를 추가합니다. 또한 기존 작은따옴표를 인용/이스케이프하여 문자열을 셸 함수에 단일 안전한 인수로 전달하도록 처리합니다.

셸 함수에는 `exec()`, `system()` 및 백틱 연산자가 포함됩니다.

윈도우에서 `escapeshellarg()` 대신 백분율(`%`) 기호, 느낌표(!) 및 큰따옴표를 공백으로 대체하고 문자열 주위에 큰따옴표를 추가합니다.

예제 파일 | **escapeshellarg.php**

```
1  <?php
2      system('ls ' .escapeshellarg($dir));
3
4  ?>
```

03.15 로케일 및 코드

`IntlChar`는 PHP 7.x로 업그레이드되면서 새롭게 추가된 클래스입니다. 새로운 `IntlChar` 클래스는 추가 ICU 기능을 노출합니다. 이는 유니 코드 문자를 조작하는 데 사용할 수 있습니다. `IntlChar` 클래스를 사용하기 위해서는 `Intl` 확장 기능이 설치되어 있어야 합니다.

```
<?php
    printf('%x', IntlChar::CODEPOINT_MAX);
    echo IntlChar::charName('@');
    var_dump(IntlChar::ispunct('!'));
?>
```

| 내장 함수 |

```
string setlocale ( int $category , string $locale [, string $... ] )
```

내장 함수 `setlocale()`은 로케일 정보를 설정합니다.

| 내장 함수 |

array **localeconv** (void)

내장 함수 `localeconv()`는 숫자 형식 정보를 가져옵니다.

예제 파일 | **localeconv.php**

```
1  <?php
2      setlocale(LC_ALL, 'nl_NL.UTF-8@euro');
3
4      $locale_info = localeconv();
5      print_r($locale_info);
6
7  ?>
```

화면 출력

```
Array ( [decimal_point] => . [thousands_sep] => [int_curr_symbol] => [currency_
symbol] => [mon_decimal_point] => [mon_thousands_sep] => [positive_sign]
=> [negative_sign] => [int_frac_digits] => 127 [frac_digits] => 127 [p_cs_
precedes] => 127 [p_sep_by_space] => 127 [n_cs_precedes] => 127 [n_sep_by_
space] => 127 [p_sign_posn] => 127 [n_sign_posn] => 127 [grouping] => Array (
) [mon_grouping] => Array ( ) )
```

| 내장 함수 |

string **nl_langinfo** (int \$item)

내장 함수 `nl_langinfo()`는 쿼리 언어 및 로케일 정보, `nl_langinfo()`는 locale 카테고리
의 개별 요소들을 액세스하는 데 이용합니다. 모든 요소를 반환하는 `localeconv()`와 달
리 `nl_langinfo()`는 특정 요소만을 선택할 수 있습니다.

04

JSON

JSON 포맷은 더글라스 크록포드에 의해서 RFC7159와 ECMA-404라는 표준에 의해서 정의된 기술입니다. ECMA-404는 표준 문법만 제공하며, RFC7159는 보안적인 시맨틱 부분을 고려한 문법을 제공합니다.



JSON이란 Javascript Object Notation의 약자로 XML과 비슷한 데이터의 집합의 텍스트 데이터입니다. 주로 데이터 교환을 위하여 최근에 많이 사용하는 데이터 포맷의 일종입니다.

최근 들어 비동기 통신(AJAX)이 인기를 얻고 있는 가운데 간단한 데이터 값의 서로 전송을 위하여 JSON 규격의 문자열 데이터를 많이 사용합니다. JSON은 특히 인터넷에서 데이터를 주고받는데 더욱더 유용합니다. 대부분의 웹 API들은 JSON 방식의 데이터 처리와 교환을 위한 포맷으로 이용합니다.

JSON 데이터는 C, C++, PHP 등 프로그래머들에게 친숙한 텍스트 규격 형태로 초보 개발자들도 쉽게 이해할 수 있습니다. JSON 파일을 처리하기 위해서는 복잡한 문자열 처리 루틴이 필요합니다. 하지만 최신 버전의 PHP는 최신 트렌드의 JSON을 쉽게 처리할 수 있는 내장 함수를 제공합니다.

04.1 JSON 문법

JSON은 다양한 형식의 데이터를 문자열을 이용하여 데이터 직렬화 처리를 합니다. 여러 개의 데이터를 하나의 문자열로 표현하는 데 있어 약간의 규칙과 문법 구조를 가지고 있습니다.

04.1.1 문자열 표현

JSON에서 키와 데이터를 표현하는 문자열은 큰따옴표(")로 묶어서 작성합니다. JSON 데이터를 PHP 소스 내에서 사용할 때 큰따옴표는 기존 문자열과 함께 처리하는 데 충돌이 발생할 수 있습니다.

PHP에서 JSON 데이터 문자열을 삽입할 때 JSON 안에 있는 큰따옴표는 백슬래시(\)를 추가하여 사용합니다.

04.1.2 데이터 표현

JSON에서 데이터는 키/값 한 쌍의 데이터로 표시합니다. collection 타입 표기라고도 합니다. 이러한 표기는 object, struct(구조체), 연상 배열과 같은 문법 표현에서 자주 사용됩니다.

| 표현 |

```
"키": "값"
```

키 이름은 변수와 같으며, 값은 그 변수의 데이터 값과 같다고 합니다. JSON은 배열처럼 여러 개의 키/값을 갖는 다수의 데이터로 구성할 수 있습니다. 즉 연상 배열처럼 키와 값 형태의 한 쌍으로 다수의 데이터를 처리할 수 있습니다.

04.1.3 배열 표현

JSON 데이터를 배열로 처리할 수 있습니다. 데이터는 키와 값 형태로 묶어서 입력합니다. 그리고 대괄호를 이용하여 감싸면 됩니다.

예제 파일 | **json-01.php**

```
1  <?php
2
3  $string = "
4      [10, {"a":20}, [30,"서른"] ]
5  ";
6
7  echo "=== JSON 문자열 ===<br>";
8  echo $string;
9  echo "<br>";
10
11 echo "=== 배열 처리 ===<br>";
12 $arr = json_decode($string);
13 print_r($arr);
14
15 ?>
```

화면 출력

```
=== JSON 문자열 ===
[10, {"a":20}, [30,"서른"] ]
=== 배열 처리 ===
Array ( [0] => 10 [1] => stdClass Object ( [a] => 20 ) [2] => Array ( [0] => 30 [1]
=> 서른 ) )
```

04.1.4 객체 표현

JSON의 데이터를 객체로 표현할 수 있습니다. 각각의 데이터를 중괄호로 표기하면 됩니다.

| 표현 |

```
{
    "키": "값",
    "키": "값",
    "키": "값",
    "키": "값"
}
```

예제 파일 | json-02.php

```
1  <?php
2
3      $string = "
4          {"name1":10,"name2":"안녕하세요","name3":true}
5      ";
6
7      echo "=== JSON 문자열 ===<br>";
8      echo $string;
9      echo "<br>";
10
11     echo "=== 배열 처리 ===<br>";
12     $arr = json_decode($string);
13     print_r($arr);
14
15  ?>
```

화면 출력

```
=== JSON 문자열 ===
{"name1":10,"name2":"안녕하세요","name3":true}
=== 배열 처리 ===
stdClass Object ( [name1] => 10 [name2] => 안녕하세요 [name3] => 1 )
```

JSON 문자열은 비순서화된 데이터 세트입니다. 중괄호 안에 여러 개의 데이터를 넣을 수 있기 때문입니다. 여러 개의 데이터를 묶을 때는 각각의 데이터를 콤마(,)로 구분하고 전체를 중괄호로 감싸면 됩니다.

04.1.5 객체 이중화 표현

하나의 그룹으로 만들어진 JSON 데이터를 이중으로 다시 묶을 수도 있습니다. JSON 객체 안에 또 다른 JSON 객체를 갖는 형태입니다. 이중 배열처럼 다중 관계를 가지는 구조라고 이해할 수 있습니다.

| 표현 |

```
{
  {
    "키": "값",
    "키": "값",
    "키": "값",
    "키": "값"
  },
  {
    "키": "값",
    "키": "값",
    "키": "값",
    "키": "값"
  }
}
```

이때 또한 각각의 JSON 데이터를 콤마(,)로 구분하고 다시 전체를 중괄호를 이용하여 감싸 묶을 수 있습니다.

04.1.6 객체 키 표현

JSON은 중괄호를 하나의 object, array로 인식합니다. 또한 각각의 데이터는 콤마를 이용하여 구분합니다. 또한 콜론(:) 앞은 키 이름으로 인식합니다.

이중 배열처럼 JSON 데이터의 키 값이 있을 경우에는 “키”:{} 형태로 지정할 수 있습니다.

| 표현 |

```
{
  "1":{
    "FirstName":"lee",
    "LastName":"hojin"
  }
}
```

여러 개의 이중 배열 형태로 묶기 위해서는 대괄호를 사용할 수도 있습니다.

쌍따옴표(")로 둘러싸인 키와 쌍따옴표(")로 둘러싸인 값을 콜론(:) 기호로 연결합니다. 또한 여러 개의 데이터는 콤마(,)로 구분합니다.

04.1.7 주석

JSON 데이터는 띄어쓰기나 탭을 통하여 보기 좋게 코드를 정리할 수는 있으나 주석과 같은 설명문은 넣을 수가 없습니다.

04.2 JSON 인코딩

소스상의 데이터, 배열을 JSON 형태의 문자열로 변환할 수 있습니다. 인코딩은 디코딩의 반대말입니다. PHP는 JSON 인코딩을 쉽게 처리하기 위해서 전용 함수를 제공합니다.

| 내장 함수 |

```
string json_encode ( mixed $value [, int $options = 0 [, int $depth = 512 ]] )
```

내장 함수 `json_encode()`를 통하여 간단하게 배열 데이터를 JSON 형태의 문자열로 직렬화 변환을 할 수 있습니다.

예제 파일 | json-03.php

```
1  <?php
2
3      $arr['FirstName'] = "lee";
4      $arr['LastName'] = "hojin";
5
6      $json = json_encode($arr);
7      echo $json;
8
9  ?>
```

화면 출력

```
{"FirstName":"lee","LastName":"hojin"}
```

위의 실험을 보면 배열 변수를 인코딩 함수를 이용하여 JSON 문자열로 변경합니다. json으로 직렬화된 문자열을 출력합니다.

예제 파일 | json-04.php

```
1  <?php
2
3      $arr['FirstName'] = "lee";
4      $arr['LastName'] = "hojin";
5      $user['1'] = $arr;
6
7      $arr['FirstName'] = "jiny";
8      $arr['LastName'] = "PHP";
9      $user['2'] = $arr;
10
11     $json = json_encode($user);
12     echo $json;
13
14  ?>
```

화면 출력

```
{
  "1":{
    "FirstName":"lee",
    "LastName":"hojin"
```

```

},
"2":{
  "FirstName":"jiny",
  "LastName":"PHP"
}
}

```

위의 실험은 다차원 배열의 데이터를 JSON으로 인코딩하는 예제입니다. 배열에서 연상 키 값이 있으면 “키”:{} 형태로 변경되는 것을 확인할 수 있습니다. 또한 다차원 배열을 JSON 직렬화하면서 중괄호를 계속 중첩하여 사용합니다. 다차원적인 배열도 JSON 데이터 직렬화를 처리할 수 있습니다.

04.3 JSON 디코딩

직렬화된 JSON은 데이터를 담고 있는 텍스트 문자열과 같습니다. 데이터를 직렬화하여 처리하는 것은 이기종 간에 데이터를 전달할 때 매우 편리합니다. 직렬화 변환된 데이터를 저장하고 다른 시스템으로 전송도 가능합니다.

04.3.1 배열 디코딩

내장 함수 `json_decode()`를 통하여 간단하게 직렬화된 JSON 문자열을 데이터로 변환할 수 있습니다. 디코딩은 인코딩의 반대말입니다.

PHP는 JSON 파일을 PHP에서 사용할 수 있는 데이터 형태로 쉽게 변환 수 있도록 전용 함수를 제공합니다.

| 내장 함수 |

```

mixed json_decode ( string $json [, bool $assoc = false [, int $depth = 512 [, int $options = 0 ]]] )

```


예제 파일 | json-05.php

```
1  <?php
2
3      // 큰따옴표를 사용하기 위해서 백슬래시를 추가
4      $string = "
5      {
6          \"Id\": \"01\",
7          \"FirstName\": \"lee\",
8          \"LastName\": \"hojin\",
9          \"Country\": \"Korea\"
10     }
11     ";
12
13     echo "=== JSON 문자열 ===<br>";
14     echo $string;
15     echo "<br>";
16
17     echo "=== 배열 처리 ===<br>";
18     $arr = json_decode($string);
19
20     while (list($key,$val) = each($arr)) {
21         echo $key. "=="> $val. "<br>";
22     }
23
24     ?>
```

화면 출력

=== JSON 문자열 ===

```
{ "Id": "01", "FirstName": "lee", "LastName": "hojin", "Country": "Korea" }
```

=== 배열 처리 ===

Id==>01

FirstName==>lee

LastName==>hojin

Country==>Korea

04.3.2 다중 배열 디코딩

JSON은 1차원 데이터 이외에 다단계의 값을 갖는 다차원 배열 형태로 값을 지정할 수 있습니다.

다차원 배열을 지정하기 위해서는 중괄호 외에 대괄호를 사용합니다.

| 표현 |

```
[
  {
    "키": "값",
    "키": "값",
    "키": "값",
    "키": "값"
  },
  {
    "키": "값",
    "키": "값",
    "키": "값",
    "키": "값"
  }
]
```

대괄호는 배열 안에 또 다른 json 배열이 있다는 표기입니다.

예제 파일 | [json-06.php](#)

```
1 <?php
2
3 // 큰따옴표를 사용하기 위해서 백슬래시를 추가
4 $string = "
5
6 [
7     {
8         \"Id\": \"01\",
9         \"FirstName\": \"lee\",
```

```

10         \"LastName\": \"hojin\",
11         \"Country\": \"Korea\"
12     },
13     {
14         \"Id\": \"02\",
15         \"FirstName\": \"jiny\",
16         \"LastName\": \"PHP\",
17         \"Country\": \"Korea\"
18     }
19 ]
20
21 ";
22
23 echo "=== JSON 문자열 ===<br>";
24 echo $string;
25 echo "<br>";
26
27 echo "=== 배열 처리 ===<br>";
28 $arr = json_decode($string);
29
30 for ($i=0;$i<count($arr);$i++) {
31     echo "첫 번째 배열 $i <br>";
32
33     while (list($key,$val) = each($arr[$i])) {
34         echo $key. "==>" . $val. "<br>";
35     }
36 }
37
38 ?>

```

화면 출력

=== JSON 문자열 ===

```
[
{ "Id": "01", "FirstName": "lee", "LastName": "hojin", "Country": "Korea" },
{ "Id": "02", "FirstName": "jiny", "LastName": "PHP", "Country": "Korea" }
]
```

=== 배열 처리 ===

첫 번째 배열 0

Id==>01

FirstName==>lee

```
LastName==>hojin  
Country==>Korea  
첫 번째 배열 1  
Id==>02  
FirstName==>jiny  
LastName==>PHP  
Country==>Korea
```

JSON 데이터를 소스상에서 직접 생성하여 처리할 수도 있지만, 외부 API 서비스를 접속하여 데이터를 수신한 후에 디코딩하여 프로그램에서 적용할 수 있습니다.

04.4 객체 직렬화

JSON 방식 이외에도 PHP 내에서 제공되는 함수를 통하여 객체를 직렬화 처리할 수 있습니다. 변환된 문자열은 파일 등으로 출력하여 저장할 수도 있고, 저장된 파일을 읽어서 객체로 다시 복원할 수도 있습니다.

| 내장 함수 |

```
string serialize ( mixed $value )
```

내장 함수 `serialize()`는 저장 가능한 값에 대해서 바이트 스트림 표현의 문자열로 변경합니다. 객체 등을 변환하여 저장하거나 전달할 때 매우 유용합니다.

예제 파일 | [serialize.php](#)

```
1  <?php  
2  
3      class A {  
4          public $one = 1;  
5  
6          public function show_one() {  
7              echo $this->one;  
8          }  
}
```

```

9     }
10
11     $a = new A;
12
13     // 클래스 인스턴스 객체를 직렬화하여 저장합니다.
14     $serialObj = serialize($a);
15     file_put_contents('store', $serialObj);
16
17     echo $serialObj;
18
19     ?>

```

화면 출력

```
0:1:"A":1:{s:3:"one";i:1;}
```

| 내장 함수 |

```
mixed unserialize ( string $str [, array $options ] )
```

내장 함수 `unserialize()`는 `serialize()` 함수를 통하여 직렬화된 기능을 역직렬화 작업을 수행합니다. 역직렬화를 하기 위해서는 직렬화된 문자열만으로는 처리할 수 없습니다. 역직렬화를 위한 이전의 클래스 객체의 정의가 함께 필요합니다.

예제 파일 | `unserialize.php`

```

1  <?php
2
3      class A {
4          public $one = 1;
5
6          public function show_one() {
7              echo $this->one;
8          }
9      }
10
11      // 저장된 직렬화 문자열 파일을 읽어서, 역직렬화를 수행합니다.
12      $s = file_get_contents('store');

```

```
13  $a = unserialize($s);  
14  
15  // $a 객체의 show_one() 함수를 호출합니다.  
16  $a->show_one();  
17  
18  ?>
```

화면 출력

1

05

날짜

서비스를 위한 응용프로그램을 개발하는 데 있어서 날짜와 시간 정보는 매우 중요합니다. 대부분의 데이터는 시간에 종속적인 경우가 많습니다. 날짜와 시간 정보는 데이터를 처리하는 데 있어서 떼려야 뗄 수 없는 밀접한 관계를 가지고 있습니다.

이번 장에서는 PHP에서 날짜와 시간에 관련된 함수들과 클래스를 살펴보도록 하겠습니다.

05.1 실시간 환경 설정

PHP가 설치되어 있는 지역과 환경에 따라서 기준이 다를 것입니다. 서버의 설정 환경에 맞게 날짜와 시간을 계산할 필요가 있습니다. 특히 글로벌 서비스를 준비하고 있는 경우 날짜와 시간 관련 처리에 많은 고민이 필요합니다.

PHP는 날짜와 시간을 처리할 수 있는 다양한 내장 함수를 지원합니다. 하지만 날짜와 시간 관련 함수들은 php.ini 설정값의 영향을 받습니다.

05.2 날짜

동근 지구의 각 나라별로 날짜를 지정하고 처리하는 방법은 다양합니다. 이와 관련하여 국제 기준 시간과 날짜를 기준으로 계산합니다.

| 내장 함수 | **기준 날짜**

```
string gmdate ( string $format [, int $timestamp = time() ] )
```

내장 함수 gmdate()는 CMT/CUT 기준의 날짜와 시간을 반환합니다. 기존 date() 함수와의 차이점은 기준 날짜를 GMT로 사용한다는 것입니다.

예제 파일 | **gmdate.php**

```
1  <?php
2
3      echo date("Y-m-d H:i:s");
4      echo "<br>";
5
6      echo "GMT(greenwich Mean Time) 기준 = ".gmdate("Y-m-d H:i:s");
7
8  ?>
```

화면 출력

2017-06-18 06:31:56

GMT(greenwich Mean Time) 기준 = 2017-06-18 06:31:56

| 내장 함수 |

```
array date_parse ( string $date )
```

지정한 날짜에 대해서 상세 정보를 배열로 반환합니다.

예제 파일 | **date_parse.php**

```
1  <?php
```



```

2     print_r(date_parse("2017-08-07 10:00:00.5"));
3
4     ?>

```

화면 출력

```

Array ( [year] => 2017 [month] => 8 [day] => 7 [hour] => 10 [minute] => 0 [second]
=> 0 [fraction] => 0.5 [warning_count] => 0 [warnings] => Array ( ) [error_
count] => 0 [errors] => Array ( ) [is_localtime] => )

```

05.3 시간

날짜와 마찬가지로 시간 또한 매우 중요합니다. 일반적인 시스템의 시/분/초와 마이크로 초도 읽어올 수 있습니다. 또한 타임 스탬프를 지원합니다.

05.3.1 시/분/초

시스템을 통하여 현재의 시간을 알아내는 것은 시간 처리 작업을 위한 첫 단계입니다. 내장 함수 `time()`은 현재의 시간을 출력합니다.

| 내장 함수 | 현재 시간

```
int time ( void )
```

현재의 시간은 Unix epoch 이후의 시간을 timestamp 형식의 숫자값으로 표기가 되는 데 이를 가독성 있게 표기하기 위해서는 `date()` 포맷 함수를 이용하여 출력합니다.

예제 파일 | time.php

```

1  <?php
2      echo "현재의 시간은 = " . time() . "입니다. <br>";
3      // 두 번째 인자로 주어진 시간을 포맷에 맞게 출력합니다.
4      echo date("H:i:s",time());
5  ?>

```

화면 출력

현재의 시간은 = 1497766932입니다.
06:22:12

| 내장 함수 | 날짜와 시간

```
array getdate ([ int $timestamp = time() ] )
```

내장 함수 `getdate()`는 날짜와 시간 정보를 읽어옵니다. 날짜와 시간의 반환값은 배열로 받아옵니다. 배열의 키를 통하여 각각의 값을 가지고 올 수 있습니다.

예제 파일 | `getdate.php`

```
1  <?php
2      $date = getdate();
3
4      echo "초 = ". $date['seconds'] . "<br>";
5      echo "분 = ". $date['minutes'] . "<br>";
6      echo "시 = ". $date['hours'] . "<br>";
7      echo "월날짜 = ". $date['mday'] . "<br>";
8      echo "요일(숫자) = ". $date['wday'] . "<br>";
9      echo "월(숫자) = ". $date['mon'] . "<br>";
10     echo "연도 = ". $date['year'] . "<br>";
11     echo "일년날짜수 = ". $date['yday'] . "<br>";
12     echo "요일 = ". $date['weekday'] . "<br>";
13     echo "달 = ". $date['month'] . "<br>";
14
15  ?>
```

화면 출력

초 = 38
분 = 12
시 = 7
월날짜 = 18
요일(숫자) = 0
월(숫자) = 6
연도 = 2017
일년날짜수 = 168

요일 = Sunday

달 = June

| 내장 함수 | 현재 시간 배열

```
mixed gettimeofday ([ bool $return_float = false ] )
```

내장 함수 gettimeofday()는 현재의 시간을 배열 형태로 반환합니다. 배열의 키를 통하여 각각의 값에 접근할 수 있습니다.

예제 파일 | gettimeofday.php

```
1  <?php
2      $time = gettimeofday();
3
4      echo "현재 초 = " . $time['sec'] . "<br>";
5      echo "마이크로 초 = " . $time['usec'] . "<br>";
6      echo "서부 Greenwich 표준 = " . $time['minuteswest'] . "<br>";
7      echo "서머타임 보정 = " . $time['dsttime'] . "<br>";
8
9  ?>
```

화면 출력

현재 초 = 1497770514

마이크로 초 = 65156

서부 Greenwich 표준 = 0

서머타임 보정 = 0

| 내장 함수 |

```
string gmstrftime ( string $format [, int $timestamp = time() ] )
```

내장 함수 gmstrftime()은 로컬 설정에 따른 GMT/UTC 날짜, 시간 형식을 지정합니다.

예제 파일 | gmstrftime.php

```
1 <?php
2     setlocale(LC_TIME, 'en_US');
3     echo strftime("%b %d %Y %H:%M:%S", mktime(20, 0, 0, 12, 31, 98)) . "\n";
4     echo gmstrftime("%b %d %Y %H:%M:%S", mktime(20, 0, 0, 12, 31, 98)) . "\n";
5
6 ?>
```

화면 출력

Dec 31 1998 20:00:00 Dec 31 1998 20:00:00

| 내장 함수 |

int **idate** (string \$format [, int \$timestamp = time()])

내장 함수 idate()는 정수로 현지 시간/날짜 형식을 지정합니다.

예제 파일 | idate.php

```
1 <?php
2     $timestamp = strtotime('1st January 2017');
3
4     // this prints the year in a two digit format
5     // however, as this would start with a "0", it
6     // only prints "4"
7     echo idate('y', $timestamp);
8
9 ?>
```

화면 출력

17

| 내장 함수 |

array **localtime** ([int \$timestamp = time() [, bool \$is_associative = false]])

내장 함수 localtime()은 현지 시간을 반환합니다.

예제 파일 | localtime.php

```
1 <?php
2     $localtime = localtime();
3     $localtime_assoc = localtime(time(), true);
4     print_r($localtime);
5     print_r($localtime_assoc);
6
7 ?>
```

화면 출력

```
Array ( [0] => 24 [1] => 6 [2] => 9 [3] => 7 [4] => 7 [5] => 117 [6] => 1 [7]
=> 218 [8] => 0 ) Array ( [tm_sec] => 24 [tm_min] => 6 [tm_hour] => 9 [tm_
mday] => 7 [tm_mon] => 7 [tm_year] => 117 [tm_wday] => 1 [tm_yday] => 218 [tm_
isdst] => 0 )
```

05.3.2 마이크로 초

시/분/초 외에도 정밀한 마이크로 초 단위의 시간을 읽어올 수 있습니다.

마이크로 초는 프로그램의 실행 속도 등을 측정할 때 자주 사용하는 시간 함수입니다. 매우 정밀한 시간을 표시하기 때문에 프로그램 시작 전에 한 번 실행하고, 프로그램 마지막에 한 번 더 측정하여 프로그램의 동작 시간을 측정할 수 있습니다.

| 내장 함수 | 마이크로 초

```
mixed microtime ([ bool $get_as_float = false ] )
```

내장 함수 microtime()은 현재의 1/1000초 단위로 출력합니다.

예제 파일 | microtime.php

```
1 <?php
2     $mtime = microtime();
3     echo $mtime;
```

```
4
5  ?>
```

화면 출력

0.14970800 1497771297

05.3.3 타임 스탬프

시간을 타임 스탬프 형태로 사용할 수 있습니다. 타임 스탬프는 특정한 시각을 표현하는 문자열입니다. 2개 이상의 시간을 비교하거나 계산하기 위해서 자주 사용합니다.

| 내장 함수 | 타임 스탬프 생성

```
int gmmktime ([ int $hour = gmdate("H") [, int $minute = gmdate("i") [, int $second = gmdate("s") [, int $month = gmdate("n") [, int $day = gmdate("j") [, int $year = gmdate("Y")]]]]]] )
```

내장 함수 `gmmktime()`은 GMT 기준으로 타임 스탬프를 생성합니다.

예제 파일 | `gmmktime.php`

```
1  <?php
2      // Prints: jun 18, 2017
3      echo "jun 18, 2017 is on a " . date("l", gmmktime(0, 0, 0, 6, 18, 2017));
4  ?>
```

화면 출력

jun 18, 2017 is on a Sunday

| 내장 함수 | 지정 날짜

```
int mktime ([ int $hour = date("H") [, int $minute = date("i") [, int $second = date("s") [, int $month = date("n") [, int $day = date("j") [, int $year = date("Y") ]]]]] ] )
```

내장 함수 mktime()은 지정된 날짜의 타임 스탬프를 생성합니다.

예제 파일 | **mktime.php**

```
1  <?php
2      $hour = date("H");
3      $minute = date("i");
4      $second = date("s");
5      $month = date("n");
6      $day = date("d");
7      $year = date("Y");
8
9      $timeStamp = mktime($hour, $minute, $second, $month, $day, $year);
10
11     // 두 번째 인자로 주어진 시간을 포맷에 맞게 출력합니다.
12     echo date("Y-m-d H:i:s", $timeStamp);
13
14  ?>
```

화면 출력

2017-06-18 07:32:17

| 내장 함수 |

```
int strtotime ( string $time [, int $now = time() ] )
```

영어 텍스트 datetime을 Unix 타임 스탬프로 구문 분석합니다.

예제 파일 | **strtotime.php**

```
1  <?php
2      echo strtotime("now"), "<br>";
3      echo strtotime("10 September 2000"), "<br>";
4      echo strtotime("+1 day"), "<br>";
5      echo strtotime("+1 week"), "<br>";
6      echo strtotime("+1 week 2 days 4 hours 2 seconds"), "<br>";
7      echo strtotime("next Thursday"), "<br>";
8      echo strtotime("last Monday"), "<br>";
9
```

```
10 ?>
```

화면 출력

```
1502097067
968544000
1502183467
1502701867
1502889069
1502323200
1501459200
```

05.3.4 일출/일몰

내장 함수 `date_sun_info()`는 일몰(sunset)/일출(sunrise) 및 황혼(twilight) 시작/끝에 대한 정보를 읽어옵니다.

| 내장 함수 |

```
array date_sun_info ( int $time , float $latitude , float $longitude )
```

예제 파일 | `date_sun_info.php`

```
1 <?php
2     $sun_info = date_sun_info(strtotime("2017-08-07"), 31.7667, 35.2333);
3     foreach ($sun_info as $key => $val) {
4         echo "$key: " . date("H:i:s", $val) . "\n";
5     }
6
7 ?>
```

화면 출력

```
sunrise: 02:59:11 sunset: 16:30:21 transit: 09:44:46 civil_twilight_begin:
02:33:08 civil_twilight_end: 16:56:24 nautical_twilight_begin: 02:01:51
nautical_twilight_end: 17:27:41 astronomical_twilight_begin: 01:29:07
astronomical_twilight_end: 18:00:25
```


| 내장 함수 |

```
mixed date_sunrise ( int $timestamp [, int $format = SUNFUNCS_RET_STRING [, float $latitude = ini_get("date.default_latitude") [, float $longitude = ini_get("date.default_longitude") [, float $zenith = ini_get("date.sunrise_zenith") [, float $gmt_offset = 0 ]]]]] )
```

내장 함수 `date_sunrise()`는 위치와 날짜에 대한 일출 시간을 계산합니다.

예제 파일 | `date_sunrise.php`

```
1  <?php
2
3      /* calculate the sunrise time for Lisbon, Portugal
4      Latitude: 38.4 North
5      Longitude: 9 West
6      Zenith ~= 90
7      offset: +1 GMT
8      */
9
10     echo date("D M d Y"). ', sunrise time : ' .date_sunrise(time(),
11         SUNFUNCS_RET_STRING, 38.4, -9, 90, 1);
12  ?>
```

| 내장 함수 |

```
mixed date_sunset ( int $timestamp [, int $format = SUNFUNCS_RET_STRING [, float $latitude = ini_get("date.default_latitude") [, float $longitude = ini_get("date.default_longitude") [, float $zenith = ini_get("date.sunset_zenith") [, float $gmt_offset = 0 ]]]]] )
```

내장 함수 `date_sunset()`은 위치와 날짜에 대한 일몰 시간을 계산합니다.

예제 파일 | `date_sunset.php`

```
1  <?php
```

```

2
3      /* calculate the sunset time for Lisbon, Portugal
4      Latitude: 38.4 North
5      Longitude: 9 West
6      Zenith ~= 90
7      offset: +1 GMT
8      */
9
10     echo date("D M d Y"). ', sunset time : ';
11     echo date_sunset(time(), SUNFUNCS_RET_STRING, 38.4, -9, 90, 1);
12
13  ?>

```

05.4 출력 포맷

날짜와 시간을 표기하는 방법은 여러 문화권에 따라 다양합니다. PHP는 포맷 표기를 통하여 다양한 형태로 날짜와 시간을 표기할 수 있습니다.

05.4.1 시간/날짜

내장 함수 `date()`는 지정한 포맷에 맞추어서 시간과 날짜를 표시합니다.

| 내장 함수 |

```
string date ( string $format [, int $timestamp = time() ] )
```

만일 포맷만 입력한 경우에는 현재의 시간을 표시합니다. 두 번째 인자로 시간을 표시하면 해당 시간을 포맷 형태로 출력합니다.

시간과 날짜를 출력하는 다양한 포맷 기호가 있습니다. 시간 포맷을 사용할 때는 대소문자를 구분하여 사용합니다. 각각의 의미와 표시가 다릅니다.

- a: 소문자 am/pm을 표시합니다.
- A: 대문자 AM/PM을 표시합니다.
- d: 날짜를 두 자리 숫자로 표시합니다. 예) 01, 21, 31
- D: 요일을 영문 약자 세 자리로 표시합니다. 예) fri, sun, mon
- F: 영문으로 월을 표시합니다. 예) January, May, Jun
- m: 월을 숫자 두 자리로 표시합니다. 예) 01, 03, 12
- M: 월을 영문 약자 세 자리로 표시합니다. 예) jan, oct, dec
- h: 시간을 12시간 형태로 표시합니다.
- H: 시간을 24시간 형태로 표시합니다.
- i: 분을 표시합니다.
- j: 날짜를 표시합니다. 10 이하의 날짜일 경우에는 앞에 0을 표시하지 않습니다.
- l: 영문으로 요일을 나타냅니다. 예) friday, sunday
- s: 초를 두 자리로 표시합니다.
- S: 영어의 서수를 추가하여 표시합니다. 예) "th", "nd"
- t: 해당 월의 말일 날짜 수를 표시합니다. 예) 28, 30, 31
- U: 특정 기간 이후의 초를 표시합니다.
- Y: 연도를 네 자리로 표시합니다.
- w: 요일을 숫자로 표시합니다. 예) 일요일=0, 월요일=1, 화요일=2
- y: 연도를 두 자리로 표시합니다.
- z: 1년의 날짜 수를 표시합니다.

예제 파일 | **date.php**

```

1  <?php
2      // 현재의 날짜와 시간을 표시합니다.
3      echo date("Y-m-d H:i:s");
4      echo "<br>";
5
6      // 두 번째 인자로 주어진 시간을 포맷에 맞게 출력합니다.
7      echo date("Y-m-d H:i:s",mktime(0,0,0,1,1,2017));
8
9  ?>

```

화면 출력

```
2017-06-18 06:14:08
2017-01-01 00:00:00
```

| 내장 함수 |

```
array date_parse_from_format ( string $format , string $date )
```

지정된 형식에 따른 날짜 정보를 읽어옵니다.

예제 파일 | `date_parse_from_format.php`

```
1 <?php
2     $date = "6.1.2017 13:00+01:00";
3     print_r(date_parse_from_format("j.n.Y H:iP", $date));
4
5 ?>
```

화면 출력

```
Array ( [year] => 2017 [month] => 1 [day] => 6 [hour] => 13 [minute] => 0 [second]
=> 0 [fraction] => [warning_count] => 0 [warnings] => Array ( ) [error_count]
=> 0 [errors] => Array ( ) [is_localtime] => 1 [zone_type] => 1 [zone] => -60
[is_dst] => )
```

05.4.2 포맷

내장 함수 `strftime()`은 현재의 시간과 날짜를 지정한 타입으로 변환하여 출력합니다.

| 내장 함수 |

```
string strftime ( string $format [, int $timestamp = time() ] )
```

포맷 기호

- %a: 일요일부터 토요일까지의 약식 텍스트 표현
- %A: 일요일부터 토요일까지의 전체 텍스트 표현
- %d: 두 자리 날짜(맨 앞에 0이 붙음) 01 - 31
- %e: 월의 한 자리이며 한 자리 앞의 공백이 있습니다.
- %j: 일, 001에서 366 사이의 0을 시작하는 세 자리 숫자
- %u: ISO-8601 요일의 숫자 표현 1(월요일)~7(일요일)
- %w: 요일의 숫자 표현 0(일요일 기준)~6(토요일 기준)

주간 표시

- %U: 첫 번째 일요일을 첫 번째 주로 시작하는 해당 연도의 주 번호 13(해당 연도의 13번째 주 전체)
- %V: ISO-8601: 주어진 주를 1988년 주 숫자로, 첫 주부터 시작하여 주 4일 이상, 월요일은 주 01~53주(여기서 겹치는 주는 53일입니다)
- %W: 첫 번째 월요일을 첫 번째 주로 시작하는 숫자의 주 표시입니다. 46(월요일로 시작하는 연도의 46번째 주)

날짜 표시

- %b: 1월에서 12월까지 로케일 기반으로 한 약식 월 이름
- %B: 1에서 12월까지의 로케일 기반으로 한 전체의 월 이름
- %h: 로케일(%b의 별칭)을 기준으로 한 약식 월 이름 1월에서 12월까지
- %m: 이 달의 두 자리 표현 01(1월분)~12(12월분)

연도 표시

- %C: 세기의 두 자리 표현
- %g: ISO-8601: 1988 표준(%V 참조)에 의한 연도의 두 자리 표시
- %G: 전체 네 자리 버전의 %g
- %y: 연도의 두 자릿수 표현
- %Y: 연도에 대한 네 자리 표시

시간

- %H: 24시간제 형식의 시를 00 자리에서 23 자리로 두 자리로 표시합니다.
- %k: 24시간 형식의 시간(한 자리 앞의 공백은 0에서 23까지)
- %I: 12 자리 형식의 두 자리 표시 01 - 12
- %l (소문자 'l') 시간은 12시간 형식이며 한 자리 숫자는 1에서 12 사이입니다.
- %M: 분 표시 두 자릿수 00 - 59
- %p: 대문자 'AM' 또는 'PM'(예: 00:31 AM, 22:23 PM)
- %P: 소문자 'am' 또는 'pm'(예: 00:31, 22:23)
- %r "%I: %M : %S %p"와 동일 (예: 09:34:17 오후 21:34:17)
- %R "%H: %M"과 같습니다.(예: 오전 12시 35 분 00:35, 오후 4:44 16:44)
- %S: 두 번째 00에서 59 사이의 두 자리 표시
- %T "%H: %M : %S"와 동일 (예 : 오후 9:34:17의 경우 21:34:17)
- %X: 날짜가 없는 로케일 기반의 선호 시간 표현 (예: 03:59:16 또는 15:59:16)
- %z: 시간대 오프셋입니다.
- %Z: 시간대 약어입니다.

시간 및 일자

- %c: 로케일에 따른 선호 날짜 및 시간 스탬프(예: 2 월 5 일 화요일 00:45:10 2009 년 2 월 5 일 12:45:10 AM)
- %D: "%m / %d / %y"와 동일(예: 2009 년 2 월 5 일 02/05/09)
- %F: "%Y-%m-%d"와 동일합니다.(예: 2009-02-05, 2009 년 2 월 5 일)
- %s: Unix Epoch 시간 타임 스탬프(예: 1979 년 9 월 10 일 305815200 08:40:00 AM)
- %x: 로케일에 기초한 선호 날짜 표현, 시간 없음(예: 02/05/09 for February 5, 2009)

그외 포맷 처리

- %n: 개행 문자 (" \ n")
- %t: 탭 문자 (" \ t")
- %%: 리터럴 퍼센트 문자 ("%")

```
1  <?php
2
3      echo "축약된 요일 이름 = ". strftime("%a") . "<br>";
4      echo "요일 이름 = ". strftime("%A") . "<br>";
5      echo "날짜(10진수) = ". strftime("%d") . "<br>";
6
7      echo "Day of the month, with a space preceding single digits. Not
      implemented as described on Windows. See below for more information.
      = ". strftime("%e") . "<br>";
8
9      echo "Day of the year, 3 digits with leading zeros = ". strftime("%j")
      . "<br>";
10     echo "ISO-8601 numeric representation of the day of the week = ".
      strftime("%u") . "<br>";
11     echo "Numeric representation of the day of the week = ".
      strftime("%w") . "<br>";
12
13     echo "Week number of the given year, starting with the first Sunday as
      the first week = ". strftime("%U") . "<br>";
14     echo "ISO-8601:1988 week number of the given year, starting with the
      first week of the year with at least 4 weekdays, with Monday being the
      start of the week 01 through 53 = ". strftime("%V") . "<br>";
15
16     echo "A numeric representation of the week of the year, starting
      with the first Monday as the first week 46 (for the 46th week of the
      year beginning with a Monday) = ". strftime("%W") . "<br>";
17
18     echo "Abbreviated month name, based on the locale = ". strftime("%b")
      . "<br>";
19     echo "Full month name, based on the locale = ". strftime("%B")
      . "<br>";
20     echo "Abbreviated month name, based on the locale (an alias of %b) = ".
      strftime("%h") . "<br>";
21     echo "Two digit representation of the month = ". strftime("%m") . "<br>";
22     echo "Two digit representation of the century (year divided by 100,
      truncated to an integer) = ". strftime("%C") . "<br>";
23     echo "Two digit representation of the year going by ISO-8601:1988
      standards (see %V) = ". strftime("%g") . "<br>";
24     echo "The full four-digit version of %g = ". strftime("%G") . "<br>";
```

```

25
26 echo "Two digit representation of the year = ". strftime("%y") . "<br>";
27 echo "Four digit representation for the year = ". strftime("%Y") . "<br>";
28 echo "Two digit representation of the hour in 24-hour format = ".
    strftime("%H") . "<br>";
29 echo "Hour in 24-hour format, with a space preceding single digits = ".
    strftime("%k") . "<br>";
30 echo "Two digit representation of the hour in 12-hour format = ".
    strftime("%I") . "<br>";
31 echo "Hour in 12-hour format, with a space preceding single digits = ".
    strftime("%l") . "<br>";
32 echo "Two digit representation of the minute = ". strftime("%M")
    . "<br>";
33 echo "UPPER-CASE 'AM' or 'PM' based on the given time = ".
    strftime("%P") . "<br>";
34 echo "lower-case 'am' or 'pm' based on the given time = ".
    strftime("%p") . "<br>";
35
36 echo "Same as %I:%M:%S %p = ". strftime("%r") . "<br>";
37 echo "Same as %H:%M = ". strftime("%R") . "<br>";
38 echo "Two digit representation of the second = ". strftime("%S")
    . "<br>";
39 echo "Same as %H:%M:%S = ". strftime("%T") . "<br>";
40 echo "Preferred time representation based on locale, without the date
    = ". strftime("%X") . "<br>";
41 echo "The time zone offset. Not implemented as described on Windows.
    See below for more information. = ". strftime("%z") . "<br>";
42 echo "The time zone abbreviation. Not implemented as described on
    Windows. See below for more information. = ". strftime("%Z") . "<br>";
43
44 echo "Preferred date and time stamp based on locale = ".
    strftime("%c") . "<br>";
45 echo "Same as %m/%d/%y = ". strftime("%D") . "<br>";
46 echo "Same as %Y-%m-%d (commonly used in database timestamps) = ".
    strftime("%F") . "<br>";
47 echo "Unix Epoch Time timestamp (same as the time() function) = ".
    strftime("%s") . "<br>";
48
49 echo "Preferred date representation based on locale, without the time
    = ". strftime("%x") . "<br>";
50 echo "A newline character = ". strftime("%n") . "<br>";

```



```

51     echo "A Tab character = ". strftime("%t") . "<br>";
52     echo "A literal percentage character = ". strftime("%%") . "<br>";
53
54     ?>

```

화면 출력

축약된 요일 이름 = Sun

요일 이름 = Sunday

날짜(10진수) = 18

Day of the month, with a space preceding single digits. Not implemented as described on Windows. See below for more information. = 18

Day of the year, 3 digits with leading zeros = 169

ISO-8601 numeric representation of the day of the week = 7

Numeric representation of the day of the week = 0

Week number of the given year, starting with the first Sunday as the first week = 25

ISO-8601:1988 week number of the given year, starting with the first week of the year with at least 4 weekdays, with Monday being the start of the week 01 through 53 = 24

A numeric representation of the week of the year, starting with the first Monday as the first week 46 (for the 46th week of the year beginning with a Monday) = 24

Abbreviated month name, based on the locale = Jun

Full month name, based on the locale = June

Abbreviated month name, based on the locale (an alias of %b) = Jun

Two digit representation of the month = 06

Two digit representation of the century (year divided by 100, truncated to an integer) = 20

Two digit representation of the year going by ISO-8601:1988 standards (see %V) = 17

The full four-digit version of %g = 2017

Two digit representation of the year = 17

Four digit representation for the year = 2017

Two digit representation of the hour in 24-hour format = 07

Hour in 24-hour format, with a space preceding single digits =

Two digit representation of the hour in 12-hour format = 07

Hour in 12-hour format, with a space preceding single digits =

Two digit representation of the minute = 02

UPPER-CASE 'AM' or 'PM' based on the given time =

lower-case 'am' or 'pm' based on the given time = AM

Same as %I:%M:%S %p = 07:02:32 AM
 Same as %H:%M = 07:02
 Two digit representation of the second = 32
 Same as %H:%M:%S = 07:02:32
 Preferred time representation based on locale, without the date = 07:02:32
 The time zone offset. Not implemented as described on Windows. See below for more information. = +0900
 The time zone abbreviation. Not implemented as described on Windows. See below for more information. = ♦♦oya♦ ♦꺆♦
 Preferred date and time stamp based on locale = Sun Jun 18 07:02:32 2017
 Same as %m/%d/%y = 06/18/17
 Same as %Y-%m-%d (commonly used in database timestamps) = 2017-06-18
 Unix Epoch Time timestamp (same as the time() function) =
 Preferred date representation based on locale, without the time = 06/18/17
 A newline character =
 A Tab character =
 A literal percentage character = %

| 내장 함수 |

array **strtotime** (string \$date , string \$format)

내장 함수 strtotime()으로 생성된 시간/날짜를 파싱합니다. 참고로 strtotime()은 윈도우 환경에서는 지원하지 않습니다.

예제 파일 | **strtotime.php**

```
1 <?php
2     $format = '%d/%m/%Y %H:%M:%S';
3     $strf = strtotime($format);
4
5     echo "$strf\n";
6
7     print_r(strtotime($strf, $format));
8
9 ?>
```

화면 출력

08/08/2017 15:06:04

```
Array ( [tm_sec] => 4 [tm_min] => 6 [tm_hour] => 15 [tm_mday] => 8 [tm_mon] => 7 [tm_year] => 117 [tm_wday] => 2 [tm_yday] => 219 [unparsed] => )
```

05.5 유효성

사용자로 직접 날짜를 입력받는 경우 유효성을 체크하여 데이터를 처리하면 보다 안전한 처리를 할 수 있습니다.

| 내장 함수 |

```
bool checkdate ( int $month , int $day , int $year )
```

내장 함수 `checkdate()`는 입력한 날짜 정보가 유효한지를 검사할 수 있습니다. 날짜 유효성을 체크하여 결과를 논리값으로 반환합니다.

예제 파일 | **checkdate.php**

```
1  <?php
2
3      $year = "2017";
4      $month = "06";
5      $day = "33";
6
7      if (checkdate($day,$month,$year)) {
8          echo "$year - $month - $day ";
9          echo "유효한 날짜입니다.<br>";
10     } else {
11         echo "$year - $month - $day ";
12         echo "정확하지 않은 날짜입니다.<br>";
13     }
14
15  ?>
```

05.6 DateTime 클래스

PHP는 내장 함수 이외에 날짜, 시간 처리를 보다 다양하게 할 수 있는 클래스를 지원합니다.

05.6.1 클래스 정의

클래스의 정의 및 구성은 다음과 같습니다.

```
DateTime implements DateTimeInterface {
    /* Constants */
    const string ATOM = "Y-m-d\TH:i:sP" ;
    const string COOKIE = "l, d-M-Y H:i:s T" ;
    const string ISO8601 = "Y-m-d\TH:i:sO" ;
    const string RFC822 = "D, d M y H:i:s O" ;
    const string RFC850 = "l, d-M-y H:i:s T" ;
    const string RFC1036 = "D, d M y H:i:s O" ;
    const string RFC1123 = "D, d M Y H:i:s O" ;
    const string RFC2822 = "D, d M Y H:i:s O" ;
    const string RFC3339 = "Y-m-d\TH:i:sP" ;
    const string RSS = "D, d M Y H:i:s O" ;
    const string W3C = "Y-m-d\TH:i:sP" ;

    /* Methods */
    public __construct ([ string $time = "now" [, DateTimeZone $timezone = NULL ] ] )
    public DateTime add ( DateInterval $interval )
    public static DateTime createFromFormat ( string $format , string $time [,
    DateTimeZone $timezone ] )
    public static array getLastErrors ( void )
}
```

```

    public DateTime modify ( string $modify )
    public static DateTime __set_state ( array $array )
    public DateTime setDate ( int $year , int $month , int $day )
    public DateTime setISODate ( int $year , int $week [, int $day = 1 ] )
    public DateTime setTime ( int $hour , int $minute [, int $second = 0 ] )
    public DateTime setTimestamp ( int $unixtimestamp )
    public DateTime setTimezone ( DateTimeZone $timezone )
    public DateTime sub ( DateInterval $interval )
    public DateInterval diff ( DateTimeInterface $datetime2 [, bool $absolute = false ] )
    public string format ( string $format )
    public int getOffset ( void )
    public int getTimestamp ( void )
    public DateTimeZone getTimezone ( void )
    public __wakeup ( void )
}

```

05.6.2 메서드

메서드 diff()는 두 DateTime 객체 간의 차이점을 반환합니다. date_diff() 함수는 객체 지향 클래스 DateTimeZone::diff의 별칭입니다.

| 메서드 |

```

public DateInterval DateTime::diff ( DateTimeInterface $datetime2 [, bool $absolute = false ] )

```

예제 파일 | [date_diff.php](#)

```

1  <?php
2      // Object oriented style
3      $datetime1 = new DateTime('2017-08-08');
4      $datetime2 = new DateTime('2017-08-13');
5      $interval = $datetime1->diff($datetime2);

```

```

6      echo $interval->format('%R%a days');
7
8      echo "<br>";
9      // Procedural style
10     $datetime1 = date_create('2017-08-11');
11     $datetime2 = date_create('2017-08-13');
12     $interval = date_diff($datetime1, $datetime2);
13     echo $interval->format('%R%a days');
14
15     ?>

```

화면 출력

```

+5 days
+2 days

```

| 메서드 |

```
public DateTime DateTime::sub ( DateInterval $interval )
```

메서드 sub()는 DateTime 객체에서 일, 월, 년, 시, 분, 초를 뺍니다. date_sub() 함수는 객체지향 클래스 DateTimeZone::sub의 별칭입니다.

예제 파일 | **date_sub.php**

```

1  <?php
2      // Object oriented style
3      $date = new DateTime('2017-08-20');
4      $date->sub(new DateInterval('P10D'));
5      echo $date->format('Y-m-d') . "<br>";
6
7      // Procedural style
8      $date = date_create('2017-08-20');
9      date_sub($date, date_interval_create_from_date_string('15 days'));
10     echo date_format($date, 'Y-m-d');
11
12     ?>

```

화면 출력

```
2017-08-10
2017-08-05
```

| 메서드 |

```
public string DateTime::format ( string $format )
```

메서드 format()은 주어진 형식에 따라 날짜 형식을 반환합니다. date_format() 함수는 객체지향 클래스 DateTimeZone::format의 별칭입니다.

예제 파일 | date_format.php

```
1  <?php
2      // Object oriented style
3      $date = new DateTime('2018-07-01');
4      echo $date->format('Y-m-d H:i:s');
5
6      echo "<br>";
7
8      // Procedural style
9      $date = date_create('2018-08-01');
10     echo date_format($date, 'Y-m-d H:i:s');
11
12  ?>
```

화면 출력

```
2018-07-01 00:00:00
2018-08-01 00:00:00
```

| 메서드 |

```
public DateTime DateTime::setISODate ( int $year , int $week [, int $day = 1 ] )
```

메서드 setISODate()는 ISO 날짜를 설정합니다. date_isodate_set() 함수는 객체지향

클래스 DateTimeZone::setISODate의 별칭입니다.

예제 파일 | **date_isodate_set.php**

```
1  <?php
2      // Object oriented style
3      $date = new DateTime();
4
5      $date->setISODate(2017, 2);
6      echo $date->format('Y-m-d') . "<br>";
7
8      $date->setISODate(2017, 2, 7);
9      echo $date->format('Y-m-d') . "<br>";
10
11     // Procedural style
12     $date = date_create();
13
14     date_isodate_set($date, 2018, 2);
15     echo date_format($date, 'Y-m-d') . "<br>";
16
17     date_isodate_set($date, 2018, 2, 7);
18     echo date_format($date, 'Y-m-d') . "<br>";
19  ?>
```

화면 출력

```
2017-01-09
2017-01-15
2018-01-08
2018-01-14
```

| 메서드 |

```
public DateTime DateTime::setTime ( int $hour , int $minute [, int $second = 0 ] )
```

메서드 setTime()은 시간을 설정합니다. date_time_set() 함수는 객체지향 클래스 DateTimeZone::setTime의 별칭입니다.

예제 파일 | `date_time_set.php`

```
1  <?php
2      // Object oriented style
3      $date = new DateTime('2017-08-08');
4
5      $date->setTime(14, 55);
6      echo $date->format('Y-m-d H:i:s') . "<br>";
7
8      $date->setTime(14, 55, 24);
9      echo $date->format('Y-m-d H:i:s') . "<br>";
10
11     // Procedural style
12     $date = date_create('2017-08-09');
13
14     date_time_set($date, 14, 55);
15     echo date_format($date, 'Y-m-d H:i:s') . "<br>";
16
17     date_time_set($date, 14, 55, 24);
18     echo date_format($date, 'Y-m-d H:i:s') . "<br>";
19
20  ?>
```

화면 출력

```
2017-08-08 14:55:00
2017-08-08 14:55:24
2017-08-09 14:55:00
2017-08-09 14:55:24
```

| 메서드 |

```
public DateTime DateTime::modify ( string $modify )
```

메서드 `modify()`는 타임 스탬프를 변경합니다. `date_modify()` 함수는 객체지향 클래스 `DateTimeZone::modify`의 별칭입니다.

예제 파일 | `date_modify.php`

```
1  <?php
2      // Object oriented style
3      $date = new DateTime('2017-12-12');
4      $date->modify('+1 day');
5      echo $date->format('Y-m-d');
6
7      echo "<br>";
8
9      // Procedural style
10     $date = date_create('2018-12-24');
11     date_modify($date, '+1 day');
12     echo date_format($date, 'Y-m-d');
13
14  ?>
```

화면 출력

2017-12-13
2018-12-25

| 메서드 |

```
public int DateTime::getTimestamp ( void )
```

메서드 `getTimestamp()`는 Unix 타임 스탬프를 가져옵니다. `date_timestamp_get()` 함수는 객체지향 클래스 `DateTimeZone::getTimestamp()`의 별칭입니다.

예제 파일 | `date_timestamp_get.php`

```
1  <?php
2      // Object oriented style
3      $date = new DateTime();
4      echo $date->getTimestamp();
5
6      echo "<br>";
7
8      // Procedural style
```

```

9     $date = date_create();
10    echo date_timestamp_get($date);
11
12    ?>

```

화면 출력

```

1502176161
1502176161

```

| 메서드 |

```

public DateTime DateTime::setTimestamp ( int $unixtimestamp )

```

메서드 `setTimestamp()`는 Unix 타임 스탬프로 날짜와 시간을 설정합니다. `date_timestamp_set()` 함수는 객체지향 클래스 `DateTimeZone::setTimestamp`의 별칭입니다.

예제 파일 | `date_timestamp_set.php`

```

1  <?php
2      // Object oriented style
3      $date = new DateTime();
4      echo $date->format('U = Y-m-d H:i:s') . "<br>";
5
6      $date->setTimestamp(1502176161);
7      echo $date->format('U = Y-m-d H:i:s') . "<br>";
8
9      // Procedural style
10     $date = date_create();
11     echo date_format($date, 'U = Y-m-d H:i:s') . "<br>";
12
13     date_timestamp_set($date, 1502176161);
14     echo date_format($date, 'U = Y-m-d H:i:s') . "<br>";
15
16     ?>

```

화면 출력

```
1502176395 = 2017-08-08 07:13:15
1502176161 = 2017-08-08 07:09:21
1502176395 = 2017-08-08 07:13:15
1502176161 = 2017-08-08 07:09:21
```

| 메서드 |

```
public DateTime DateTime::setTimezone ( DateTimeZone $timezone )
```

메서드 setTimezone()은 DateTime 표준 시간대를 설정합니다. date_timezone_set() 함수는 객체지향 클래스 DateTimeZone::setTimezone의 별칭입니다.

예제 파일 | date_timezone_set.php

```
1  <?php
2      // Object oriented style
3      $date = new DateTime('2017-08-01', new DateTimeZone('Asia/Seoul'));
4      echo $date->format('Y-m-d H:i:sP') . "<br>";
5
6      $date->setTimezone(new DateTimeZone('Europe/London'));
7      echo $date->format('Y-m-d H:i:sP') . "<br>";
8
9      // Procedural style
10     $date = date_create('2017-08-08', timezone_open('Asia/Tokyo'));
11     echo date_format($date, 'Y-m-d H:i:sP') . "<br>";
12
13     date_timezone_set($date, timezone_open('America/New_York'));
14     echo date_format($date, 'Y-m-d H:i:sP') . "<br>";
15
16  ?>
```

화면 출력

```
2017-08-01 00:00:00+09:00
2017-07-31 16:00:00+01:00
2017-08-08 00:00:00+09:00
2017-08-07 11:00:00-04:00
```

| 메서드 |

```
public DateTimeZone DateTime::getTimezone ( void )
```

메서드 `getTimezone()`은 `DateTime`을 기준으로 시간대를 반환합니다. `date_timezone_get()` 함수는 객체지향 클래스 `DateTimeZone::getTimezone`의 별칭입니다.

예제 파일 | `date_timezone_get.php`

```
1  <?php
2      // Object oriented style
3      $date = new DateTime(null, new DateTimeZone('Asia/Seoul'));
4      $tz = $date->getTimezone();
5      echo $tz->getName();
6
7      echo "<br>";
8
9      // Procedural style
10     $date = date_create(null, timezone_open('America/New_York'));
11     $tz = date_timezone_get($date);
12     echo timezone_name_get($tz);
13
14  ?>
```

화면 출력

```
Asia/Seoul
America/New_York
```

| 메서드 |

```
public int DateTime::getOffset ( void )
```

메서드 `getOffset()`은 시간대 오프셋을 반환합니다. `date_offset_get()` 함수는 객체지향 클래스 `DateTimeZone::getOffset`의 별칭입니다.

예제 파일 | **date_offset_get.php**

```
1 <?php
2     // Object oriented style
3     $winter = new DateTime('2018-12-21', new DateTimeZone('America/New_York'));
4     $summer = new DateTime('2017-08-8', new DateTimeZone('America/New_York'));
5
6     echo $winter->getOffset() . "<br>";
7     echo $summer->getOffset() . "<br>";
8
9     // Procedural style
10    $winter = date_create('2018-12-21', timezone_open('America/New_York'));
11    $summer = date_create('2017-08-9', timezone_open('America/New_York'));
12
13    echo date_offset_get($winter) . "<br>";
14    echo date_offset_get($summer) . "<br>";
15
16 ?>
```

화면 출력

```
-18000
-14400
-18000
-14400
```

| 메서드 |

```
public static array DateTime::getLastErrors ( void )
```

메서드 `getLastErrors()`는 경고 및 오류를 반환합니다. `date_get_last_errors()` 함수는 객체지향 클래스 `DateTimeZone::getLastErrors`의 별칭입니다.

예제 파일 | **date_get_last_errors.php**

```
1 <?php
2     // Object oriented style
3     try {
```

```

4      $date = new DateTime('asdfasdf');
5  } catch (Exception $e) {
6      // 데모용
7      print_r(DateTime::getLastErrors());
8
9      // 실제 객체지향적인 방법
10     // echo $e->getMessage();
11 }
12
13 //Procedural style
14 $date = date_create('asdfasdf');
15 print_r(date_get_last_errors());
16
17 ?>

```

화면 출력

```

Array ( [warning_count] => 1 [warnings] => Array ( [6] => Double timezone
specification ) [error_count] => 1 [errors] => Array ( [0] => The timezone
could not be found in the database ) ) Array ( [warning_count] => 1 [warnings]
=> Array ( [6] => Double timezone specification ) [error_count] => 1 [errors]
=> Array ( [0] => The timezone could not be found in the database ) )

```

05.7 달력

PHP의 달력 확장 기능은 달력 처리를 쉽고 간소하게 하기 위한 함수들을 지원합니다. 달력 함수들은 BC 4713부터 줄리안 데이 카운트(Julian Day Count)를 기준으로 합니다.

달력 형식을 변환하려면 먼저 줄리안 일수로 변환한 다음 원하는 달력으로 변환해야 합니다. Julian 일수와 Julian 달력은 서로 다릅니다.

05.7.1 설정

이 기능을 사용하기 위해서는 `--enable-calendar` 옵션을 적용하여 컴파일되어야 합니다. 윈도우 버전은 확장 기능으로 빌드인되어 있습니다.

| 내장 함수 |

int **cal_days_in_month** (int \$calendar , int \$month , int \$year)

내장 함수 cal_days_in_month()는 지정한 연도와 월의 마지막 일수를 반환합니다.

예제 파일 | **cal_days_in_month.php**

```
1 <?php
2     $number = cal_days_in_month(CAL_GREGORIAN, 8, 2017);
3     echo "2017년 8월의 마지막 일자는 {$number} 입니다.";
4
5 ?>
```

화면 출력

2017년 8월의 마지막 일자는 31 입니다.

| 내장 함수 |

array **cal_from_jd** (int \$jd , int \$calendar)

내장 함수 cal_from_jd()는 Julian Day Count를 지정된 달력의 날짜로 변환합니다.

예제 파일 | **cal_from_jd.php**

```
1 <?php
2     $today = unixtojd(mktime(0, 0, 0, 8, 6, 2017));
3     print_r(cal_from_jd($today, CAL_GREGORIAN));
4
5 ?>
```

화면 출력

Array ([date] => 8/6/2017 [month] => 8 [day] => 6 [year] => 2017 [dow] => 0
[abbrevdayname] => Sun [dayname] => Sunday [abbrevmonth] => Aug [monthname]
=> August)

| 내장 함수 |

```
array cal_info ([ int $calendar = -1 ] )
```

내장 함수 `cal_info()`는 캘린더에 대한 정보를 반환합니다.

- 0 또는 `CAL_GREGORIAN` – Gregorian Calendar
- 1 또는 `CAL_JULIAN` – Julian Calendar
- 2 또는 `CAL_JEWISH` – Jewish Calendar
- 3 또는 `CAL_FRENCH` – French Revolutionary Calendar

예제 파일 | `cal_info.php`

```
1 <?php
2     $info = cal_info(0);
3     print_r($info);
4
5 ?>
```

화면 출력

```
Array (
[months] => Array ( [1] => January [2] => February [3] => March [4] => April [5]
=> May [6] => June [7] => July [8] => August [9] => September [10] => October [11]
=> November [12] => December )
[abbrevmonths] => Array ( [1] => Jan [2] => Feb [3] => Mar [4] => Apr [5] =>
May [6] => Jun [7] => Jul [8] => Aug [9] => Sep [10] => Oct [11] => Nov [12]
=> Dec )
[maxdaysinmonth] => 31
[calname] => Gregorian
[calsymbol] => CAL_GREGORIAN )
```

05.7.2 Julian

내장 함수 `cal_to_jd()`는 달력에서 줄리안 일수를 반환합니다.

| 내장 함수 |

```
int cal_to_jd ( int $calendar , int $month , int $day , int $year )
```

예제 파일 | **cal_to_jd.php**

```
1  <?php
2      echo cal_to_jd( CAL_GREGORIAN , 8 , 6 , 2017 );
3
4  ?>
```

화면 출력

2457972

| 내장 함수 |

```
int gregoriantojd ( int $month , int $day , int $year )
```

내장 함수 `gregoriantojd()`는 Gregorian Calendar 기준 Julian 일수를 반환합니다.

예제 파일 | **gregoriantojd.php**

```
1  <?php
2      echo gregoriantojd( 8 , 6 , 2017 );
3  ?>
```

화면 출력

2457972

| 내장 함수 |

```
mixed jddayofweek ( int $julianday [, int $mode = CAL_DOW_DAYNO ] )
```

내장 함수 `jddayofweek()`는 요일을 반환합니다.

예제 파일 | **jddayofweek.php**

```
1  <?php
2      $julianday = gregoriantojd( 8 , 6 , 2017 );
3
4      // Return the day number as an int (0=Sunday, 1=Monday, etc)
5      echo jddayofweek($julianday) . "<br>";
6
7      // 1 Returns string containing the day of week (English-Gregorian)
8      echo jddayofweek($julianday,1) . "<br>";
9
10     // 2 Return a string containing the abbreviated day of week (English-
    Gregorian)
11     echo jddayofweek($julianday,2) . "<br>";
12
13  ?>
```

화면 출력

0
Sunday
Sun

| 내장 함수 |

string **jdmonthname** (int \$julianday , int \$mode)

내장 함수 `jdmonthname()`은 월 이름을 반환합니다.

예제 파일 | **jdmonthname.php**

```
1  <?php
2      $julianday = gregoriantojd( 8 , 6 , 2017 );
3
4      // 0
5      // Gregorian - abbreviated Jan, Feb, Mar, Apr, May, Jun, Jul, Aug,
    Sep, Oct, Nov, Dec
6      echo jdmonthname ( $julianday , 0 ) . "<br>";
7
8      // 1
```

```

9      // Gregorian January, February, March, April, May, June, July, August,
      September, October, November, December
10     echo jdmonthname ( $julianday , 1 ) . "<br>";
11
12     // 2
13     // Julian - abbreviated Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep,
      Oct, Nov, Dec
14     echo jdmonthname ( $julianday , 2 ) . "<br>";
15
16     // 3
17     // Julian January, February, March, April, May, June, July, August,
      September, October, November, December
18     echo jdmonthname ( $julianday , 3 ) . "<br>";
19
20     // 4
21     // Jewish Tishri, Heshvan, Kislev, Tevet, Shevat, AdarI, AdarII,
      Nisan, Iyyar, Sivan, Tammuz, Av, Elul
22     echo jdmonthname ( $julianday , 4 ) . "<br>";
23
24     // 5
25     // French Republican Vendemiaire, Brumaire, Frimaire, Nivose,
      Pluviose, Ventose, Germinal, Floreal, Prairial, Messidor, Thermidor,
      Fructidor, Extra
26     echo jdmonthname ( $julianday , 5 ) . "<br>";
27
28     ?>

```

화면 출력

```

Aug
August
Jul
July
Av

```

| 내장 함수 |

```
string jdtojulian ( int $julianday )
```

내장 함수 `jdtojulian()`은 Julian 일수를 Julian 날짜로 변환합니다.

| 내장 함수 |

```
int juliantojd ( int $month , int $day , int $year )
```

내장 함수 `juliantojd()`는 Julian 날짜를 Julian 일수로 변환합니다.

| 내장 함수 |

```
int jdtonix ( int $day )
```

내장 함수 `jdtonix()`는 Julian 일수를 Unix 타임 스탬프로 변환합니다.

| 내장 함수 |

```
int unixtojd ([ int $timestamp = time() ] )
```

내장 함수 `unixtojd()`는 Unix 타임 스탬프를 Julian 일수로 변환합니다.

05.7.3 Gregorian

Gregorian 기준의 날짜를 변환하여 사용할 수 있습니다. PHP에서는 이와 관련된 몇 가지 함수를 지원합니다.

| 내장 함수 |

```
string jdtogregorian ( int $julianday )
```

내장 함수 `jdtogregorian()`는 Julian 일수를 Gregorian 날짜로 변환합니다.

| 내장 함수 |

```
int gregoriantojd ( int $month , int $day , int $year )
```

내장 함수 `gregoriantojd()`는 Gregorian 날짜를 Julian 일수로 변환합니다.

| 내장 함수 |

```
string jdtogregorian ( int $julianday )
```

내장 함수 `jdtogregorian()`은 Julian 일수를 Gregorian 날짜로 변환합니다.

05.7.4 Jewish

Jewish 기준의 날짜를 변환하여 사용할 수 있습니다. PHP에서는 이와 관련된 몇 가지 함수를 지원합니다.

| 내장 함수 |

```
string jdtojewish ( int $juliandaycount [, bool $hebrew = false [, int $fl = 0 ]] )
```

내장 함수 `jdtojewish()`는 Julian 일수를 Jewish 달력 일자로 변환합니다.

| 내장 함수 |

```
int jewishtojd ( int $month , int $day , int $year )
```

내장 함수 `jewishtojd()`는 Jewish 날짜를 Julian 일수로 변환합니다.

05.7.5 French Republican

French Republican 기준의 날짜를 변환하여 사용할 수 있습니다. PHP에서는 이와 관련된 몇 가지 함수를 지원합니다.

| 내장 함수 |

string **jdtofrrench** (int \$juliandaycount)

French Republican Calendar를 Julian 일수로 변환합니다.

예제 파일 | **jdtofrrench.php**

```
1 <?php
2     $julianday = gregoriantojd( 8 , 6 , 2017 );
3     echo jdtofrrench($julianday);
4
5 ?>
```

| 내장 함수 |

int **frenchtojd** (int \$month , int \$day , int \$year)

내장 함수 frenchtojd()는 French Revolutionary Calendar 기준 Julian 일수를 반환합니다.

예제 파일 | **frenchtojd.php**

```
1 <?php
2     echo frenchtojd( 8 , 6 , 2017 );
3
4 ?>
```

05.7.6 부활절

부활절 관련 함수들을 지원합니다.

| 내장 함수 |

```
int easter_date ([ int $year = date("Y") ] )
```

내장 함수 `easter_date()`는 지정한 연도의 부활절 자정에 대한 유닉스 타임 스탬프를 반환합니다.

예제 파일 | [easter_date.php](#)

```
1  <?php
2      echo "2000 부활절 자정 = ". date("M-d-Y", easter_date(2000));
3      echo "<br>";
4
5      echo "2010 부활절 자정 = ". date("M-d-Y", easter_date(2010));
6      echo "<br>";
7
8      echo "2017 부활절 자정 = ". date("M-d-Y", easter_date(2017));
9      echo "<br>";
10
11  ?>
```

화면 출력

```
2000 부활절 자정 = Apr-22-2000
2010 부활절 자정 = Apr-03-2010
2017 부활절 자정 = Apr-15-2017
```

| 내장 함수 |

```
int easter_days ([ int $year = date("Y") [, int $method = CAL_EASTER_DEFAULT ] ] )
```

내장 함수 `easter_days()`는 부활절의 해에 3월 21일 이후 일수를 반환합니다.

예제 파일 | **easter_days.php**

```
1  <?php
2      echo easter_days(1999) . "<br>";
3      // 14, i.e. April 4
4
5      echo easter_days(1492) . "<br>";
6      // 32, i.e. April 22
7
8      echo easter_days(1913) . "<br>";
9      // 2, i.e. March 23
10
11  ?>
```

화면 출력

```
14
32
2
```

05.8 타임존

전 세계를 대상으로 서비스를 기획한다고 하면 지역별 시간대 관리는 매우 중요합니다. 각 대륙, 지역별로 구분하여 시간 관리 프로그램을 만들어야 정확한 월드 와이드 서비스를 구축할 수 있습니다.

05.8.1 클래스

PHP는 타임존을 관리 및 설정할 수 있는 `DateTimeZone` 클래스를 지원합니다.

```
DateTimeZone {
    /* Constants */
    const integer AFRICA = 1 ;
    const integer AMERICA = 2 ;
    const integer ANTARCTICA = 4 ;
    const integer ARCTIC = 8 ;
```

```

const integer ASIA = 16 ;
const integer ATLANTIC = 32 ;
const integer AUSTRALIA = 64 ;
const integer EUROPE = 128 ;
const integer INDIAN = 256 ;
const integer PACIFIC = 512 ;
const integer UTC = 1024 ;
const integer ALL = 2047 ;
const integer ALL_WITH_BC = 4095 ;
const integer PER_COUNTRY = 4096 ;

/* Methods */
public __construct ( string $timezone )
public array getLocation ( void )
public string getName ( void )
public int getOffset ( DateTime $datetime )
public array getTransitions ([ int $timestamp_begin [, int $timestamp_end ]])
public static array listAbbreviations ( void )
public static array listIdentifiers ([ int $what = DateTimeZone::ALL [, string
$country = NULL ]])
}

```

05.8.2 메서드

| 메서드 |

```

public DateTimeZone::__construct ( string $timezone )

```

메서드 __construct()는 새 DateTimeZone 객체를 만듭니다. timezone_open() 함수는 객체지향 클래스 DateTimeZone::__construct의 별칭입니다.

예제 파일 | `timezone_open.php`

```
1  <?php
2      // Error handling by catching exceptions
3      $timezones = array('Europe/London', 'Mars/Phobos', 'Jupiter/Europa');
4
5      foreach ($timezones as $tz) {
6          try {
7              $mars = new DateTimeZone($tz);
8          } catch(Exception $e) {
9              echo $e->getMessage() . '<br />';
10         }
11     }
12
13  ?>
```

화면 출력

```
DateTimeZone::__construct(): Unknown or bad timezone (Mars/Phobos)
DateTimeZone::__construct(): Unknown or bad timezone (Jupiter/Europa)
```

| 메서드 |

```
public array DateTimeZone::getLocation ( void )
```

메서드 `getLocation()`은 시간대의 위치 정보를 반환합니다. `timezone_location_get()` 함수는 객체지향 클래스 `DateTimeZone::getLocation`의 별칭입니다. 국가 코드, 위도/경도 및 주석을 포함하여 시간대의 위치 정보를 반환합니다.

예제 파일 | `timezone_location_get.php`

```
1  <?php
2      $tz = new DateTimeZone("Europe/Prague");
3      print_r($tz->getLocation());
4      print_r(timezone_location_get($tz));
5
6  ?>
```

화면 출력

```
Array ( [country_code] => CZ [latitude] => 50.08333 [longitude] => 14.43333  
[comments] => ) Array ( [country_code] => CZ [latitude] => 50.08333 [longitude]  
=> 14.43333 [comments] => )
```

| 메서드 |

```
public string DateTimeZone::getName ( void )
```

메서드 getName()은 타임 존의 이름을 반환합니다. timezone_name_get() 함수는 객체지향 클래스 DateTimeZone::getName의 별칭입니다.

| 메서드 |

```
public static array DateTimeZone::listAbbreviations ( void )
```

메서드 listAbbreviations()는 dst, offset 및 시간대 이름이 들어 있는 연관 배열을 반환합니다. timezone_abbreviations_list() 함수는 객체지향 클래스 DateTimeZone::listAbbreviations의 별칭입니다.

예제 파일 | [timezone_abbreviations_list.php](#)

```
1 <?php  
2     $timezone_abbreviations = DateTimeZone::listAbbreviations();  
3     print_r($timezone_abbreviations["acst"]);  
4  
5 ?>
```

화면 출력

```
Array ( [0] => Array ( [dst] => 1 [offset] => -14400 [timezone_id] => America/  
Porto_Acre ) [1] => Array ( [dst] => [offset] => 32400 [timezone_id] =>  
Australia/Adelaide ) [2] => Array ( [dst] => [offset] => 34200 [timezone_id]  
=> Australia/Adelaide ) [3] => Array ( [dst] => 1 [offset] => -14400 [timezone_  
id] => America/Eirunepe ) [4] => Array ( [dst] => 1 [offset] => -14400 [timezone_  
id] => America/Rio_Branco ) [5] => Array ( [dst] => 1 [offset] => -14400
```

```
[timezone_id] => Brazil/Acre ) [6] => Array ( [dst] => [offset] => 32400
[timezone_id] => Australia/Broken_Hill ) [7] => Array ( [dst] => [offset] =>
32400 [timezone_id] => Australia/Darwin ) [8] => Array ( [dst] => [offset] =>
32400 [timezone_id] => Australia/North ) [9] => Array ( [dst] => [offset] =>
32400 [timezone_id] => Australia/South ) [10] => Array ( [dst] => [offset] =>
32400 [timezone_id] => Australia/Yancowinna ) [11] => Array ( [dst] => [offset]
=> 34200 [timezone_id] => Asia/Jayapura ) [12] => Array ( [dst] => [offset] =>
34200 [timezone_id] => Australia/Broken_Hill ) [13] => Array ( [dst] => [offset]
=> 34200 [timezone_id] => Australia/Darwin ) [14] => Array ( [dst] => [offset]
=> 34200 [timezone_id] => Australia/North ) [15] => Array ( [dst] => [offset]
=> 34200 [timezone_id] => Australia/South ) [16] => Array ( [dst] => [offset]
=> 34200 [timezone_id] => Australia/Yancowinna ) )
```

| 메서드 |

```
public static array DateTimeZone::listIdentifiers ([ int $what = DateTimeZone::ALL [,
string $country = NULL ] ] )
```

메서드 `listIdentifiers()`는 정의된 모든 시간대 식별자가 포함된 숫자로 색인된 배열을 반환합니다. `timezone_identifiers_list()` 함수는 객체지향 클래스 `DateTimeZone::listIdentifiers`의 별칭입니다.

예제 파일 | `timezone_identifiers_list.php`

```
1 <?php
2     $timezone_identifiers = DateTimeZone::listIdentifiers();
3     for ($i=0; $i < 5; $i++) {
4         echo "$timezone_identifiers[$i]<br>";
5     }
6
7 ?>
```

화면 출력

```
Africa/Abidjan
Africa/Accra
Africa/Addis_Ababa
Africa/Algiers
Africa/Asmara
```

| 메서드 |

```
public int DateTimeZone::getOffset ( DateTime $datetime )
```

메서드 `getOffset()`은 GMT로부터의 시간대 오프셋을 반환합니다. `timezone_offset_get()` 함수는 객체지향 클래스 `DateTimeZone::getOffset`의 별칭입니다.

예제 파일 | [timezone_offset_get.php](#)

```
1  <?php
2
3      $dateTimeZoneAmsterdam= new DateTimeZone("Europe/Amsterdam");
4      $dateTimeZoneSeoul = new DateTimeZone("Asia/Seoul");
5
6      $dateTimeAmsterdam = new DateTime("now", $dateTimeZoneAmsterdam);
7      $dateTimeSeoul = new DateTime("now", $dateTimeZoneSeoul);
8
9      $timeOffset = $dateTimeZoneSeoul->getOffset($dateTimeAmsterdam);
10
11      var_dump($timeOffset);
12
13  ?>
```

화면 출력

```
int(32400)
```

| 메서드 |

```
public array DateTimeZone::getTransitions ([ int $timestamp_begin [, int $timestamp_end ] ] )
```

메서드 `getTransitions()`은 시간대의 모든 전환을 반환합니다. `timezone_transitions_get()` 함수는 객체지향 클래스 `DateTimeZone::getTransitions`의 별칭입니다.

예제 파일 | `timezone_transitions_get.php`

```
1 <?php
2     $timezone = new DateTimeZone("Europe/London");
3     $transitions = $timezone->getTransitions();
4     print_r(array_slice($transitions, 0, 3));
5
6 ?>
```

화면 출력

```
Array ( [0] => Array ( [ts] => -2147483648 [time] => 1901-12-13T20:45:52+0000
[offset] => -75 [isdst] => [abbr] => LMT ) [1] => Array ( [ts] => -2147483648
[time] => 1901-12-13T20:45:52+0000 [offset] => 0 [isdst] => [abbr] => GMT ) [2]
=> Array ( [ts] => -1691964000 [time] => 1916-05-21T02:00:00+0000 [offset] =>
3600 [isdst] => 1 [abbr] => BST ) )
```

| 내장 함수 |

`string timezone_version_get (void)`

내장 함수 `timezone_version_get()`은 `timezonedb`의 버전을 가져옵니다.

예제 파일 | `timezone_version_get.php`

```
1 <?php
2     echo timezone_version_get();
3
4 ?>
```

화면 출력

2017.2

| 내장 함수 |

`bool date_default_timezone_set (string $timezone_identifier)`

내장 함수 `date_default_timezone_set()`은 날짜/시간 함수에서 사용하는 기본 시간대를 설정합니다.

| 내장 함수 |

```
string date_default_timezone_get ( void )
```

내장 함수 `date_default_timezone_get()`은 날짜/시간 함수에서 사용하는 기본 표준 시간대를 가져옵니다.

예제 파일 | `date_default_timezone_get.php`

```
1  <?php
2      date_default_timezone_set('Asia/Seoul');
3
4      if (date_default_timezone_get()) {
5          echo 'date_default_timezone_set: ' . date_default_timezone_get() .
6              '<br />';
7      }
8
9      if (ini_get('date.timezone')) {
10         echo 'date.timezone: ' . ini_get('date.timezone');
11     }
12  ?>
```

화면 출력

date_default_timezone_set: Asia/Seoul

05.8.3 타임존 명칭

전 세계 타임존은 다음과 같이 구분할 수 있습니다.

Africa

- Africa/Abidjan Africa/Accra Africa/Addis_Ababa Africa/Algiers Africa/

Asmara

- Africa/Asmera Africa/Bamako Africa/Bangui Africa/Banjul Africa/Bissau
- Africa/Blantyre Africa/Brazzaville Africa/Bujumbura Africa/Cairo Africa/Casablanca
- Africa/Ceuta Africa/Conakry Africa/Dakar Africa/Dar_es_Salaam Africa/Djibouti
- Africa/Douala Africa/El_Aaiun Africa/Freetown Africa/Gaborone Africa/Harare
- Africa/Johannesburg Africa/Juba Africa/Kampala Africa/Khartoum Africa/Kigali
- Africa/Kinshasa Africa/Lagos Africa/Libreville Africa/Lome Africa/Luanda
- Africa/Lubumbashi Africa/Lusaka Africa/Malabo Africa/Maputo Africa/Maseru
- Africa/Mbabane Africa/Mogadishu Africa/Monrovia Africa/Nairobi Africa/Ndjamena
- Africa/Niamey Africa/Nouakchott Africa/Ouagadougou Africa/Porto—Novo Africa/Sao_Tome
- Africa/Timbuktu Africa/Tripoli Africa/Tunis Africa/Windhoek

America

- America/Adak America/Anchorage America/Anguilla
- America/Antigua America/Araguaina America/Argentina/Buenos_Aires
- America/Argentina/Catamarca America/Argentina/ComodRivadavia America/Argentina/Cordoba
- America/Argentina/Jujuy America/Argentina/La_Rioja America/Argentina/Mendoza
- America/Argentina/Rio_Gallegos America/Argentina/Salta America/Argentina/San_Juan
- America/Argentina/San_Luis America/Argentina/Tucuman America/

Argentina/Ushuaia

- America/Aruba America/Asuncion America/Atikokan
- America/Atka America/Bahia America/Bahia_Banderas
- America/Barbados America/Belem America/Belize
- America/Blanc–Sablon America/Boa_Vista America/Bogota
- America/Boise America/Buenos_Aires America/Cambridge_Bay
- America/Campo_Grande America/Cancun America/Caracas
- America/Catamarca America/Cayenne America/Cayman
- America/Chicago America/Chihuahua America/Coral_Harbour
- America/Cordoba America/Costa_Rica America/Creston
- America/Cuiaba America/Curacao America/Danmarkshavn
- America/Dawson America/Dawson_Creek America/Denver
- America/Detroit America/Dominica America/Edmonton
- America/Eirunepe America/El_Salvador America/Ensenada
- America/Fort_Wayne America/Fortaleza America/Glace_Bay
- America/Godthab America/Goose_Bay America/Grand_Turk
- America/Grenada America/Guadeloupe America/Guatemala
- America/Guayaquil America/Guyana America/Halifax
- America/Havana America/Hermosillo America/Indiana/Indianapolis
- America/Indiana/Knox America/Indiana/Marengo America/Indiana/Petersburg
- America/Indiana/Tell_City America/Indiana/Vevay America/Indiana/Vincennes
- America/Indiana/Winamac America/Indianapolis America/Inuvik
- America/Iqaluit America/Jamaica America/Jujuy
- America/Juneau America/Kentucky/Louisville America/Kentucky/Monticello
- America/Knox_IN America/Kralendijk America/La_Paz
- America/Lima America/Los_Angeles America/Louisville

- America/Lower_Princes America/Maceio America/Managua
- America/Manaus America/Marigot America/Martinique
- America/Matamoros America/Mazatlan America/Mendoza
- America/Menominee America/Merida America/Metlakatla
- America/Mexico_City America/Miquelon America/Moncton
- America/Monterrey America/Montevideo America/Montreal
- America/Montserrat America/Nassau America/New_York
- America/Nipigon America/Nome America/Noronha
- America/North_Dakota/Beulah America/North_Dakota/Center America/
North_Dakota/New_Salem
- America/Ojinaga America/Panama America/Pangnirtung
- America/Paramaribo America/Phoenix America/Port-au-Prince
- America/Port_of_Spain America/Porto_Acre America/Porto_Velho
- America/Puerto_Rico America/Rainy_River America/Rankin_Inlet
- America/Recife America/Regina America/Resolute
- America/Rio_Branco America/Rosario America/Santa_Isabel
- America/Santarem America/Santiago America/Santo_Domingo
- America/Sao_Paulo America/Scoresbysund America/Shiprock
- America/Sitka America/St_Barthelemy America/St_Johns
- America/St_Kitts America/St_Lucia America/St_Thomas
- America/St_Vincent America/Swift_Current America/Tegucigalpa
- America/Thule America/Thunder_Bay America/Tijuana
- America/Toronto America/Tortola America/Vancouver
- America/Virgin America/Whitehorse America/Winnipeg
- America/Yakutat America/Yellowknife

Antarctica

- Antarctica/Casey Antarctica/Davis Antarctica/DumontD'Urville Antarctica/
Macquarie Antarctica/Mawson

- Antarctica/McMurdo Antarctica/Palmer Antarctica/Rothera Antarctica/
South_Pole Antarctica/Syowa
- Antarctica/Vostok

Arctic

- Arctic/Longyearbyen

Asia

- Asia/Aden Asia/Almaty Asia/Amman Asia/Anadyr Asia/Aqtau
- Asia/Aqtobe Asia/Ashgabat Asia/Ashkhabad Asia/Baghdad Asia/Bahrain
- Asia/Baku Asia/Bangkok Asia/Beirut Asia/Bishkek Asia/Brunei
- Asia/Calcutta Asia/Choibalsan Asia/Chongqing Asia/Chungking Asia/
Colombo
- Asia/Dacca Asia/Damascus Asia/Dhaka Asia/Dili Asia/Dubai
- Asia/Dushanbe Asia/Gaza Asia/Harbin Asia/Hebron Asia/Ho_Chi_Minh
- Asia/Hong_Kong Asia/Hovd Asia/Irkutsk Asia/Istanbul Asia/Jakarta
- Asia/Jayapura Asia/Jerusalem Asia/Kabul Asia/Kamchatka Asia/Karachi
- Asia/Kashgar Asia/Kathmandu Asia/Katmandu Asia/Khandyga Asia/
Kolkata
- Asia/Krasnoyarsk Asia/Kuala_Lumpur Asia/Kuching Asia/Kuwait Asia/
Macao
- Asia/Macau Asia/Magadan Asia/Makassar Asia/Manila Asia/Muscat
- Asia/Nicosia Asia/Novokuznetsk Asia/Novosibirsk Asia/Omsk Asia/Oral
- Asia/Phnom_Penh Asia/Pontianak Asia/Pyongyang Asia/Qatar Asia/
Qyzylorda
- Asia/Rangoon Asia/Riyadh Asia/Saigon Asia/Sakhalin Asia/Samarkand
- Asia/Seoul Asia/Shanghai Asia/Singapore Asia/Taipei Asia/Tashkent
- Asia/Tbilisi Asia/Tehran Asia/Tel_Aviv Asia/Thimbu Asia/Thimphu

- Asia/Tokyo Asia/Ujung_Pandang Asia/Ulaanbaatar Asia/Ulan_Bator Asia/Urumqi
- Asia/Ust-Nera Asia/Vientiane Asia/Vladivostok Asia/Yakutsk Asia/Yekaterinburg
- Asia/Yerevan

Atlantic

- Atlantic/Azores Atlantic/Bermuda Atlantic/Canary Atlantic/Cape_Verde Atlantic/Faeroe
- Atlantic/Faroe Atlantic/Jan_Mayen Atlantic/Madeira Atlantic/Reykjavik Atlantic/South_Georgia
- Atlantic/St_Helena Atlantic/Stanley

Australia

- Australia/ACT Australia/Adelaide Australia/Brisbane Australia/Broken_Hill Australia/Canberra
- Australia/Currie Australia/Darwin Australia/Eucla Australia/Hobart Australia/LHI
- Australia/Lindeman Australia/Lord_Howe Australia/Melbourne Australia/North Australia/NSW
- Australia/Perth Australia/Queensland Australia/South Australia/Sydney Australia/Tasmania
- Australia/Victoria Australia/West Australia/Yancowinna

Europe

- Europe/Amsterdam Europe/Andorra Europe/Athens Europe/Belfast Europe/Belgrade
- Europe/Berlin Europe/Bratislava Europe/Brussels Europe/Bucharest

Europe/Budapest

- Europe/Busingen Europe/Chisinau Europe/Copenhagen Europe/Dublin
Europe/Gibraltar
- Europe/Guernsey Europe/Helsinki Europe/Isle_of_Man Europe/Istanbul
Europe/Jersey
- Europe/Kaliningrad Europe/Kiev Europe/Lisbon Europe/Ljubljana
Europe/London
- Europe/Luxembourg Europe/Madrid Europe/Malta Europe/Mariehamn
Europe/Minsk
- Europe/Monaco Europe/Moscow Europe/Nicosia Europe/Oslo Europe/
Paris
- Europe/Podgorica Europe/Prague Europe/Riga Europe/Rome Europe/
Samara
- Europe/San_Marino Europe/Sarajevo Europe/Simferopol Europe/Skopje
Europe/Sofia
- Europe/Stockholm Europe/Tallinn Europe/Tirane Europe/Tiraspol
Europe/Uzhgorod
- Europe/Vaduz Europe/Vatican Europe/Vienna Europe/Vilnius Europe/
Volgograd
- Europe/Warsaw Europe/Zagreb Europe/Zaporozhye Europe/Zurich

Indian

- Indian/Antananarivo Indian/Chagos Indian/Christmas Indian/Cocos
Indian/Comoro
- Indian/Kerguelen Indian/Mahe Indian/Maldives Indian/Mauritius Indian/
Mayotte
- Indian/Reunion

Pacific

- Pacific/Apia Pacific/Auckland Pacific/Chatham Pacific/Chuuk Pacific/Easter
- Pacific/Efate Pacific/Enderbury Pacific/Fakaofu Pacific/Fiji Pacific/Funafuti
- Pacific/Galapagos Pacific/Gambier Pacific/Guadacanal Pacific/Guam Pacific/Honolulu
- Pacific/Johnston Pacific/Kiritimati Pacific/Kosrae Pacific/Kwajalein Pacific/Majuro
- Pacific/Marquesas Pacific/Midway Pacific/Nauru Pacific/Niue Pacific/Norfolk
- Pacific/Noumea Pacific/Pago_Pago Pacific/Palau Pacific/Pitcairn Pacific/Pohnpei
- Pacific/Ponape Pacific/Port_Moresby Pacific/Rarotonga Pacific/Saipan Pacific/Samoa
- Pacific/Tahiti Pacific/Tarawa Pacific/Tongatapu Pacific/Truk Pacific/Wake
- Pacific/Wallis Pacific/Yap

06

파일 제어

컴퓨터에서 수많은 데이터를 파일 형태로 하드디스크 장치에 저장합니다. 파일 시스템은 수많은 파일들을 관리하고 입출력을 처리합니다. C 언어와 같이 전형적인 프로그램 언어들은 파일 입출력을 통하여 데이터를 저장하고 관리합니다.

PHP는 서버 사이드 웹 인터프리터 언어입니다. 하지만 PHP는 파일 시스템의 다양한 파일의 입출력을 제어함으로써 데이터를 처리할 수 있습니다. 이와 관련하여 다양한 파일을 처리할 수 있는 내장 함수들을 제공하고 있습니다. 파일 관련 함수들은 파일을 읽고, 쓰고, 디렉터리, 권한 등을 제어할 수 있습니다.

PHP에서 파일 시스템의 파일들을 제어한다는 것은 보다 향상된 PHP 응용프로그램을 개발할 수 있다는 것입니다. 템플릿 파일들을 제어하고, 환경 설정을 파일로 저장할 수 있습니다. 또한 각종 로그 기록 등을 처리하는 데 파일 처리 함수들은 매우 유용합니다.

06.1 파일 시스템

파일 시스템은 파일을 관리하는 운영체제의 일부분입니다. 파일 시스템은 운영체제별로 차이가 있으며 다양한 형태의 시스템이 존재합니다. FAT, FAT32, NTFS, Ext3 등 운영체제와 밀접한 다양한 파일 시스템 구조가 있습니다.

운영체제 및 파일 시스템의 특성에 따라서 처리하는 PHP 개발 환경은 약간의 차이가 발생합니다. 예를 들어 유닉스 플랫폼은 디렉터리 구분을 슬래시(/)로 구분합니다. 하지만 윈도우 플랫폼은 디렉터리 구분을 백슬래시(\)를 통하여 구분합니다.

또한 윈도우 시스템의 파일 시스템과 리눅스/맥OS 등의 파일 시스템의 차이로 인하여 몇몇 기능은 동작하지 않을 수 있습니다. 파일 처리 함수를 사용할 때는 이런 점을 주의하면서 개발해야 합니다.

PHP 파일 처리 함수 및 기능은 PHP 코어에 기본적으로 탑재되어 있습니다. 별도의 외부 모듈을 추가하지 않아도 바로 사용할 수 있습니다. PHP의 파일 처리 함수들은 php.ini 환경 설정의 영향을 받습니다.

06.2 디렉터리

파일에 대해서 먼저 학습을 하기 전에 디렉터리 시스템에 대해서 살펴보도록 하겠습니다. 디렉터리는 윈도우와 같은 시스템에서는 ‘폴더’라고 부르기도 합니다. 서로 같은 의미라고 보면 됩니다.

디렉터리는 수많은 파일을 구분하고 정리하기 위해서 고안된 개념입니다. 초창기 하드디스크의 용량이 매우 작고 파일들이 몇 개 안 될 때는 큰 문제가 없었습니다. 하지만 시스템이 커지고 파일들도 많아짐으로써 유사한 파일을 묶어 관리해야 하는 필요성이 생겼습니다.

디렉터리는 파일과 다르게 하나의 묶음 공간이며, 여러 개의 파일을 관리합니다. 또한 1개의 디렉터리는 또 다른 여러 개의 디렉터를 관리할 수 있습니다.

06.2.1 디렉터리 확인

파일 시스템은 파일과 디렉터를 한곳에서 출력 관리를 합니다. 대부분의 파일들은 확장자를 포함하고, 디렉터리의 경우에는 확장자를 갖지 않습니다. 하지만 파일명이 꼭 확장자를 가져야 하는 것은 아니기 때문에 입력되는 값이 파일인지, 디렉터리인지를 구분하기 어려울 수 있습니다.

디렉터리 관련 작업을 할 때는 먼저 입력한 값이 디렉터리인지를 확인하는 것이 좋습니다. 또한 존재하는 디렉터리 이름인지도 검사해야 합니다.

우리는 보통 경로를 지정할 때는 상대 경로, 절대 경로 등 다양한 방법을 사용합니다. 이러한 다양한 경로의 표현은 자칫 실수로 인하여 오류를 발생할 수 있습니다. 존재하지 않은 디렉터리 접근, 정확하지 않은 디렉터리 접근은 프로그램에 심각한 오류를 발생할 수 있습니다.

사전에 디렉터를 확인하는 코드는 파일을 처리하는 데 있어서 발생할 수 있는 오류를 줄여주고, 코드의 안정적인 동작을 도와줍니다.

PHP에서는 디렉터리의 존재 여부를 확인할 수 있는 다음과 같은 내장 함수를 지원합니다.

| 내장 함수 |

```
bool is_dir ( string $filename )
```

디렉터를 확인하는 is_dir() 함수는 매개변수로 입력된 값이 디렉터리인지 파일인지를 확인하여 논리값으로 반환합니다.

예제 파일 | file_dir-01.php

```
1  <?php
2
3      $pathDir = "testing";
4
5      if (!is_dir($pathDir)) {
```

```

6      echo "Err] $pathDir 디렉터리가 존재하지 않습니다.<br>";
7  } else {
8      echo "OK] $pathDir 디렉터리 작업이 가능합니다.<br>";
9  }
10
11  $pathDir = "abcd";
12
13  if (!is_dir($pathDir)) {
14      echo "Err] $pathDir 디렉터리가 존재하지 않습니다.<br>";
15  } else {
16      echo "OK] $pathDir 디렉터리 작업이 가능합니다.<br>";
17  }
18
19  $pathDir = "abcd/123";
20
21  if (!is_dir($pathDir)) {
22      echo "Err] $pathDir 디렉터리가 존재하지 않습니다.<br>";
23  } else {
24      echo "OK] $pathDir 디렉터리 작업이 가능합니다.<br>";
25  }
26
27  ?>

```

화면 출력

```

Err] testing 디렉터리가 존재하지 않습니다.
OK] abcd 디렉터리 작업이 가능합니다.
OK] abcd/123 디렉터리 작업이 가능합니다.

```

위의 예제는 실행하고 있는 스크립트 안에 있는 디렉터리 여부를 확인합니다. 입력한 경로 값의 디렉터리가 존재하는지, 또는 파일이 아닌 유효한 디렉터리인지를 논리값으로 반환합니다. 디렉터리 검사는 서브 디렉터리를 포함하여 한 번에 검사할 수도 있습니다.

06.2.2 디렉터리 생성

PHP 코드를 통하여 새로운 디렉터리를 생성할 수 있습니다. 디렉터리를 신규로 생성하기 위해서는 PHP 또는 사용자가 해당 디렉터리에 쓰기 권한이 있어야 합니다. 디렉터리는 수많은 파일을 관리하는 데 매우 유용합니다.

PHP에서는 새로운 디렉터리 생성을 위한 다음과 같은 내장 함수를 지원합니다.

| 내장 함수 |

```
bool mkdir ( string $pathname [, int $mode = 0777 [, bool $recursive = false [,  
resource $context ]]] )
```

PHP 내장 함수 mkdir()은 새로운 디렉터리를 생성합니다. 함수명이 리눅스 계열의 운영 체제에서 디렉터리 생성 명령인 mkdir과 매우 유사합니다.

mkdir() 함수는 몇 개의 매개변수를 동시에 입력받습니다. 하지만 기본적으로 신규 디렉터리만 생성할 때는 경로명만 입력하면 됩니다. 만일, 폴더 권한 등을 동시에 설정이 필요할 때는 두 번째 권한 설정을 함께 입력하면 됩니다.

예제 파일 | file_dir-02.php

```
1  <?php
2      $pathDir = "aaa";
3
4      if (!is_dir($pathDir)) {
5          echo "Err] $pathDir 디렉터리가 존재하지 않습니다.<br>";
6
7          echo "새로운 디렉터리를 생성합니다.<br>";
8          if (mkdir($pathDir)) {
9              echo "$pathDir 디렉터리를 생성했습니다.<br>";
10         } else {
11             echo "Err] 디렉터리 생성 실패! <br>";
12         }
13
14     } else {
15         echo "OK] $pathDir 디렉터리 작업이 가능합니다.<br>";
16     }
17
18  ?>
```

화면 출력

Err] aaa 디렉터리가 존재하지 않습니다.

새로운 디렉터리를 생성합니다.
aaa 디렉터를 생성했습니다.

06.2.3 서브 디렉터리

디렉터리는 다수의 또 다른 디렉터를 포함하고 관리할 수 있습니다. 따라서 디렉터리 안에 새로운 서브 디렉터를 추가하여 생성할 수 있습니다.

PHP 내장 함수인 `mkdir()`은 새로운 디렉터를 생성합니다. 만일 서브 디렉터를 생성할 때는 상위 디렉터리가 존재해야 합니다. 상위 디렉터리가 없이 하위 디렉터를 갖는 멀티 레벨의 디렉터를 한 번에 생성할 수 없습니다.

다수의 단계를 갖는 디렉터를 생성할 때는 항상 상위 단계의 디렉터리가 미리 있는지 확인해야 합니다. 이러한 사전 디렉터리 존재 여부를 확인하는 것은 코드상 복잡한 구조를 띠게 됩니다. 이런 경우에는 `mkdir()` 함수의 세 번째 `recursive` 옵션을 사용하면 편리합니다. 이 옵션값을 `true`로 설정하면 다단계의 서브 패스 디렉터를 한 번에 생성할 수 있습니다.

다음은 `recursive` 옵션을 이용하여 다단계 디렉터를 한 번 생성하는 예제입니다.

예제 파일 | `file_dir-03.php`

```
1  <?php
2      $pathDir = "aa/bb/cc";
3
4      if (!is_dir($pathDir)) {
5          echo "Err] $pathDir 디렉터리가 존재하지 않습니다.<br>";
6
7          echo "새로운 디렉터를 생성합니다.<br>";
8          if (mkdir($pathDir, 0777, true)) {
9              echo "$pathDir 디렉터를 생성했습니다.<br>";
10             } else {
11                 echo "Err] 디렉터리 생성 실패! <br>";
12             }
13
14     } else {
15         echo "OK] $pathDir 디렉터리 작업이 가능합니다.<br>";
```

```

16     }
17
18     ?>

```

화면 출력

Err] aa/bb/cc 디렉터리가 존재하지 않습니다.
 새로운 디렉터를 생성합니다.
 aa/bb/cc 디렉터를 생성했습니다.

06.2.4 파일 목록

디렉터리는 다수의 파일들을 포함하고 있습니다. 또한 여러 개의 디렉터리도 포함을 하고 있습니다. PHP는 디렉터리 안에 있는 파일들의 목록을 쉽게 처리할 수 있는 내장 함수를 지원합니다.

| 내장 함수 |

```

array scandir ( string $directory [, int $sorting_order = SCANDIR_SORT_ASCENDING
[, resource $context ])

```

내장 함수 scandir()은 입력된 디렉터리의 경로에 존재하는 모든 파일명과 디렉터리명을 배열 형태로 반환합니다. 또한 두 번째 매개변수 인자를 통하여 파일의 정렬 순서를 지정할 수 있습니다.

0: 오름차순(기본값)

1: 내림차순

예제 파일 | file_dir-04.php

```

1  <?php
2      $pathDir = "./";
3
4      if (is_dir($pathDir)) {
5          echo "OK] $pathDir 디렉터리 작업이 가능합니다.<br>";
6      }

```

```

7      $dirARR = scandir($pathDir);
8      for ($i=0;$i<count($dirARR);$i++) {
9          echo $dirARR[$i]."<br>";
10     }
11
12 } else {
13     echo "Err] $pathDir 디렉터리가 존재하지 않습니다.<br>";
14 }
15
16 ?>

```

화면 출력

OK] ./ 디렉터리 작업이 가능합니다.

```

.
..
aa
abcd
file_dir-01.php
file_dir-02.php
file_dir-03.php
file_dir-04.php
file_dir-05.php
hello.txt

```

입력한 디렉터리 경로의 파일과 디렉터리 목록을 출력합니다. 출력 시 상위 디렉터리를 의미하는 .와 ..도 출력됩니다. 이는 시스템에서 출력하는 디렉터리 목록 그대로 반환하기 때문입니다.

내장 함수 `dir()`은 `Directory` 클래스의 인스턴스를 통하여 디렉터리의 목록을 출력할 수 있습니다. 그러면 주어진 디렉터리가 열립니다.

| 내장 함수 |

```
Directory dir ( string $directory [, resource $context ] )
```


예제 파일 | **dir.php**

```
1 <?php
2     $d = dir("/home/php7");
3     echo "Handle: " . $d->handle . "\n";
4     echo "Path: " . $d->path . "\n";
5     while (false != ($entry = $d->read())) {
6         echo $entry."<br>";
7     }
8     $d->close();
9
10 ?>
```

화면 출력

```
Handle: Resource id #3 Path: /home/php7 .
..
readme.txt
```

내장 함수 `glob()`는 셸에서 사용되는 규칙과 비슷한 libc `glob()` 함수에 따라서 패턴과 일치하는 모든 경로의 이름을 검색합니다.

| 내장 함수 |

```
array glob ( string $pattern [, int $flags = 0 ] )
```

예제 파일 | **glob.php**

```
1 <?php
2     foreach (glob("*.exe") as $filename) {
3         echo "$filename size " . filesize($filename) . "<br>";
4     }
5
6 ?>
```

화면 출력

```
deplister.exe size 96768
php-cgi.exe size 57856
php-win.exe size 33280
```

php.exe size 101376
phpdbg.exe size 274432

06.2.5 경로

파일 시스템은 다단계의 여러 디렉터리로 이루어져 있습니다. 현재의 디렉터리 또는 다른 곳을 지정하는 상태 디렉터리를 지정할 수 있습니다.

PHP에서 실행되는 파일의 기본 경로는 현재 실행되는 스크립트의 위치입니다. 따라서 다른 경로를 지정하기 위해서는 ./***/**처럼 서브 경로 또는 상위 경로 ../처럼 상대 경로를 지정하면 됩니다.

PHP는 실행 파일 및 경로를 확인할 수 있는 몇 가지 내장 함수를 제공하고 있습니다.

| 내장 함수 | 현재 경로 확인

```
string getcwd ( void )
```

내장 함수 `getcwd()`는 현재의 스크립트가 동작하고 있는 현재의 경로를 반환합니다.

| 내장 함수 | 디렉터리 패스 경로

```
string dirname ( string $path [, int $levels = 1 ] )
```

내장 함수 `dirname()`은 입력한 경로명에서 현재 자신의 디렉터리를 제외한 상위의 경로를 출력합니다.

예제 파일 | `file_dir-05.php`

```
1  <?php
2
3      // 현재의 디렉터리를 확인합니다.
4      $path = getcwd();
5      echo $path."<br>";
```

```

6     echo "== 상위 디렉터리 패스 == ".dirname($path)."<br>";
7     echo "<br>";
8
9     $pathDir = "aa";
10    echo "절대 경로 ==> ".realpath("./".$pathDir)."<br><br>";
11
12    if (!is_dir($pathDir)) {
13        echo "Err] $pathDir 디렉터리가 존재하지 않습니다.<br>";
14    } else {
15        echo "OK] $pathDir 디렉터리 작업이 가능합니다.<br>";
16
17        if (chdir($pathDir)) {
18            echo "작업 디렉터를 $pathDir로 변경합니다.<br>";
19        } else {
20            echo "Err] 작업 디렉터리 변경 실패<br>";
21        }
22    }
23
24    // 현재의 디렉터를 확인합니다.
25    $path = getcwd();
26    echo $path."<br>";
27    echo "== 상위 디렉터리 패스 == ".dirname($path)."<br>";
28    echo "<br>";
29
30    ?>

```

화면 출력

C:\php-7.1.4-Win32-VC14-x86\03

== 상위 디렉터리 패스 == C:\php-7.1.4-Win32-VC14-x86

절대 경로 ==> C:\php-7.1.4-Win32-VC14-x86\03\aa

OK] aa 디렉터리 작업이 가능합니다.

작업 디렉터를 aa로 변경합니다.

C:\php-7.1.4-Win32-VC14-x86\03\aa

== 상위 디렉터리 패스 == C:\php-7.1.4-Win32-VC14-x86\03

06.2.6 리얼 패스

내장 함수 `realpath()` 함수는 입력한 경로명에 대해서 시스템의 절대 경로를 반환합니다. 입력되는 경로명은 상대 경로명을 사용할 수 있습니다. 만일, 상대 경로명을 입력하면 입력한 상대 경로의 절대 경로 값을 출력합니다.

| 내장 함수 | 절대 경로

```
string realpath ( string $path )
```

내장 함수 `realpath_cache_get()`은 `realpath` 캐시 엔트리 항목을 배열로 반환합니다. 확인된 경로, 만료 날짜 및 캐시에 보관 여러 옵션들을 반환합니다.

| 내장 함수 | 캐시 정보

```
realpath_cache_get ( )
```

예제 파일 | `realpath_cache_get.php`

```
1 <?php
2     var_dump(realpath_cache_get());
3
4 ?>
```

화면 출력

```
array(3) { ["C:\php-7.1.4-Win32-VC14-x86\date_get_last_errors.php"]=>
array(8) { ["key"]=> float(2823479507) ["is_dir"]=> bool(false) ["realpath"]=>
string(52) "C:\php-7.1.4-Win32-VC14-x86\date_get_last_errors.php"
["expires"]=> int(1502349080) ["is_rvalid"]=> bool(false) ["is_wvalid"]=>
bool(false) ["is_readable"]=> bool(false) ["is_writable"]=> bool(false) }
["C:\php-7.1.4-Win32-VC14-x86\realpath_cache_get.php"]=> array(8) { ["key"]=>
float(4089114914) ["is_dir"]=> bool(false) ["realpath"]=> string(50) "C:\php-
7.1.4-Win32-VC14-x86\realpath_cache_get.php" ["expires"]=> int(1502349085)
["is_rvalid"]=> bool(false) ["is_wvalid"]=> bool(false) ["is_readable"]=>
bool(false) ["is_writable"]=> bool(false) } ["C:\php-7.1.4-Win32-VC14-x86"]=>
array(8) { ["key"]=> float(3046037941) ["is_dir"]=> bool(true) ["realpath"]=>
```

```
string(27) "C:\php-7.1.4-Win32-VC14-x86" ["expires"]=> int(1502349080)
["is_rvalid"]=> bool(false) ["is_wvalid"]=> bool(false) ["is_readable"]=>
bool(false) ["is_writable"]=> bool(false) } }
```

내장 함수 `realpath_cache_size()`는 `realpath` 캐시가 사용하는 메모리 양을 리턴합니다.

| 내장 함수 | 캐시 사이즈

```
int realpath_cache_size ( void )
```

예제 파일 | `realpath_cache_size.php`

```
1  <?php
2      var_dump(realpath_cache_size());
3
4  ?>
```

화면 출력

```
int(328)
```

06.2.7 디렉터리 삭제

PHP코드를 통하여 디렉터리를 삭제할 수 있습니다. 디렉터리를 삭제하기 위해서는 시스템의 디렉터리 쓰기 권한이 있어야 합니다. 또한 생성한 디렉터리를 삭제하기 위해서는 안에 존재하고 있는 파일이 없어야 합니다. 만일 파일을 포함하고 있는 폴더를 삭제하고자 할 경우에는 `E_WARNING` 오류가 발생합니다.

PHP에서는 기존 디렉터리 삭제를 위한 다음과 같은 내장 함수를 지원합니다.

| 내장 함수 |

```
bool rmdir ( string $dirname )
```

내장 함수 `rmdir()`은 디렉터리를 삭제합니다. 보통 리눅스 계열의 운영체제에서 디렉터

리 삭제 시 `rmdir` 명령을 사용하는 것과 유사한 함수 이름을 사용하고 있습니다.

예제 파일 | `file_dir-06.php`

```
1  <?php
2      $pathDir = "aaa";
3
4      if (is_dir($pathDir)) {
5          echo "OK] $pathDir 디렉터리 작업이 가능합니다.<br>";
6
7          echo "새로운 디렉터리를 삭제합니다.<br>";
8          if (rmdir($pathDir)) {
9              echo "$pathDir 디렉터리를 삭제했습니다.<br>";
10         } else {
11             echo "Err] 디렉터리 삭제 실패! <br>";
12         }
13
14     } else {
15         echo "Err] $pathDir 디렉터리가 존재하지 않습니다.<br>";
16     }
17
18  ?>
```

화면 출력

OK] aaa 디렉터리 작업이 가능합니다.

새로운 디렉터리를 삭제합니다.

aaa 디렉터리를 삭제했습니다.

06.2.8 디렉터리 이름 변경

PHP 코드를 통하여 디렉터리의 이름을 변경할 수 있습니다. 디렉터리의 이름을 변경하기 위해서는 쓰기 권한이 필요합니다.

| 내장 함수 |

bool **rename** (string \$oldname , string \$newname [, resource \$context])

내장 함수 `rename()`은 디렉터리 이름을 변경합니다. 입력 매개변수로는 2개의 값을 전달받습니다. 바꾸기 전의 `oldname` 이름과 새롭게 변경되는 `newname` 이름입니다.

내장 함수 `rename()`은 새로운 이름으로 디렉터리의 이름을 바꾸고 필요할 경우 디렉터리 간 이동합니다.

예제 파일 | `rename.php`

```
1  <?php
2      rename("aaa", "bbb");
3
4  ?>
```

06.2.9 경로 변경

내장 함수 `chdir()`은 현재 실행되고 있는 스크립트 경로에서 다른 상태 디렉터리를 변경합니다. 이는 현재의 실행되는 스크립트의 경로를 내부적으로 변경합니다.

| 내장 함수 | `경로 변경`

```
bool chdir ( string $directory )
```

내장 함수 `chroot()`는 현재의 디렉터리에서 입력한 루트 디렉터리로 변경합니다.

| 내장 함수 |

```
bool chroot ( string $directory )
```

윈도우 환경에서는 이 함수를 사용할 수 없습니다. 리눅스 계열의 운영체제에서만 사용 가능합니다. 이 함수를 사용하기 위해서는 루트 권한을 필요로 합니다.

예제 파일 | **chroot.php**

```
1 <?php
2     chroot("/path/to/your/chroot/");
3     echo getcwd();
4
5 ?>
```

06.2.10 디렉터리 접근

디렉터리 접근 및 처리를 위해서 좀 더 세분화된 기능의 PHP 내장 함수들을 지원합니다.

내장 함수 `opendir()`은 directory handle을 오픈, 생성합니다.

| 내장 함수 |

```
resource opendir ( string $path [, resource $context ] )
```

디렉터리를 오픈 후에는 `closir()`, `readdir()` 및 `rewinddir()`과 같은 함수로 접근할 수 있습니다.

`dir_handle`이 가리키는 디렉터리 스트림을 닫습니다. 스트림은 `opendir()`에 의해 열려 있는 스트림이어야 합니다.

| 내장 함수 |

```
void closedir ([ resource $dir_handle ] )
```

예제 파일 | **opendir.php**

```
1 <?php
2     $dir = "/home/php7/";
3
4     // 0 디렉터리를 열고 내용을 읽습니다.
```



```

5     if (is_dir($dir)) {
6         if ($dh = opendir($dir)) {
7             while (($file = readdir($dh)) !== false) {
8                 echo "파일명: $file : 파일타입: " . filetype($dir . $file)
9                     . "<br>";
10            }
11            closedir($dh);
12        } else {
13            echo $dir . " 디렉터리가 존재하지 않습니다.";
14        }
15    }
16    ?>

```

화면 출력

파일명: . : 파일타입: dir
 파일명: .. : 파일타입: dir
 파일명: readme.txt : 파일타입: file

예제 파일 | **closedir.php**

```

1  <?php
2      $dir = "/home/php7/";
3
4      // 디렉터리를 열고 디렉터리를 변수로 읽어 들인 후에 닫습니다.
5      if (is_dir($dir)) {
6          if ($dh = opendir($dir)) {
7              $directory = readdir($dh);
8              closedir($dh);
9          }
10     }
11     ?>

```

내장 함수 `readdir()`은 디렉터리 핸들에서 항목을 읽어옵니다. 파일 시스템이 저장한 순서대로 항목의 이름을 반환합니다.

| 내장 함수 |

```
string readdir ([ resource $dir_handle ] )
```

예제 파일 | **readdir.php**

```
1  <?php
2
3      if ($handle = opendir('/home/php7')) {
4          echo "Directory handle: $handle <br>";
5          echo "Entries:<br>";
6
7          echo "디렉터리를 반복하는 올바른 방법입니다. <br>";
8          while (false !== ($entry = readdir($handle))) {
9              echo "$entry <br>";
10         }
11
12         echo "디렉터리를 반복하는 잘못된 방법입니다. <br>";
13         while ($entry = readdir($handle)) {
14             echo "$entry <br>";
15         }
16
17         closedir($handle);
18     }
19
20  ?>
```

화면 출력

Directory handle: Resource id #3

Entries:

디렉터리를 반복하는 올바른 방법입니다.

.

..

readme.txt

디렉터리를 반복하는 잘못된 방법입니다.

내장 함수 `rewinddir()`은 디렉터리 핸들을 처음으로 다시 초기화합니다.

| 내장 함수 |

```
void rewinddir ([ resource $dir_handle ] )
```

예제 파일 | **rewinddir.php**

```
1  <?php
2
3      if ($handle = opendir('/home/php7')) {
4          echo "Directory handle: $handle <br>";
5          echo "Entries:<br>";
6
7          echo "디렉터리를 반복하는 올바른 방법입니다. <br>";
8          while (false !== ($entry = readdir($handle))) {
9              echo "$entry <br>";
10         }
11
12         // 처음으로 다시 초기화
13         rewinddir($handle);
14
15         echo "디렉터리를 반복하는 잘못된 방법입니다. <br>";
16         while ($entry = readdir($handle)) {
17             echo "$entry <br>";
18         }
19
20         closedir($handle);
21     }
22  ?>
```

화면 출력

Directory handle: Resource id #3

Entries:

디렉터리를 반복하는 올바른 방법입니다.

.
..

readme.txt

디렉터리를 반복하는 잘못된 방법입니다.

.
..

readme.txt

06.2.11 디렉터리 용량 표시

파일 작업을 하기 위해서 디스크의 저장 용량은 매우 중요합니다. 서버의 용량이 매우 커서 신경 쓸 필요가 없는 경우는 없을 것입니다. 만일 디스크의 용량이 부족하다면 정상적인 파일 처리 및 작업을 할 수 없을 것입니다.

PHP에서는 디렉터리의 용량을 확인할 수 있는 다음과 같은 내장 함수를 지원합니다.

| 내장 함수 | 전체 공간

```
float disk_total_space ( string $directory )
```

내장 함수 `disk_total_space()`는 현재 디스크의 전체 용량을 확인할 수 있습니다.

| 내장 함수 | 여유 공간

```
float disk_free_space ( string $directory )
```

내장 함수 `disk_free_space()`는 현재 작업하는 디렉터리의 남아 있는 여분의 용량을 확인할 수 있습니다.

예제 파일 | `diskspace.php`

```
1  <?php
2      $pathDir = "./";
3
4      if (is_dir($pathDir)) {
5          echo "OK] $pathDir 디렉터리 작업이 가능합니다.<br>";
6
7          $totalSpace = disk_total_space($pathDir);
8          $freeSpace = disk_free_space($pathDir);
9          echo "여유 공간: ".$freeSpace." / 전체 공간: $totalSpace <br>";
10
11     } else {
12         echo "Err] $pathDir 디렉터리가 존재하지 않습니다.<br>";
13     }
```

```
14
15 ?>
```

화면 출력

OK] ./ 디렉터리 작업이 가능합니다.
여유 공간: 48003436544 / 전체 공간: 239147675648

06.3 권한 설정

요즘들어 대부분의 운영체제는 멀티 사용자 환경을 지원합니다. 따라서 파일 시스템은 각각 사용자와 관련된 파일, 디렉터리 접근을 처리할 수 있도록 권한을 부여합니다.

이러한 권한은 파일 시스템을 효과적으로 운영, 관리하는 데 매우 유용합니다. PHP에서는 슈퍼 유저를 대상으로 몇 가지 권한 관련 내장 함수를 제공하고 있습니다.

06.3.1 소유권 변경

멀티 유저 환경의 파일 시스템은 각각의 파일에 대해서 소유권 설정되어 있습니다. 즉, 해당 파일을 생성하고 관리하는 사용자를 말합니다.

내장 함수 `chown()`은 파일의 소유권을 변경합니다. 소유권을 변경하기 위해서는 PHP 실행이 슈퍼 유저로 실행되어야 합니다.

| 내장 함수 |

```
bool chown ( string $filename , mixed $user )
```

예제 파일 | `chown.php`

```
1 <?php
2
3 // 파일명과 사용자
4 $file_name= "test.php";
```

```

5    $path = "/home/php7/" . $file_name ;
6    $user_name = "root";
7
8    // 소유권을 변경합니다.
9    chown($path, $user_name);
10
11   // 결과를 확인합니다.
12   $stat = stat($path);
13   print_r(posix_getpwuid($stat['uid']));
14
15   ?>

```

소유권을 변경하는 것과 달리 파일의 소유권을 읽어올 수도 있습니다. 내장 함수 `fileowner()`는 파일의 소유자 정보를 읽어옵니다.

| 내장 함수 |

```
int fileowner ( string $filename )
```

예제 파일 | [fileowner.php](#)

```

1  <?php
2      $filename = 'fileowner.php';
3      print_r(posix_getpwuid(fileowner($filename)));
4
5      ?>

```

화면 출력

```
Array ( [name] => jinyshop [passwd] => x [uid] => 519 [gid] => 520 [gecos] =>
[dir] => /home/jinyshop [shell] => /bin/bash )
```

06.3.2 모드 변경

파일 시스템은 소유권 이외에도 파일의 접근 권한을 가지고 있습니다. 해당 파일을 읽기만 가능하게 할 것인지, 아니면 읽기/쓰기 모두를 허용할 것인지를 설정할 수 있습니다.

또한 파일의 실행 권한도 부여할 수 있습니다.

내장 함수 `chmod()`는 파일의 모드를 변경할 수 있습니다. 모드를 변경한 후에는 성공 여부를 논리값으로 반환합니다.

| 내장 함수 |

```
bool chmod ( string $filename , int $mode )
```

예제 파일 | **chmod.php**

```
1  <?php
2
3      if ( chmod("test.txt",0755)) {
4          echo "모드 변경";
5      } else {
6          echo "변경 실패";
7      }
8
9  ?>
```

내장 함수 `umask()`는 현재의 `umask`를 변경합니다.

| 내장 함수 |

```
int umask ([ int $mask ] )
```

예제 파일 | **umask.php**

```
1  <?php
2      $old = umask(0);
3      chmod("/path/some_dir/some_file.txt", 0755);
4      umask($old);
5
6      // Checking
7      if ($old != umask()) {
```

```

8         die('An error occurred while changing back the umask');
9     }
10
11     ?>

```

파일의 퍼미션을 권한을 읽어 올 수 있습니다. 내장 함수 `fileperms()`는 파일의 권한 값을 정수값 형태로 반환합니다. 반환되는 정수값은 우리가 알기 쉬운 형태의 코드로 확인을 하기 위해서는 별도의 처리가 필요로 합니다.

| 내장 함수 |

```
int fileperms ( string $filename )
```

예제 파일 | [fileperms.php](#)

```

1  <?php
2      $permissions = fileperms("fileperms.php");
3      echo "permissions = " . $permissions . "<br>";
4
5      // Owner
6      $info .= (($permissions & 0x0100) ? 'r' : '-');
7      $info .= (($permissions & 0x0080) ? 'w' : '-');
8      $info .= (($permissions & 0x0040) ?
9          (($permissions & 0x0800) ? 's' : 'x' ) :
10         (($permissions & 0x0800) ? 'S' : '-'));
11
12     // Group
13     $info .= (($permissions & 0x0020) ? 'r' : '-');
14     $info .= (($permissions & 0x0010) ? 'w' : '-');
15     $info .= (($permissions & 0x0008) ?
16         (($permissions & 0x0400) ? 's' : 'x' ) :
17         (($permissions & 0x0400) ? 'S' : '-'));
18
19     // World
20     $info .= (($permissions & 0x0004) ? 'r' : '-');
21     $info .= (($permissions & 0x0002) ? 'w' : '-');
22     $info .= (($permissions & 0x0001) ?

```



```

23         (($permissions & 0x0200) ? 't' : 'x' ) :
24         (($permissions & 0x0200) ? 'T' : '-'));
25
26     echo $info;
27
28     ?>

```

화면 출력

```

permissions = 33188
rw-r--r--

```

파일 권한 중에는 실행 가능한 권한 모드가 있습니다. 입력한 경로의 파일이 실행 가능한 상태 권한인지를 확인할 수 있는 내장 함수 `is_executable()`을 제공합니다.

| 내장 함수 |

```
bool is_executable ( string $filename )
```

파일의 실행 가능 여부를 논리값 형태로 반환합니다.

예제 파일 | `is_executable.php`

```

1  <?php
2
3      $file = 'check.sh';
4
5      if (is_executable($file)) {
6          echo $file.' is executable';
7      } else {
8          echo $file.' is not executable';
9      }
10
11  ?>

```

06.3.3 그룹 변경

리눅스와 같이 멀티 사용자를 지원하는 운영체제, 파일 시스템의 경우 그룹이라는 권한 설정이 있습니다. 그룹은 여러 사용자를 하나의 그룹으로 묶어서 권한을 관리하는 개념입니다.

PHP 코드를 통하여 파일의 그룹을 변경할 수 있습니다. 파일 그룹을 변경하기 위해서는 root 권한이 필요합니다.

보통 웹 페이지는 nobody 그룹을 많이 사용하는데 권한에 따라서 chgrp() 함수를 통하여 파일의 그룹을 변경할 수 있습니다.

| 내장 함수 |

```
bool chgrp ( string $filename , mixed $group )
```

예제 파일 | **chgrp.php**

```
1  <?php
2      $filename = 'install.txt';
3      $format = "%s's Group ID @ %s: %d <br>";
4      printf($format, $filename, date('r'), filegroup($filename));
5
6      // 100
7      chgrp($filename, 8);
8
9      // filegroup () 결과를 캐시하지 않습니다.
10     clearstatcache();
11
12     printf($format, $filename, date('r'), filegroup($filename));
13
14  ?>
```

설정된 파일 그룹 정보를 가져와서 확인할 수 있습니다. 내장 함수 filegroup()는 파일이 속해 있는 그룹의 ID 숫자값을 반환합니다.

| 내장 함수 |

```
int filegroup ( string $filename )
```

예제 파일 | filegroup.php

```
1  <?php
2  $filename = 'index.php';
3  $filegroup_Id = filegroup($filename);
4  echo "filegroup ID = " . $filegroup_Id . "<br>";
5
6  print_r(posix_getgrgid($filegroup_Id));
7
8  ?>
```

화면 출력

```
filegroup ID = 520
```

```
Array ( [name] => jinyshop [passwd] => x [members] => Array ( ) [gid] => 520 )
```

06.4 파일 정보

파일은 디스크 공간에 존재하는 데이터입니다. PHP 코드상에서 새로운 파일을 생성할 수 있고, 또한 존재하는 파일을 읽어서 처리할 수도 있습니다.

생성된 파일을 좀 더 안전하게 접근 처리하기 위해서는 파일의 상태 정보를 잘 확인하는 것이 중요합니다. PHP는 파일의 상태를 확인할 수 있는 다양한 내장 함수를 제공하고 있습니다.

06.4.1 파일 확인

안전한 파일 처리를 위해서는 유효한 파일 경로를 접근하여 처리해야 합니다. 입력한 파일 경로가 존재하지 않은 파일이거나, 파일이 아닌 디렉터리일 경우에는 오류가 발생합니다.

잘못된 파일 경로로 인하여 발생할 수 있는 오류를 줄이기 위해서는 먼저 파일의 존재 여부를 확인하는 것이 중요합니다. PHP에서는 기존 디렉터리 삭제에 위한 다음과 같은 내장 함수를 지원합니다.

| 내장 함수 |

```
bool file_exists ( string $filename )
```

내장 함수 `file_exists()`는 파일의 존재 여부를 확인하여 논리값으로 결과를 반환합니다.

예제 파일 | `file_exists.php`

```
1  <?php
2      $filename = "./readme.txt";
3
4      if (file_exists($filename)) {
5          echo $filename." 파일명이 존재합니다.<br>";
6      } else {
7          echo "파일을 찾을 수 없습니다.<br>";
8      }
9
10 ?>
```

화면 출력

`./readme.txt` 파일명이 존재합니다.

입력한 경로의 파일이 존재한다면 다시 한번 입력한 경로 파일이 실제 파일인지를 확인하는 것이 좋습니다. 내장 함수 `is_file()`은 입력한 경로가 파일인지를 확인하여 논리값으로 반환합니다.

| 내장 함수 |

```
bool is_file ( string $filename )
```

예제 파일 | **is_file.php**

```
1 <?php
2     var_dump(is_file('is_file.php'));
3
4 ?>
```

화면 출력

bool(true)

지정한 경로의 파일에 대한 타입을 확인할 수 있습니다. 내장 함수 `filetype()`은 입력한 경로에 대해서 파일 타입인지, 디렉터리 타입인지를 분별하여 문자열을 반환합니다.

| 내장 함수 |

```
string filetype ( string $filename )
```

예제 파일 | **filetype.php**

```
1 <?php
2     echo filetype('/etc/passwd') . "<br>";
3     // file
4     echo filetype('/etc/') . "<br>";
5     // dir
6
7 ?>
```

화면 출력

file
dir

06.4.2 날짜와 시간

모든 파일들은 자신이 생성된 날짜와 최근 수정된 시간 정보를 가지고 있습니다. 이러한 파일들의 날짜와 시간 정보는 최신 파일로 업데이트를 유지하기 위한 정보로 매우 유용합니다.

파일들의 날짜와 시간 정보를 PHP 코드를 통하여 확인할 수 있습니다. PHP에서는 파일 날짜 관련 다음과 같은 내장 함수를 지원합니다.

| 내장 함수 | 접근 일자

```
int filetime ( string $filename )
```

내장 함수 `filetime()`은 해당 파일의 마지막 사용된 시간을 반환합니다.

| 내장 함수 | 수정 일자

```
int filetime ( string $filename )
```

내장 함수 `filetime()`은 해당 파일의 마지막 수정된 시간을 반환합니다.

예제 파일 | **fileinfo-01.php**

```
1  <?php
2
3  $filename = './readme.txt';
4  if (file_exists($filename)) {
5      echo "$filename 마지막 접근 일자: " . date("F d Y H:i:s.",
6          filetime($filename));
7      echo "<br>";
8      echo "$filename 마지막 수정 일자: " . date ("F d Y H:i:s.",
9          filetime($filename));
10 }
11 ?>
```

화면 출력

```
./readme.txt 마지막 접근 일자: June 13 2017 06:19:03.
./readme.txt 마지막 수정 일자: June 13 2017 06:19:03.
```

내장 함수 `filetime()`은 파일의 inode 변경 시간을 읽어옵니다. 시간은 Unix 타임 스탬프로 반환됩니다.

| 내장 함수 |

```
int filetime ( string $filename )
```

예제 파일 | **filetime.php**

```
1  <?php
2  $filename = 'install.txt';
3  if (file_exists($filename)) {
4      echo "$filename 의 변경 시간 : " . date("F d Y H:i:s.",
5          filetime($filename));
6  }
7  ?>
```

내장 함수 touch()는 파일의 수정 시간을 재설정할 수 있습니다.

| 내장 함수 |

```
bool touch ( string $filename [, int $time = time() [, int $atime ]] )
```

예제 파일 | **touch.php**

```
1  <?php
2      // 터치 시간입니다
3      // 과거 한 시간으로 설정.
4      $time = time() - 3600;
5
6      // 파일의 수정 시간을 재설정합니다.
7      if (!touch('touch.php', $time)) {
8          echo 'wrong...';
9      } else {
10         echo 'success';
11     }
12
13  ?>
```

화면 출력

success

내장 함수 stat()은 파일에 대한 정보를 반환합니다.

| 내장 함수 |

```
array stat ( string $filename )
```

예제 파일 | stat.php

```
1  <?php
2      /* Get file stat */
3      $stat = stat('C:\php\php.exe');
4
5      /*
6       * Print file access time, this is the same
7       * as calling fileatime()
8       */
9      echo 'Access time: ' . $stat['atime'];
10
11     /*
12      * Print file modification time, this is the
13      * same as calling filemtime()
14      */
15     echo 'Modification time: ' . $stat['mtime'];
16
17     /* Print the device number */
18     echo 'Device number: ' . $stat['dev'];
19
20  ?>
```

06.4.3 파일 종류

PHP에서 파일을 처리할 때 경로, 파일명, 확장자 등의 정보를 path 값으로 처리하여 전달합니다. 입력되는 경로 값을 기준으로 파일명, 확장자, 경로 등의 문자열을 분리하여 처

리할 수 있는 유용한 내장 함수들을 제공합니다.

| 내장 함수 |

```
string basename ( string $path [, string $suffix ] )
```

내장 함수 `basename()`은 입력된 경로명에서 파일명 부분만을 추출하여 분리합니다.

예제 파일 | **basename.php**

```
1 <?php
2     $path = "../aa/bb/cc/eee/fff/thisfile.php";
3     echo basename($path);
4
5 ?>
```

화면 출력

thisfile.php

06.4.4 파일 경로 및 확장자

파일 처리 시 전달되는 매개변수 파일명은 상대 경로 및 절대 경로에 대한 디렉터리 구조와 파일명, 확장자를 가지고 있습니다.

```
mixed pathinfo ( string $path [, int $options = PATHINFO_DIRNAME | PATHINFO_BASENAME | PATHINFO_EXTENSION | PATHINFO_FILENAME ] )
```

`pathinfo()` 함수는 파일명의 경로를 포함한 정보를 배열로 반환합니다. 반환된 배열값의 키로 `dirname`, `basename`, `extension`, `filename`을 통하여 경로 값을 분석할 수 있습니다.

예제 파일 | `pathinfo.php`

```
1 <?php
2     $path_parts = pathinfo('./readme.txt');
3
4     echo $path_parts['dirname'], "<br>";
5     echo $path_parts['basename'], "<br>";
6     echo $path_parts['extension'], "<br>";
7     echo $path_parts['filename'], "<br>"; // since PHP 5.2.0
8
9 ?>
```

화면 출력

```
readme
.txt
txt
readme
```

06.4.5 링크

리눅스, 맥OS와 같은 운영체제의 파일 시스템은 링크라는 파일 처리 개념이 있습니다. 링크는 물리적으로 파일이 존재하는 것이 아니라 실제 파일과 연결된 형태의 연결고리입니다.

입력한 경로의 파일이 심볼 링크 형태의 파일인지를 확인합니다. 내장 함수 `is_link()`는 입력한 경로가 링크 형태의 연결인지를 판별해서 논리값으로 출력합니다.

| 내장 함수 |

```
bool is_link ( string $filename )
```

예제 파일 | `is_link.php`

```
1 <?php
2     $link = 'symlink';
3
4     if (is_link($link)) {
```

```

5         echo $link . " is symbolic link";
6     } else {
7         echo $link . " is not symbolic link";
8     }
9
10    ?>

```

화면 출력

symlink is symbolic link

파일에 대해서 새로운 심볼 링크를 생성합니다 내장 함수 `symlink()`는 2개의 인자값을 전달받아 심볼 링크를 생성합니다.

| 내장 함수 |

```
bool symlink ( string $target , string $link )
```

예제 파일 | `symlink.php`

```

1  <?php
2      $target = 'symlink.php';
3      $link = 'symlink';
4      symlink($target, $link);
5      echo readlink($link);
6
7  ?>

```

화면 출력

symlink.php

내장 함수 `readlink()`는 심볼릭 링크의 대상을 반환합니다.

| 내장 함수 |

```
string readlink ( string $path )
```

하드 링크 형태로 파일을 생성합니다. 내장 함수 `link()`는 2개의 인자값을 받아 하드 링크를 생성합니다.

| 내장 함수 |

```
bool link ( string $target , string $link )
```

예제 파일 | `link.php`

```
1  <?php
2      $target = 'link.php';
3      $link = 'hardlink.php';
4
5      link($target, $link);
6
7  ?>
```

내장 함수 `linkinfo()`는 하드 링크 파일의 정보를 반환합니다.

| 내장 함수 |

```
int linkinfo ( string $path )
```

내장 함수 `lchgrp()`는 심볼 링크의 그룹 소유권을 변경합니다.

| 내장 함수 |

```
bool lchgrp ( string $filename , mixed $group )
```

예제 파일 | `lchgrp.php`

```
1  <?php
2      $target = 'output.php';
3      $link = 'output.html';
```

```

4     symlink($target, $link);
5
6     lchgrp($link, 8);
7
8     ?>

```

내장 함수 `lchown()`은 심볼 링크의 사용자 소유권을 변경합니다.

| 내장 함수 |

```
bool lchown ( string $filename , mixed $user )
```

예제 파일 | **lchown.php**

```

1  <?php
2      $target = 'output.php';
3      $link = 'output.html';
4      symlink($target, $link);
5
6      lchown($link, 8);
7      ?>

```

내장 함수 `lstat()`은 파일 또는 심볼릭 링크에 대한 정보를 반환합니다.

| 내장 함수 |

```
array lstat ( string $filename )
```

예제 파일 | **lstat.php**

```

1  <?php
2      symlink('uploads.php', 'uploads');
3
4      // Contrast information for uploads.php and uploads
5      array_diff(stat('uploads'), lstat('uploads'));
6      ?>

```

06.4.6 그 외 처리

리눅스와 같이 유닉스 계열 타입의 파일 시스템은 파일에 대한 inode 값을 가지고 있습니다. 내장 함수 `fileinode()`는 입력한 경로의 inode 값을 반환합니다.

| 내장 함수 |

```
int fileinode ( string $filename )
```

예제 파일 | `fileinode.php`

```
1  <?php
2      $filename = 'fileinode.php';
3      echo fileinode($filename). "<br>";
4
5  ?>
```

화면 출력

6961340

파일 상태 캐시를 지웁니다. `clearstatcache()` 함수를 사용하여 PHP가 파일에 대해 캐시하는 정보를 지울 수 있습니다.

| 내장 함수 |

```
void clearstatcache ([ bool $clear_realpath_cache = false [, string $filename ] ] )
```

파일 이름이나 문자열을 지정된 패턴과 비교합니다. `fnmatch()`는 전달된 문자열이 주어진 셸 와일드 카드 패턴과 일치하는지 확인합니다.

| 내장 함수 |

```
bool fnmatch ( string $pattern , string $string [, int $flags = 0 ] )
```

예제 파일 | **fnmatch.php**

```
1 <?php
2     $filename = "gray";
3     if (fnmatch("*gr[ae]y", $filename)) {
4         echo "some form of gray ...";
5     } else {
6         echo "not match";
7     }
8
9 ?>
```

화면 출력

some form of gray ...

06.5 파일 열기

PHP는 다양한 방법으로 파일을 읽고, 쓰는 처리가 가능합니다. 파일의 포인터를 통하여 처리하는 방식은 C 언어와 같이 오래 전부터 사용하던 형태의 파일 처리 방법입니다. PHP도 파일 포인터를 오픈하여 파일을 처리할 수 있습니다.

PHP에서 파일을 제어하기 위해서 먼저 파일 포인터를 생성해야 합니다. 파일 포인터는 파일을 접근하기 위해서 오픈하는 것입니다. 단지 파일의 시작되는 포인터를 얻는 형태의 작업입니다.

PHP는 파일의 시작 포인트를 관리할 수 있는 다음과 같은 내장 함수를 제공합니다.

| 내장 함수 |

```
resource fopen ( string $filename , string $mode [, bool $use_include_path = false [, resource $context ] ] )
```

내장 함수 `fopen()`은 파일을 접근하고 사용하기 위한 포인트를 반환합니다. 매개변수 인자로 통상 2개의 값을 입력하는데 첫 번째는 경로를 포함한 파일명이고, 두 번째는 파일

의 접근 모드입니다.

06.5.1 파일 모드

모드란 파일을 읽기용으로 오픈할 것인지, 쓰기용으로 오픈할 것인지 등을 정의하는 값입니다. PHP의 파일 모드 상태는 10가지 방식으로 구분됩니다.

- r: 읽기
- r+: 읽기/쓰기
- w: 쓰기
- w+: 쓰기/읽기
- a: 쓰기
- a+: 쓰기/읽기
- x: 쓰기
- x+: 쓰기/읽기
- c: 쓰기
- c+: 쓰기/읽기

크게 기본 모드와 기본 모드에서 + 기호가 들어가 있는 상태로 구분할 수 있습니다.

r 모드는 단순히 파일을 읽기만 가능하도록 오픈하는 것입니다. 파일의 첫 번째 위치 값을 반환합니다.

w 모드는 파일을 쓰기 전용으로 오픈하는 것입니다. 파일의 포인터는 첫 번째의 위치 값을 반환합니다. w 모드는 새로운 파일을 생성합니다. 만일 기존에 동일한 이름의 파일이 존재할 경우 모든 내용은 덮어쓰기로 지워집니다.

a 모드는 기존에 있는 파일 뒤에 새로운 내용을 추가하여 쓸 수 있습니다. 파일의 위치 값은 맨 마지막을 가르키게 됩니다. 만일 파일이 없는 경우에는 새로운 파일을 생성합니다.

x 모드는 새로운 파일을 생성하여 쓰기로 오픈합니다. 만일, 기존에 동일한 이름의 파일이 있다면 실패로 FALSE를 반환합니다.

r+ 모드는 읽기 모드에 쓰기 모드를 추가합니다. 파일의 포인터는 처음 시작 위치 값을 반환합니다.

w+ 모드는 읽기와 쓰기를 지원합니다. 기존 파일의 내용을 모두 삭제하고 반환되는 시작 포인터는 처음을 가르킵니다. 만일 파일이 존재하지 않으면 새로운 파일을 생성합니다.

a+ 모드는 기존 모드에 추가로 쓰기 기능과 읽기를 지원합니다. 파일의 반환 포인터는 맨 마지막을 가르킵니다. 만일 파일이 존재하지 않으면 새로운 파일을 생성합니다.

x+ 모드는 파일을 읽기와 쓰기를 지원합니다. 만일 기존에 파일이 존재하면 FALSE를 반환합니다.

06.5.2 파일 닫기

내장 함수 `fopen()`으로 오픈한 파일 포인터는 리소스 관리를 위해서 사용 후에 반드시 닫아야 합니다. `fclose()` 함수 매개변수로 `fopen()` 함수로 받은 리소스 포인터를 전달하면 됩니다.

| 내장 함수 |

```
bool fclose ( resource $handle )
```

예제 파일 | **fclose.php**

```
1  <?php
2
3      $filename = './readme.txt';
4      if (file_exists($filename)) {
5          if ($fp = fopen($filename, "r")) {
6              echo "파일 포인터를 읽기 전용으로 오픈합니다.<br>";
7              fclose($fp);
8          } else {
9              echo "파일을 읽을 수 없습니다.<br>";
10         }
11     } else {
```

```

12     echo "파일이 존재하지 않습니다.";
13 }
14
15 ?>

```

화면 출력

파일 포인터를 읽기 전용으로 엡힙니다.

06.6 파일 데이터 읽기

PHP 코드를 통하여 파일을 읽어 처리해 보겠습니다. 이전에 학습한 내장 함수 `fopen()` 을 통하여 파일 포인터를 엡힙니다. 이렇게 엡힌 파일 포인터를 통하여 파일에 접근하여 내용을 읽어올 수 있습니다. 접근된 파일에 데이터를 읽어올 수 있는 방법은 여러 가지 형태가 있습니다. 여기서는 데이터 처리를 세 가지 방법으로 구분하여 설명하겠습니다.

06.6.1 파일 끝

데이터를 가지고 있는 파일은 크기는 매우 다양합니다. 간단히 몇 줄을 포함할 수도 있고, 몇 GB의 엄청난 파일도 있을 것입니다. 접근한 파일의 데이터를 읽기 위해서는 반복된 자료 읽기 작업이 수행됩니다.

자료 읽기 작업을 반복 수행할 때 얼마나 반복을 수행할지 정의해야 합니다. 쉬운 방법으로는 파일 끝을 나타내는 기호를 확인하여 처리하는 것입니다.

파일들은 대부분 파일의 끝을 표시하는 EOF라는 코드값(-1)을 가지고 있습니다. EOF는 End of File의 약자입니다. 파일을 읽을 때 이 EOF 여부를 체크하여 파일의 끝을 확인할 수 있습니다.

PHP는 파일의 끝을 확인할 수 있는 특별한 내장 함수를 제공합니다.

| 내장 함수 |

```
bool feof ( resource $handle )
```

feof() 함수는 EOF 코드를 확인하여 논리값으로 출력합니다. feof() 함수를 사용할 때는 반복문으로 while 구문을 사용합니다.

내장 함수 fpassthru()는 EOF까지 열려 있는 파일을 읽고, 결과를 출력 버퍼에 저장합니다.

| 내장 함수 |

```
int fpassthru ( resource $handle )
```

예제 파일 | fpassthru.php

```
1  <?php
2
3      // open the file in a binary mode
4      $name = './img/ok.png';
5      $fp = fopen($name, 'rb');
6
7      // send the right headers
8      header("Content-Type: image/png");
9      header("Content-Length: " . filesize($name));
10
11     // dump the picture and stop the script
12     fpassthru($fp);
13     exit;
14
15  ?>
```

06.6.2 한 글자 읽기

내장 함수 fopen() 함수로 얻은 파일 포인터에서 실제적인 데이터를 읽어 보도록 하겠습

니다. 데이터를 읽는 방법은 다양하지만, 우선 파일에서 한 글자의 character를 읽어 보겠습니다.

내장 함수 `fgetc()`은 파일 포인터에서 한 글자의 문자를 읽어서 반환합니다. 한 글자를 읽은 후에는 다음 글자를 읽을 수 있도록 파일 포인터를 한 단계씩 이동합니다.

| 내장 함수 |

```
string fgetc ( resource $handle )
```

한 글자씩 읽는 `fgetc()` 함수는 파일을 처리하는 데 매우 유용합니다. 파일의 내용을 분석하고 파싱 처리를 하는 데도 편리합니다. 하지만 파일 전체를 한 글자씩 읽어야 하기 때문에 처리 시간이 오래 걸립니다.

예제 파일 | **fgetc.php**

```
1  <?php
2      $filename = './readme.txt';
3
4      $fp = fopen($filename, "r") or die("파일을 읽을 수 없습니다!");
5
6      // EOF 체크
7      while (!feof($fp)) {
8
9          $i++;
10         echo $i.: ". fgetc($fp)."<br>";
11     }
12     echo "파일 카운트 : $i";
13
14     fclose($fp);
15
16     ?>
```

화면 출력

```
1: h
2: e
3: l
```

```
4: l
5: o
6:
7: j
8: i
9: n
10: y
11: P
12: H
13: P
14: 3
15:
파일 카운트: 15
```

위의 예제는 파일에서 한 글자씩 읽어서 화면에 출력을 하는 예제입니다. 파일의 포인터를 얻어 feof()의 조건을 만족하는 만큼 while 반복문을 실행합니다. 이때 반복문으로 while()문을 이용하는 것은 파일의 크기를 미리 알 수 없기 때문입니다. 계속적으로 반복을 하면서 feof() 함수에서 EOF를 확인하고 true를 반환할 때 while문은 종료됩니다.

06.6.3 한 줄 읽기

내장 함수 fgets()는 파일 포인터에서 한 글자씩 읽어옵니다. 한 글자씩 읽어오는 처리는 실행 속도가 매우 느립니다. 이를 보완하고자 블록 개념으로 파일의 데이터를 읽을 수 있는 fgets() 함수를 제공합니다.

내장 함수 fgets()는 파일에서 한 줄의 데이터를 읽어서 반환합니다. fgets() 함수는 환경 설정 파일, 로그 파일 등을 읽고 처리하는 데 좀 더 유용합니다. 이러한 파일은 정보를 한 줄 단위로 구분하여 저장하기 때문에 한 줄씩 읽는 fgets()는 보다 유용합니다.

| 내장 함수 |

```
string fgets ( resource $handle [, int $length ] )
```

파일에서 한 줄의 끝은 리턴 문자(\n)를 포함합니다. fgets() 함수는 텍스트 파일에서 반

환된 문장(\n)을 확인하여 데이터의 한 줄을 읽어옵니다. 또는 두 번째 인자로 일정 크기의 값을 지정하면 지정된 데이터의 크기만 읽어올 수도 있습니다.

예제 파일 | **fgets.php**

```
1  <?php
2      $filename = './readme2.txt';
3
4      // @ 표시가 앞에 있으면 오류 메시지를 표시하지 않겠다는 의미입니다.
5      $fp = @fopen($filename, "r");
6
7      if ($fp) {
8          while (($buffer = fgets($fp, 4096)) !== false) {
9              echo "라인 ".$i++.".": ".$buffer."<br>";
10         }
11
12         if (!feof($fp)) {
13             echo "Err] fgets() 읽기 실패\n";
14         }
15
16         fclose($fp);
17     }
18
19  ?>
```

화면 출력

라인: 안녕하세요.

라인 1: 여러 줄로 구성된 파일을

라인 2: 읽어올 수 있습니다.

내장 함수 fgets()는 파일에서 HTML 및 PHP 태그가 제거된 행을 반환합니다.

| 내장 함수 |

```
string fgets ( resource $handle [, int $length [, string $allowable_tags ] ] )
```

예제 파일 | **fgetss.php**

```
1  <?php
2  $str = <<<EOD
3  <html><body>
4  <p>Welcome! Today is the <?php echo(date('jS')); ?> of <?= date('F');
   ?>.</p>
5  </body></html>
6  Text outside of the HTML block.
7  EOD;
8      file_put_contents('sample.php', $str);
9
10     $handle = @fopen("sample.php", "r");
11     if ($handle) {
12         while (!feof($handle)) {
13             $buffer = fgetss($handle, 4096);
14             echo $buffer;
15         }
16         fclose($handle);
17     }
18
19     ?>
```

화면 출력

Welcome! Today is the of . Text outside of the HTML block.

06.6.4 파일 크기

파일을 읽고 처리하는 데 있어서 EOF를 검출하여 처리를 했습니다. PHP에서는 EOF를 확인하지 않고도 지정한 파일의 크기를 알 수 있는 내장 함수를 제공합니다.

| 내장 함수 |

```
int filesize ( string $filename )
```

내장 함수 filesize()는 파일의 크기를 쉽게 구할 수 있습니다. filesize() 함수는 파일의

크기를 정수값으로 반환합니다. 만일 파일이 존재하지 않으면 Warning 오류를 발생합
니다.

예제 파일 | filesize.php

```
1  <?php
2      $filename = "./readme.txt";
3
4      if (file_exists($filename)) {
5          echo $filename." 파일의 크기는".filesize($filename)." 입니다..
```

화면 출력

./readme.txt 파일의 크기는 14 입니다..
./readme2.txt 파일의 크기는 80 입니다..

06.6.5 지정 용량 읽기

fgetc(), fgets() 함수로 용량이 큰 파일을 글자 단위로 읽는 것은 서버와 스크립트 작업에 많은 부하를 줄 수 있습니다. 이런 경우 효율적으로 데이터를 읽기 위해서 일정 크기의 버퍼링을 통하여 파일을 읽을 수 있습니다.

한 번에 버퍼 형태로 데이터를 읽어오는 것이 빠른 성능 처리를 할 수 있습니다. fread() 함수는 지정한 바이트 크기만큼의 데이터를 읽어와 데이터를 반환합니다.

| 내장 함수 |

```
string fread ( resource $handle , int $length )
```

예제 파일 | **fread.php**

```
1  <?php
2      $filename = "./readme.txt";
3
4      if (file_exists($filename)) {
5          if ($fp = fopen($filename, "r")) {
6              $contents = fread($fp, filesize($filename));
7              echo $contents;
8          }
9          fclose($filename);
10     } else {
11         echo "파일을 찾을 수 없습니다.<br>";
12     }
13
14  ?>
```

화면 출력

hello jinyPHP3

06.6.6 포인터 조작

fgetc(), fgets() 등을 통하여 파일의 데이터를 읽는 경우에 함수를 반복적으로 호출하여 이용했습니다. 이렇게 함수를 한 번씩 호출할 때마다 파일 포인터는 자동으로 다음 위치로 이동을 합니다.

이렇게 이동된 파일 포인트의 위치를 확인하거나, 임의의 포인트로 이동할 수도 있습니다. 파일 포인트의 처리를 위한 몇 가지 내장 함수를 제공합니다.

| 내장 함수 |

```
int ftell ( resource $handle )
```

내장 함수 `ftell()`은 현재의 파일 포인터의 위치를 확인할 수 있습니다.

| 내장 함수 |

```
int fseek ( resource $handle , int $offset [, int $whence = SEEK_SET ] )
```

내장 함수 `fseek()`은 파일 포인터를 이동할 수 있습니다.

예제 파일 | `file_point.php`

```
1  <?php
2      $filename = './readme.txt';
3
4      if ($fp = fopen($filename, "r")) {
5          $data = fgets($fp);
6          echo $data."<br>";
7          echo "파일 포인터 = ".ftell($fp);
8
9          echo "<br>";
10
11         // 파일 포인터 처음으로 이동
12         fseek($fp, 0);
13         $data = fgets($fp);
14         echo $data."<br>";
15         echo "파일 포인터 = ".ftell($fp);
16
17         fclose($fp);
18     }
19
20  ?>
```

화면 출력

hello jinyPHP3

```
파일 포인트 = 14
hello jinyPHP3
파일 포인트 = 14
```

06.6.7 파일 콘텐츠 출력

PHP는 전형적인 복잡한 파일 포인트를 사용하지 않고, 간단하게 파일의 내용을 출력할 수 있도록 특별한 내장 함수를 지원합니다.

| 내장 함수 |

```
int readfile ( string $filename [, bool $use_include_path = false [, resource $context ] ] )
```

`readfile()` 함수는 정상적으로 파일을 읽었을 경우에 콘텐츠의 바이트 수를 반환합니다.

예제 파일 | **readfile.php**

```
1  <?php
2      echo readfile("log.txt");
3
4  ?>
```

06.7 파일 쓰기

`fopen()` 함수를 통하여 생성한 파일 입출력 포인터는 파일을 읽기와 함께 쓰기 작업도 할 수 있습니다. 파일 오픈 시 모드를 `w/a/x/c` 중의 하나로 설정하면 생성된 포인터를 통하여 파일을 저장할 수 있습니다.

쓰기 모드는 파일의 존재 여부와 상관 없습니다. 파일이 있으면 덮어쓰거나, 추가로 더 연결하여 쓸 수도 있습니다. 또는 파일이 없을 경우에는 새로이 파일이 생성됩니다.

예제 파일 | **file_write.php**

```
1  <?php
2      $filename = "./log.txt";
3
4      $fp = fopen($filename, "w");
5      if ($fp) {
6          echo "파일을 저장할 수 있습니다.<br>";
7      } else {
8          echo "Err] 파일을 오픈할 수 없습니다.<br>";
9      }
10  ?>
```

화면 출력

파일을 저장할 수 있습니다.

06.7.1 쓰기(W)

쓰기 모드로 오픈된 파일 포인터에 지정한 크기의 데이터를 출력할 수 있습니다. 포인터로 데이터를 출력하는 것은 파일 쓰기와 동일합니다. 하지만 파일 오픈 시 지정한 모드에 따라서 약간씩 쓰기 모드는 다릅니다. 새롭게 파일을 생성하거나 추가할 수 있습니다.

PHP는 파일에 텍스트/데이터를 삽입하기 위한 다음과 같은 내장 함수를 제공합니다.

| 내장 함수 |

```
int ( resource $handle , string $string [, int $length ] )
```

내장 함수 `fwrite()` 함수는 파일 포인터에 데이터를 저장하고, 파일 포인터를 다음 글자 저장을 위한 포인터의 위치도 함께 이동합니다.

예제 파일 | **fwrite.php**

```
1  <?php
2      $filename = "log.txt";
3
```

```

4 // w 모드는 새로운 파일을 생성합니다. 파일을 저장하는 위치는 처음입니다.
5 $fp = fopen($filename, "w");
6 if ($fp) {
7     echo "파일 내용을 저장합니다.<br>";
8
9     $log = $_SERVER['SERVER_NAME'].":". $_SERVER['REMOTE_ADDR'].":". $_SERVER['REQUEST_URI']. "\n";
10    $size = fwrite($fp, $log);
11    echo "저장: $size<br>";
12
13    fclose($fp);
14 } else {
15     echo "Err] 파일을 오픈할 수 없습니다.<br>";
16 }
17
18 ?>

```

화면 출력

파일 내용을 저장합니다.
저장: 17

06.7.2 쓰기 추가(a)

w 모드로 오픈된 파일 포인터는 새로운 내용으로 파일을 덮어쓰기 형태로 동작합니다. 만일 기존에 있는 파일에 추가로 내용을 쓰고 싶을 경우에는 a 모드로 파일 포인터를 오픈합니다.

a 모드로 오픈된 파일 포인터는 파일의 마지막 부분의 포인터를 반환합니다. 따라서 마지막 포인터를 기준으로 데이터를 추가로 삽입할 수 있습니다. 만일 지정한 파일이 없으면 새롭게 파일을 생성합니다. a 모드는 항상 추가할 수 있도록 파일 포인터는 맨 마지막을 가르킵니다.

06.7.3 파일 권한

리눅스와 같은 시스템의 경우 권한 설정을 통하여 파일 쓰기가 거부될 수도 있습니다.

다양한 운영체제와 권한 체계를 확인 후에 파일 쓰기 작업을 하는 것을 추천합니다. 사전 확인하는 코드를 삽입하여 안정적인 동작 처리를 수행할 수 있습니다.

| 내장 함수 |

```
bool is_writable( 파일명 )
```

내장 함수 `is_writable()`은 현재의 파일과 디렉터리에서 파일 쓰기가 가능한지를 논리값으로 반환합니다.

파일 쓰기 권한과 비슷하게 읽기 권한도 체크할 수 있는 내장 함수를 제공합니다.

| 내장 함수 |

```
bool is_readable( 파일명 )
```

내장 함수 `is_readable()`은 파일을 읽기 가능한지 확인합니다.

예제 파일 | `file_perms.php`

```
1  <?php
2      $filename = "log.txt";
3
4      // a 모드는 기존 파일에 새로운 추가 내용을 삽입합니다.
5      $fp = fopen($filename, "a");
6      if ($fp) {
7
8          if (is_writable($filename)) {
9              echo "파일 저장 허용.<br>";
10
11              $log = $_SERVER['SERVER_NAME'].":". $_SERVER['REMOTE_
12                  ADDR'].":". "\n";
13              $size = fwrite($fp, $log);
14              echo "저장: $size<br>";
15
16              fclose($fp);
```

```

16
17     } else {
18         echo "파일 쓰기 권한이 없습니다.<br>";
19     }
20
21 } else {
22     echo "Err] 파일을 오픈할 수 없습니다.<br>";
23 }
24
25 ?>

```

화면 출력

파일 저장 허용. 저장: 16

06.7.4 임시 파일

작업 디렉터리에 파일을 생성하여 쓰기 등의 출력 작업을 합니다. 이와 별개로 작업을 위해서 임시로 작업 디렉터리에 파일을 생성하는 불필요한 파일을 생성할 수 있습니다. 이런 경우, 임시 파일을 생성하여 처리할 수 있습니다.

내장 함수 `tmpfile()`은 임시 파일을 생성할 수 있습니다.

| 내장 함수 |

resource **tmpfile** (void)

예제 파일 | **tmpfile.php**

```

1  <?php
2      $temp = tmpfile();
3      fwrite($temp, "writing to tmpfile");
4      fseek($temp, 0);
5      echo fread($temp, 1024);
6      fclose($temp); // this removes the file
7  ?>

```

| 내장 함수 |

```
string tempnam ( string $dir , string $prefix )
```

내장 함수 `tempnam()`은 고유한 임시 파일을 만듭니다.

예제 파일 | **tempnam.php**

```
1  <?php
2      $tmpfname = tempnam("/tmp", "FOO");
3
4      $handle = fopen($tmpfname, "w");
5      fwrite($handle, "writing to tempfile");
6      fclose($handle);
7
8      // do here something
9      unlink($tmpfname);
10
11  ?>
```

| 내장 함수 |

```
string sys_get_temp_dir ( void )
```

내장 함수 `sys_get_temp_dir()`은 임시 파일로 사용되는 디렉터리 경로를 반환합니다.

예제 파일 | **sys_get_temp_dir.php**

```
1  <?php
2      $temp_file = sys_get_temp_dir();
3      echo $temp_file;
4      ?>
```

화면 출력

C:\Users\infoh\AppData\Local\Temp

06.7.5 파일 잠금과 해제

파일은 여러 사람들이 읽거나 쓸 수 있습니다. 만일 동시에 파일을 쓰거나, 읽을 경우에는 서로의 우선순위와 동시 작업 등으로 인하여 액세스 충돌이 발생할 수 있습니다.

이런 경우 먼저 파일의 액세스 등의 충돌을 방지하기 위하여 파일을 잠시 잠금을 설정할 수 있습니다.

| 내장 함수 |

```
bool flock ( resource $handle , int $operation [, int &$wouldblock ] )
```

내장 함수 flock()은 해당 파일 포인터를 잠금 처리합니다.

- LOCK_SH: 공유 잠금, 쓰기만 차단합니다.
- LOCK_EX: 독점 잠금, 읽기/쓰기 모두 차단합니다.
- LOCK_UN: 잠금 해제
- LOCK_NB: 잠겨 있을 경우 false를 반환합니다.

flock() 함수의 LOCK_UN 옵션을 통하여 잠금을 해제할 수 있습니다. 하지만 잠금을 해제하기 전에 fflush() 함수를 통하여 버퍼를 정리한 후에 잠금을 해제하는 것을 권장합니다.

| 내장 함수 |

```
bool fflush ( resource $handle )
```

예제 파일 | file_lock.php

```
1  <?php
2      $filename = "log.txt";
3
4      if (is_writable($filename)) {
5          echo "파일 저장 허용.<br>";
```

```

6
7     // a 모드는 기존 파일에 새로운 추가 내용을 삽입합니다.
8     $fp = fopen($filename, "a");
9     if (is_resource($fp)) {
10
11         // 잠금 설정
12         flock($fp, LOCK_EX);
13
14         $log = $_SERVER['SERVER_NAME'].":". $_SERVER['REMOTE_
15 ADDR'].":". "\n";
16         $size = fwrite($fp, $log);
17         echo "저장: $size<br>";
18
19         // 버퍼 정리 후에 잠금 해제
20         fflush($fp);
21         flock($fp, LOCK_UN);
22
23         fclose($fp);
24     } else {
25         echo "Err] 파일을 오픈할 수 없습니다.<br>";
26     }
27
28 } else {
29     echo "파일 쓰기 권한이 없습니다.<br>";
30 }
31
32 ?>

```

화면 출력

파일 저장 허용.

저장: 16

06.8 파일 삭제

생성된 파일 또는 기존의 파일을 PHP 코드를 통하여 삭제할 수 있습니다. 파일을 삭제하기 전에 반드시 백업 및 주의하여 삭제하는 습관이 필요합니다.

PHP에서는 파일 삭제와 관련된 다음과 같은 내장 함수를 제공합니다.

| 내장 함수 |

```
bool unlink ( string $filename [, resource $context ] )
```

내장 함수 unlink()는 서버의 파일을 삭제합니다. 파일을 삭제하기 위해서는 해당 디렉터리에 쓰기 권한이 있어야 합니다.

예제 파일 | **unlink.php**

```
1  <?php
2      $filename = "log.txt";
3
4      if (file_exists($filename)) {
5          if (is_writeable($filename)) {
6              unlink($filename);
7              echo $filename." 파일을 삭제합니다.<br>";
8          } else {
9              echo "삭제 권한이 없습니다.<br>";
10         }
11     } else {
12         echo "Err] 파일이 존재하지 않습니다.<br>";
13     }
14
15  ?>
```

화면 출력

log.txt 파일을 삭제합니다.

06.9 파일 복사

기존 존재하는 파일을 새로운 파일 이름으로 복사할 수 있습니다. 파일을 복사할 때 파일명이 중복되면 덮어쓰기로 복사하게 됩니다.

PHP에서는 파일 복사와 관련된 다음과 같은 내장 함수를 제공합니다.

| 내장 함수 |

```
bool copy ( string $source , string $dest [, resource $context ] )
```

내장 함수 `copy()`는 파일 복사후 성공 여부를 논리값으로 반환합니다.

예제 파일 | **copy.php**

```
1  <?php
2      $filename = "readme.txt";
3
4      if (file_exists($filename)) {
5
6          if (copy($filename, "readme_copy.txt")) {
7              echo "복사 성공!<br>";
8          } else {
9              echo "복사 실패!<br>";
10         }
11
12     } else {
13         echo "Err] 파일이 존재하지 않습니다.<br>";
14     }
15
16  ?>
```

화면 출력

복사 성공!

내장 함수 `is_uploaded_file()`은 HTTP POST를 통해 파일이 업로드됐는지 확인합니다.

| 내장 함수 |

```
bool is_uploaded_file ( string $filename )
```

예제 파일 | **is_uploaded_file.php**

```
1  <?php
2
3      if (is_uploaded_file($_FILES['userfile']['tmp_name'])) {
4          echo "File ". $_FILES['userfile']['name'] ." uploaded
           successfully.\n";
5          echo "Displaying contents\n";
6          readfile($_FILES['userfile']['tmp_name']);
7      } else {
8          echo "Possible file upload attack: ";
9          echo "filename '". $_FILES['userfile']['tmp_name'] . "'.";
10     }
11
12  ?>
```

내장 함수 `move_uploaded_file()`은 업로드된 파일을 새 위치로 이동합니다.

```
bool move_uploaded_file ( string $filename , string $destination )
```

예제 파일 | **move_uploaded_file.php**

```
1  <?php
2      $uploads_dir = '/uploads';
3      foreach ($_FILES["pictures"]["error"] as $key => $error) {
4          if ($error == UPLOAD_ERR_OK) {
5              $tmp_name = $_FILES["pictures"]["tmp_name"][$key];
6
7              $name = basename($_FILES["pictures"]["name"][$key]);
8              move_uploaded_file($tmp_name, "$uploads_dir/$name");
9          }
10     }
11
12  ?>
```

06.10 file

PHP는 파일 포인터를 통하여 입출력을 하는 것과 달리 보다 쉽게 파일을 출력할 수 있는 함수를 몇 가지 제공합니다.

| 내장 함수 | 한 줄 단위 출력

```
array file ( string $filename [, int $flags = 0 [, resource $context ] ] )
```

내장 함수 file()은 fgets() 함수처럼 파일의 데이터를 한 줄씩 읽어서 배열로 반환합니다. 반환된 데이터는 반복문을 통하여 한 줄씩 출력할 수 있습니다.

| 내장 함수 | 파일 전체 입력

```
string file_get_contents ( string $filename [, bool $use_include_path = false [, resource $context [, int $offset = 0 [, int $maxlen ] ] ] ] )
```

내장 함수 file_get_contents()는 readfile() 함수와 비슷하게 파일의 전체 콘텐츠 내용을 읽어옵니다.

| 내장 함수 | 파일 전체 출력

```
int file_put_contents ( string $filename , mixed $data [, int $flags = 0 [, resource $context ] ] )
```

내장 함수 file_put_contents()는 fwrite()와 같이 콘텐츠를 파일로 저장할 수 있는 기능을 제공합니다.

예제 파일 | files.php

```
1 <?php
2     $filename = "readme.txt";
3
```

```

4     if (file_exists($filename)) {
5
6         // 파일 전체 출력
7         echo "=== 파일 전체 출력 ===<br>";
8         $body = file_get_contents($filename);
9         echo $body;
10
11        // 한 줄씩 출력
12        echo "<br> === 한 줄씩 출력 ===<br>";
13        $lines = file("readme2.txt");
14        foreach ($lines as $line_num => $line) {
15            echo "Line #<b>{$line_num}</b> : " . htmlspecialchars($line) .
16                "<br />\n";
17        }
18
19        // 파일 저장
20        echo "<br> === 파일 저장 ===<br>";
21        file_put_contents("readme3.txt", $body);
22    } else {
23        echo "Err] 파일이 존재하지 않습니다.<br>";
24    }
25
26    ?>

```

화면 출력

```

=== 파일 전체 출력 ===
hello jinyPHP3
=== 한 줄씩 출력 ===
Line #0 : 안녕하세요.
Line #1 : 여러 줄로 구성된 파일을
Line #2 : 읽어올 수 있습니다.

=== 파일 저장 ===

```

단, file 관련 함수를 통하면 fopen() 함수로 처리하던 잠금 설정은 할 수 없습니다. 하지만 쉽게 파일의 입출력을 할 수 있는 장점이 있습니다.

06.11 그 외 함수

내장 함수 `popen()`은 파이프 오픈을 처리합니다.

| 내장 함수 |

resource **popen** (string \$command , string \$mode)

예제 파일 | **popen.php**

```
1  <?php
2      $handle = popen("/bin/ls", "r");
3
4  ?>
```

내장 함수 `pclose()`은 오픈된 파이프를 닫습니다.

| 내장 함수 |

int **pclose** (resource \$handle)

예제 파일 | **pclose.php**

```
1  <?php
2      $handle = popen('/bin/ls', 'r');
3      pclose($handle);
4
5  ?>
```

내장 함수 `fscanf()`는 형식에 따라 파일에서 입력 구문을 분석합니다.

| 내장 함수 |

mixed **fscanf** (resource \$handle , string \$format [, mixed &\$...])

예제 파일 | **fscanf.php**

```
1  <?php
2  $handle = fopen("users.txt", "r");
3  while ($userinfo = fscanf($handle, "%s\t%s\t%s\n")) {
4      list ($name, $profession, $countrycode) = $userinfo;
5      //... do something with the values
6  }
7  fclose($handle);
8
9  ?>
```

내장 함수 `fstat()`는 열린 파일에 대한 정보를 반환합니다.

| 내장 함수 |

array **fstat** (resource \$handle)

예제 파일 | **fstat.php**

```
1  <?php
2
3      // open a file
4  $fp = fopen("/etc/passwd", "r");
5
6      // gather statistics
7  $fstat = fstat($fp);
8
9      // close the file
10  fclose($fp);
11
12      // print only the associative part
13  print_r(array_slice($fstat, 13));
14
15  ?>
```

내장 함수 `ftruncate()`는 열린 파일을 지정된 길이로 절단합니다.

| 내장 함수 |

```
bool ftruncate ( resource $handle , int $size )
```

예제 파일 | **ftruncate.php**

```
1  <?php
2      $filename = 'lorem_ipsum.txt';
3
4      $handle = fopen($filename, 'r+');
5      ftruncate($handle, rand(1, filesize($filename)));
6      rewind($handle);
7      echo fread($handle, filesize($filename));
8      fclose($handle);
9  ?>
```

내장 함수 `parse_ini_file()`은 구성 파일을 구문 분석합니다.

| 내장 함수 |

```
array parse_ini_file ( string $filename [, bool $process_sections = false [, int  
$scanner_mode = INI_SCANNER_NORMAL ]])
```

예제 파일 | **parse_ini_file.php**

```
1  <?php
2
3      define('BIRD', 'Dodo bird');
4
5      // Parse without sections
6      $ini_array = parse_ini_file("sample.ini");
7      print_r($ini_array);
8
9      // Parse with sections
10     $ini_array = parse_ini_file("sample.ini", true);
11     print_r($ini_array);
12  ?>
```

내장 함수 `parse_ini_string()`은 구성 문자열을 구문 분석합니다.

| 내장 함수 |

```
array parse_ini_string ( string $ini [, bool $process_sections = false [, int $scanner_  
mode = INI_SCANNER_NORMAL ] ] )
```

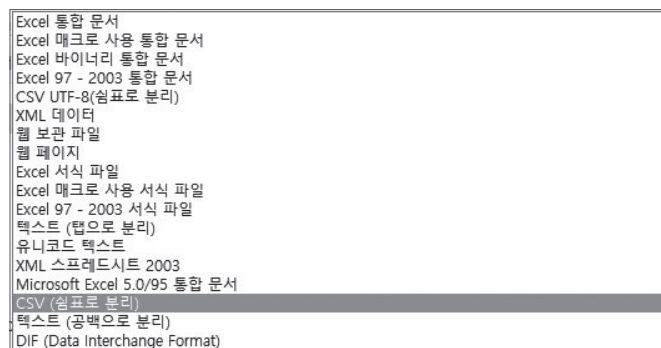

07

CSV

CSV 파일은 엑셀 등 데이터베이스의 자료들을 서로 주고받을 때 자주 사용하는 순수 데이터 포맷 파일의 일종입니다. CSV는 데이터를 외부 파일로 출력하고, 외부 데이터를 일괄 처리하여 입력받을 때 많이 사용합니다.

07.1 샘플 데이터

엑셀 프로그램 등을 통하여 작성한 데이터를 CVS 포맷으로 저장하면 됩니다. 파일 저장 형식을 선택하면 다양한 파일 형식을 볼 수 있습니다.



CSV 파일로 엑셀 파일을 저장할 때 각각의 데이터는 콤마(,)로 구분하여 한 줄에 1개의 레코드 데이터를 저장합니다.

07.2 CSV 쓰기

PHP에서 생성한 데이터를 CSV 형태의 파일로 출력할 수 있습니다. 출력된 CSV 파일은 엑셀 등의 응용프로그램에서 읽고, 데이터를 가공할 수 있습니다.

fputcsv() 함수는 입력된 Array 배열을 CSV 형식으로 줄을 만들고 파일 포인터에 씁니다.

| 내장 함수 |

```
int fputcsv ( resource $handle , array $fields [, string $delimiter = "," [, string $enclosure = "" [, string $escape_char = "\" ]]] )
```

fputcsv()는 필드 배열로 전달된 행을 CSV로 형식으로 지정합니다. 지정된 파일 핸들에 새로운 행으로 끝내도록 출력합니다.

예제 파일 | csv_write.php

```
1  <?php
2
3      $list = array (
4          array(1, "aaa", 18, "Seoul"),
5          array(2, "bbb", 20, "Daejeon"),
6          array(3, "ccc", 30, "Incheon ")
7      );
8
9      $csvFile = "sample.csv";
10
11     $fp = fopen($csvFile, 'w');
12     if (!is_resource($fp)) {
13         die("저장할 파일 포인터를 생성할 수 없습니다.");
14     } else {
```

```

15
16         // 파일 독점 잠금 설정
17         flock($fp, LOCK_EX);
18
19         // 배열을 CSV 파일로 저장
20         foreach ($list as $fields) {
21             fputcsv($fp, $fields);
22         }
23
24         // 파일 잠금 해제
25         fflush($fp);
26         flock($fp, LOCK_UN);
27
28         fclose($fp);
29     }
30
31     ?>

```

지정한 Array 데이터를 sample.csv 파일로 출력을 합니다.

파일 내용

```

1,aaa,18,Seoul
2,bbb,20,Daejeon
3,ccc,30,"Incheon "

```

07.3 CSV 읽기

PHP는 CVS 파일의 데이터를 쉽게 읽고 처리하기 위해서 fgetcsv() 함수를 제공합니다. fgetcsv() 함수는 파일 포인터에서 한 줄 단위로 데이터를 읽어와 \$delimiter 기호를 기준으로 CSV 필드를 구문 분석합니다.

| 내장 함수 |

```

array fgetcsv ( resource $handle [, int $length = 0 [, string $delimiter = "," [, string
$enclosure = "" [, string $escape = "\" ]]] )

```

fgetcsv()가 CSV 형식의 필드에 대해 읽은 행을 구문 분석하고 읽은 필드가 들어 있습니다. 배열을 반환한다는 점을 제외하면 fgets()와 비슷한 동작을 수행합니다.

예제 파일 | csv_read.php.php

```
1  <?php
2      $csvFile = "sample.csv";
3
4      // 먼저 CSV 파일 존재 유무를 확인합니다.
5      if (!file_exists($csvFile)) {
6          die("CSV 파일이 존재하지 않습니다.");
7      } else {
8
9          // CSV 파일을 읽어옵니다.
10         $row = 1;
11         if (($fp = fopen($csvFile, "r")) !== FALSE) {
12             // CSV 데이터 한 줄을 읽어옵니다.
13             $length = 1000;
14             $delimiter = ",";
15             while (($data = fgetcsv($fp, $length, $delimiter)) !==
16                 FALSE) {
17                 $columns = count($data);
18
19                 echo "라인($row): 컬럼수($columns)<br/>";
20                 $row++;
21                 for ($i=0; $i < $columns; $i++) {
22                     echo $data[$i] . "<br />";
23                 }
24             }
25             fclose($fp);
26         }
27     }
28
29  ?>
```

화면 출력

라인(1): 컬럼수(4)
1
aaa


```

18
Seoul
라인(2): 컬럼수(4)
2
bbb
20
Daejeon
라인(3): 컬럼수(4)
3
ccc
30
Incheon

```

| 내장 함수 |

```

array str_getcsv ( string $input [, string $delimiter = "," [, string $enclosure = "" [,
string $escape = "\\" ]]] )

```

내장 함수 `str_getcsv()`는 CSV 문자열을 배열로 구문 분석합니다.

예제 파일 | `str_getcsv.php`

```

1  <?php
2      $csv = array_map('str_getcsv', file('sample.csv'));
3      print_r($csv);
4
5  ?>

```

화면 출력

```

Array ( [0] => Array ( [0] => 1 [1] => aaa [2] => 18 [3] => Seoul ) [1] => Array (
[0] => 2 [1] => bbb [2] => 20 [3] => Daejeon ) [2] => Array ( [0] => 3 [1] =>
ccc [2] => 30 [3] => Incheon ) )

```


08

정규표현식

PHP는 정규표현에 관련된 기능과 함수를 제공합니다. 정규표현식은 PHP 언어 이외에도 다른 언어에서 일반적으로 사용하는 패턴을 처리하는 개별 언어의 일종입니다.

PHP는 정규표현에 대해서 perl(PCRE) 방식을 지원합니다. 기존 POSIX 스타일의 정규표현은 PHP 5.3.x 이후에서 삭제되었습니다.

정규표현은 특수한 문자들의 패턴입니다.

08.1 정규 패턴

내장 함수 `preg_match()`는 패턴 정규표현식을 입력된 문자열 `$subject`에서 찾습니다. `preg_match()` 함수는 `subject` 문자열에서 패턴을 매칭하여 결과값을 `$matches` 변수에 배열로 반환합니다.

| 내장 함수 |

```
int preg_match ( string $pattern , string $subject [, array &$matches [, int $flags = 0 [,  
int $offset = 0 ]]] )
```

이 전에 `ereg()` 함수를 사용했으나 PHP 7.x에서 예전 함수는 삭제되었습니다.

- `matches`: 세 번째 인자 `matches`가 있을 경우에는 검색 결과를 배열로 반환합니다.
- `flags`: `PREG_OFFSET_CAPTURE` 값은 매치된 모든 문자열의 시작 위치를 함께 반환합니다.
- `offset`: 문자열 검색의 시작 포인트를 지정할 수 있습니다. 기본값은 0입니다.

예제 파일 | `preg-01.php`

```
1  <?php
2
3      $string = "2017-06-33";
4      $pattern = "/([0-9]{4})-([0-9]{1,2})-([0-9]{1,2})/";
5
6      if (preg_match($pattern, $string,$matches) ) {
7          echo"날짜 형식이 맞습니다.<br>";
8
9          if (checkdate($match[2],$match[3],$match[1])) {
10             echo "유효한 날짜입니다.<br>";
11         } else {
12             echo "날짜 형식은 맞지만, 유효한 날짜는 아닙니다.<br>";
13         }
14     } else {
15         echo"날짜 형식이 다릅니다.<br>";
16     }
17
18  ?>
```

화면 출력

날짜 형식이 맞습니다.

날짜 형식은 맞지만, 유효한 날짜는 아닙니다.

08.2 패턴 변환

내장 함수 `preg_replace()`는 패턴을 찾아서 다음 문자열로 변환합니다. 문자열 `subject`에서 패턴 배열의 값과 같은 값들을 검사하여 `replacement` 문자열로 변환합니다.

| 내장 함수 |

```
mixed preg_replace ( mixed $pattern , mixed $replacement , mixed $subject [, int $limit [, int &$count ]])
```

예제 파일 | `preg-02.php`

```
1  <?php
2      $string = 'jun 21, 2017';
3      $pattern = '/(\w+) (\d+), (\d+)/i';
4      $replacement = '${1} ${2}, ${3}';
5      echo preg_replace($pattern, $replacement, $string);
6      echo "<br>";
7
8      $string = 'hello world! my name is jiny.';
9
10     $patterns[0] = '/hello world/';
11     $patterns[1] = '/my name is/';
12     $patterns[2] = '/jiny/';
13
14     $replacements[2] = '안녕하세요';
15     $replacements[1] = '제 이름은 ';
16     $replacements[0] = '지니';
17
18     echo preg_replace($patterns, $replacements, $string);
19
20  ?>
```

화면 출력

```
jun 21, 2017
안녕하세요! 제 이름은 지니.
```

08.3 패턴 분리

내장 함수 `preg_split()`는 정규표현 패턴에 따라서 문자열을 분리하여 배열로 반환합니다.

| 내장 함수 |

```
array preg_split ( string $pattern , string $subject [, int $limit [, int $flags ] ] )
```

예제 파일 | `preg-03.php`

```
1  <?php
2      // " ", \r, \t, \n, \f를 포함하여
3      $keywords = preg_split("/[\\s,]+/", "php language, programming");
4      echo "임의 개수의 콤마와 스페이스로 구문을 나눕니다. ";
5      print_r($keywords);
6      echo "<br>";
7
8
9      $str = 'string';
10     echo "문자 단위로 분리합니다. ";
11     $chars = preg_split('//', $str, -1, PREG_SPLIT_NO_EMPTY);
12     print_r($chars);
13     echo "<br>";
14
15  ?>
```

화면 출력

임의 개수의 콤마와 스페이스로 구문을 나눕니다. Array ([0] => php [1] => language [2] => programming)

문자 단위로 분리합니다. Array ([0] => s [1] => t [2] => r [3] => i [4] => n [5] => g)

| 내장 함수 |

```
string preg_quote ( string $str [, string $delimiter ] )
```

내장 함수 `preg_quote()`는 정규표현식의 문자를 인용합니다. `preg_quote()` 함수는 입력된 문자열에 대해서 모든 문자 앞에 백슬래시를 추가합니다.

이러한 표현은 정규표현식을 처리하는 문자열 처리할 때 유용합니다.

정규표현식 특수 문자: `. \w + * ? [^] $ () { } = ! < > | :`

예제 파일 | `preg-04.php`

```
1  <?php
2      $keywords = '$123 for a hojin/jiny';
3      $keywords = preg_quote($keywords, '/');
4      echo $keywords;
5      echo "<br>";
6
7      // preg_quote($word)는 정규표현식에서 이스터라이크(*) 처리
8      $body = "안녕하세요 이 책은 매우 *쉽게* 작성을 하였습니다.";
9      $word = "*매우*";
10     $body = preg_replace("/".preg_quote($word)."/", "<i>" . $word . "</i>", $body);
11     echo $body;
12
13  ?>
```

화면 출력

`\$123 for a hojin\jiny`

안녕하세요 이 책은 매우 *쉽게* 작성을 하였습니다.

09

URL

09.1 네트워크 연결

09.1.1 IP 주소

IP 주소는 Internet Protocol의 약자입니다. 인터넷과 연결된 컴퓨터, 서버, 모바일 와 이파이 등의 기기들은 상호 데이터를 전송하고 기기를 구분하기 위한 IP 주소를 할당합니다.

IP 주소는 컴퓨터, 모바일 기기들이 네트워크에 접속 시 상호 주소를 통하여 데이터를 주고받는 통신을 처리하게 됩니다. IP 주소는 컴퓨터들 간의 통신을 위하여 인식할 수 있는 특수한 번호입니다.

IP 주소는 0~255까지의 표현할 수 있는 1바이트의 수, 4개로 구성되어 있습니다. 또한 각각의 수 사이에는 점(.)을 통하여 구분하여 표기를 합니다.

xxx.xxx.xxx.xxx

즉 인터넷 IP 주소는 4개의 바이트인 32비트의 값으로 표현을 하게 됩니다. 32비트는 $256 \times 256 \times 256 \times 256 = 4,294,967,296$ 개의 IP 주소를 생성할 수 있습니다.

이렇게 32비트로 구성된 IP 주소를 IPv4라고 합니다. 우리가 일반적으로 알고 있는 IP 주소 형식입니다.

IP 주소 관리 업무는 APNIC 등 국제기구에서 합니다. 또한 국가별로 IP 주소를 대행하여 관하는데, 대한민국의 경우 인터넷 주소 자원 관리 기관에서 업무를 처리하고 있습니다.

09.1.2 IP 주소의 구성

IP 주소는 4바이트로 구성된 32비트 값입니다. 전 세계 접속한 수많은 컴퓨터들을 구분하기 위해서 IP 주소는 크게 2개의 부분으로 나누어 IP 값을 분리합니다.

IP 주소 = 네트워크 주소 + 호스트 주소

이렇게 범위를 분리하는 이유는 큰 그룹의 네트워크와 회사 내 호스트 컴퓨터인지를 구분하기 위함입니다. IP 주소 구분을 명칭하는 다른 이름으로는 CLASS라는 표현을 사용합니다.

네트워크 주소

4개로 구성된 IP 주소의 첫 번째 0~255까지의 값을 네트워크 주소로 명칭합니다. 첫 숫자는 반드시 0에서 127 사이에 있어야 하는 것이 규칙입니다.

호스트 주소

나머지 3개의 숫자를 호스트 주소로 명칭합니다. 이 3개의 숫자값을 기준으로 CLASS A, B, C 형태로 구분하여 표시합니다. 두 번째 숫자의 그룹을 A 클래스, 세 번째 숫자를 B 클래스, 네 번째 숫자를 C 클래스라고 표현합니다.

클래스 A

A 클래스는 숫자.xxx.xxx.xxx 형태로 $256 \times 256 \times 256$ 개 = 16,777,216개의 호스트를 할당

하고 관리할 수 있습니다. A 클래스 주소는 많은 수의 컴퓨터를 할당하고 관리할 수 있습니다. 주로 큰 기관에서 할당되는 클래스입니다.

클래스 B

B 클래스는 숫자.숫자.숫자.숫자 형태로 256×256 개 = 65,536개의 호스트를 할당하고 관리할 수 있습니다.

클래스 C

C 클래스는 숫자.숫자.숫자.숫자 형태로 256의 호스트를 할당하고 관리할 수 있습니다.

서브넷 마스크

서브넷 마스크(Subnet Mask)는 IP 주소를 비트 연산을 통하여 필터링해주는 특수한 값입니다. 또 다른 표현으로 네트워크 마스크(Network mask)라고도 합니다. 서브넷 마스크의 역할은 네트워크 부분과 호스트를 클래스 단위로 구분하는 것입니다.

Class A 255. 0. 0. 0

Class B 255. 255. 0. 0

Class C 255. 255. 255. 0

게이트웨이

게이트웨이(Gateway)란 네트워크의 특수한 IP 주소입니다. 서브넷 마스크로 필터링된 클래스 안은 내부 네트워크입니다. 다른 외부의 네트워크와는 차단되어 있습니다. 하지만 인터넷과 같이 외부의 네트워크로 연결하기 위해서는 게이트웨이라는 특별한 IP 주소를 통하여 외부와 통신을 처리하게 됩니다.

게이트웨이는 자신의 속한 클래스 안에서 임의의 IP 주소를 하나 할당하여 사용하면 됩니다. 보통 게이트웨이 주소로는 xxx.xxx.xxx.1 또는 xxx.xxx.xxx.254와 같이 첫 번째, 마지막 번호를 자주 사용합니다.

맥 주소

맥 주소(Mac Address)는 IP 주소가 할당되는 하드웨어의 고유 기기 번호입니다. IP 주소는 네트워크에 연결된 컴퓨터 주소를 말하지만, 맥 주소는 컴퓨터 하드웨어 자체의 네트워크 장비의 번호입니다.

맥 주소는 6개의 바이트로 구성됩니다. 또한 헥사 코드값으로 표현하며, 각각의 값 사이에는 콜론(:)으로 구분합니다.

xx:xx:xx:xx:xx:xx

09.1.3 IP6V

기존 32비트 주소 체계인 IP4v 는 인터넷이 처음 시작되던 시점에는 전 세계의 모든 컴퓨터를 커버할 수 있을 것이라 생각했습니다. 하지만 전 세계 컴퓨터 사용량이 늘어나고, 모바일 기기의 등장으로 한 사람이 사용하는 디지털 장비들은 더욱 늘어나게 되었습니다.

그래서 우리는 편법으로 현재 사용하는 장비에만 IP 주소를 임시적으로 할당하는 편법을 사용하게 되었습니다. 동적 IP 주소(dynamic IP address)는 주소를 장비에 할당했다가 사용하지 않을 때는 회수하여 다른 장비에 할당하는 기술입니다.

하지만 이러한 기술로도 더 이상 늘어나는 디지털 장비들을 수용하기란 어려워졌습니다. 그래서 등장한 새로운 IP 주소 체계가 IP6v입니다.

기존 IP4v의 42억9496만7296개의 숫자는 이제 충분한 숫자가 아니게 되었습니다. 이 부족한 숫자 체계를 보완하고자 만든 것이 IP6v 128비트 체계입니다.

09.1.4 IP 및 맥 주소 확인 방법

윈도우

콘솔 창에서 ipconfig/all 명령을 입력하면 본인 컴퓨터의 IP 주소와 맥주소를 확인할 수 있습니다.

```
명령 프롬프트
Microsoft Windows [Version 10.0.16299.64]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\winfoh>ipconfig/all

Windows IP 구성

호스트 이름 . . . . . : LAPTOP-M0820HEF
주 DNS 접미사 . . . . . : 
노드 ID . . . . . : 
IP 라우팅 사용 . . . . . : 아니요
WINS 프록시 사용 . . . . . : 아니요

무선 LAN 어댑터 로컬 영역 연결* 11:

미디어 상태 . . . . . : 미디어 연결 끊김
연결별 DNS 접미사 . . . . . : 
설명 . . . . . : Microsoft Wi-Fi Direct Virtual Adapter
물리적 주소 . . . . . : F8-63-3F-0F-A1-E8
DHCP 사용 . . . . . : 예
자동 구성 사용 . . . . . : 예

무선 LAN 어댑터 Wi-Fi:

연결별 DNS 접미사 . . . . . : 
설명 . . . . . : Intel(R) Dual Band Wireless-AC 8265
물리적 주소 . . . . . : F8-63-3F-0F-A1-EA
DHCP 사용 . . . . . : 예
자동 구성 사용 . . . . . : 예
IPv4 주소 . . . . . : fe80::35a3:792d:c9be:9057%6(기본 설정)
IPv4 주소 . . . . . : 192.168.0.12(기본 설정)
서브넷 마스크 . . . . . : 255.255.255.0
인대 시작 날짜 . . . . . : 2017년 12월 1일 금요일 오후 4:50:44
인대 만료 날짜 . . . . . : 2017년 12월 1일 금요일 오후 7:17:11
기본 게이트웨이 . . . . . : 192.168.0.1
DHCP 서버 . . . . . : 192.168.0.1
DHCPv6 IAID . . . . . : 150496063
DHCPv6 클라이언트 DUID . . . . . : 00-01-00-01-20-C8-C1-E6-F8-63-3F-0F-A1-EA
DNS 서버 . . . . . : 168.126.63.1
                        168.126.63.2
```

리눅스

터미널 창에서 ipconfig 명령을 통하여 확인할 수 있습니다.

안드로이드

‘설정’ → ‘일반’ → ‘휴대폰정보’ → ‘하드웨어 정보’를 선택합니다.

09.1.5 도메인 주소

IP 주소를 통하여 서버 컴퓨터를 찾기란 매우 어렵습니다. 접속하려고 하는 모든 서버의 IP 주소를 외워서 인터넷을 사용하기란 힘듭니다. 따라서 이러한 IP 주소들을 가상의 별칭으로 할당하여 사용하는 것을 도메인이라고 합니다.

도메인은 복잡한 숫자 체계인 IP 주소보다 사람이 쉽게 이해할 수 있는 이름 형태로 되어 있어서 좀 더 직관적으로 인터넷을 접속하고 사용할 수 있습니다.

인터넷 브라우저 창에서 도메인 주소를 입력하면 해당 서비스 페이지로 접속합니다. 하지만 도메인 이름으로 어떻게 해당 서버를 찾을 수 있을까요? 이런 작업이 가능한 이유는 도메인 이름을 IP 주소로 변경 및 관리해주는 DNS 서버가 있기 때문입니다.

DNS 서버들은 서로 유기적으로 동작을 하면서 도메인 이름을 IP 주소로 변경하는 기능을 합니다. 서비스를 운영하는 서버들은 자체적으로 DNS 서비스를 운영하거나, 호스팅 회사에서 운영하는 DNS 서비스를 이용하기도 합니다.

예전에는 1개의 도메인 이름에 1개의 서버 IP를 할당하여 서비스를 구축했습니다. 하지만 사용자 접속이 많아지면서 1대의 서버만으로는 서비스를 운영하기는 매우 어려울 것입니다. 만일 `www.google.com` 사이트가 서버 1대로 전 세계의 사용자에게 서비스 한다면 그 서버는 얼마나 큰 장비일까요? 대형 서비스를 운영하기 위해서는 다수의 서버 장비의 IP를 하나의 도메인으로 연결하여 사용자 접속을 분산합니다.

이런 점에서 IP 주소만 가지고 서비스를 만들거나 접속할 수는 없습니다. 도메인은 쉽게 IP 주소를 일반적인 이름으로 변경해주는 것과 동시에 대용량의 서비스를 구축할 수 있도록 환경을 제공합니다.

09.1.6 URL 쿼리스트링

도메인은 인터넷 서비스를 접속하는 데 매우 유용한 접속 방법입니다. 또한 웹 서비스들은 다양한 정보를 사용자에게 제공합니다.

사용자가 다양한 웹 서비스의 정보를 얻기 위해서 각각의 정보마다 도메인 URL을 제공한다고 하면 DNS 시스템은 과부하를 걸치게 될 것입니다. 따라서 웹 서비스들은 하나의 도메인에 각각의 서비스를 요청 정보를 함께 전송할 수 있는 쿼리스트링을 지원합니다.

쿼리스트링을 다른 표현으로 GET 방식의 URL 요청이라고 하기도 합니다. 쿼리스트링은 다음과 같이 표현합니다.

`http://www.도메인.com?변수=값&변수=값&변수=값`

도메인 주소 다음에 ? 기호는 쿼리스트링을 시작하는 표시입니다. 쿼리스트링은 '변수=값'과 같이 한 쌍의 데이터로 표현하며, 다수의 데이터를 전송할 때는 & 기호로 구분합니다.

쿼리스트링을 이용하면 쉽게 웹 서비스의 데이터를 선택하여 접속할 수 있습니다.

09.2 URL 함수

PHP에서는 URL 구조를 분석하고 처리할 수 있는 몇 개의 내장 함수들을 지원합니다. URL 함수를 이용하면 복잡한 URL 문자열을 하나씩 처리하지 않고 빠르게 작업을 수행할 수 있습니다.

09.2.1 URL 구분

내장 함수 `parse_url()`은 URL 스트링을 해석하여 구성요소를 배열로 반환합니다.

| 내장 함수 |

```
mixed parse_url ( string $url [, int $component = -1 ] )
```

예제 파일 | [url-01.php](#)

```
1  <?php
2      $url = 'http://username:password@hostname:9090/path?arg=value#anchor';
3
4      // url을 분석하여 배열로 반환합니다.
5      print_r(parse_url($url));
6      echo "<br>";
7
8      // 키 값을 통해서 하나씩 추출이 가능합니다.
9      echo "PHP_URL_SCHEME = ". parse_url($url, PHP_URL_SCHEME). "<br>";
10
11     echo "PHP_URL_USER = ". parse_url($url, PHP_URL_USER) . "<br>";
```

```

12     echo "PHP_URL_PASS =" . parse_url($url, PHP_URL_PASS) . "<br>";
13     echo "PHP_URL_HOST =" . parse_url($url, PHP_URL_HOST) . "<br>";
14     echo "PHP_URL_PORT =" . parse_url($url, PHP_URL_PORT) . "<br>";
15     echo "PHP_URL_PATH =" . parse_url($url, PHP_URL_PATH) . "<br>";
16     echo "PHP_URL_QUERY =" . parse_url($url, PHP_URL_QUERY) . "<br>";
17     echo "PHP_URL_FRAGMENT =" . parse_url($url, PHP_URL_FRAGMENT) . "<br>";
18
19     ?>

```

화면 출력

```

Array ( [scheme] => http [host] => hostname [port] => 9090 [user] => username
[pass] => password [path] => /path [query] => arg=value [fragment] => anchor )
PHP_URL_SCHEME = http
PHP_URL_USER =username
PHP_URL_PASS =password
PHP_URL_HOST =hostname
PHP_URL_PORT =9090
PHP_URL_PATH =/path
PHP_URL_QUERY =arg=value
PHP_URL_FRAGMENT =anchor

```

| 내장 함수 |

```

string http_build_query ( mixed $query_data [, string $numeric_prefix [, string $arg_
separator [, int $enc_type = PHP_QUERY_RFC1738 ]]] )

```

내장 함수 `http_build_query()`는 URL 인코딩 쿼리 문자열을 생성합니다.

예제 파일 | [http_build_query.php](#)

```

1  <?php
2      $data = array(
3          'foo' => 'bar',
4          'baz' => 'boom',
5          'cow' => 'milk',
6          'php' => 'hypertext processor'
7      );
8

```



```

9      echo http_build_query($data) . "<br>";
10     echo http_build_query($data, ' ', '&');
11
12  ?>

```

화면 출력

```

foo=bar&baz=boom&cow=milk&php=hypertext+processor
foo=bar&baz=boom&cow=milk&php=hypertext+processor

```

09.3 암호화

암호화는 URL의 쿼리스트링 작성 시 발생할 수 있는 오류나 다른 사용자에게 쉽게 url 값이 노출되는 것을 방지할 때 사용합니다. url 암호화는 인코딩 작업과 디코딩 작업의 한 쌍으로 동작합니다.

09.3.1 URL 인코딩

URL 주소는 불특정 다수의 사람들에게 오픈되어 있습니다. 오픈되어 있는 URL 주소는 쉽게 읽을 수 있습니다. PHP는 URL에 대해서 인코딩 작업을 할 수 있는 전용 함수를 제공합니다.

| 내장 함수 |

```
string urlencode ( string $str )
```

내장 함수 urlencode()는 주어진 url 문자열을 인코딩하여 반환합니다.

| 내장 함수 |

```
string urldecode ( string $str )
```

내장 함수 `urldecode()`는 인코딩된 url을 디코딩합니다.

예제 파일 | `url-02.php`

```
1  <?php
2      $url = "http://www.hojin.io/index.php";
3      $encode = urlencode($url);
4      echo "인코딩 = ". $encode."<br>";
5      echo "디코드 = ". urldecode($encode)."<br>";
6
7      $encode2 = urlencode("?aaa=1&bbb=2");
8      echo "인코딩 = ". $url.$encode2."<br>";
9      echo "디코드 = ". $url.urldecode($encode2)."<br>";
10
11  ?>
```

화면 출력

인코딩 = `http%3A%2F%2Fwww.hojin.io%2Findex.php`

디코드 = `http://www.hojin.io/index.php`

인코딩 = `http://www.hojin.io/index.php%3Faaa%3D1%26bbb%3D2`

디코드 = `http://www.hojin.io/index.php?aaa=1&bbb=2`

09.3.2 base64 URL 처리

PHP는 base64 기반으로 URL을 인코딩, 디코딩하여 처리할 수 있습니다. base64와 관련된 전용 함수들을 제공합니다.

| 내장 함수 |

```
string base64_encode ( string $data )
```

내장 함수 `base64_encode()`는 base64 방식으로 인코딩합니다.

| 내장 함수 |

```
string base64_decode ( string $data [, bool $strict = false ] )
```

내장 함수 `base64_decode()`는 base64 방식으로 인코딩합니다.

예제 파일 | [url-03.php](#)

```
1  <?php
2      $str = "안녕하세요! 지니PHP입니다.";
3      $encode = base64_encode($str);
4      echo "인코딩 = ". $encode . "<br>";
5
6      $decode = base64_decode($encode);
7      echo "디코딩 = ". $decode . "<br>";
8
9  ?>
```

화면 출력

인코딩 = 7JWI64WV7ZWY7IS47JqUISDsp4Dri4hQSFDsnoXri4jri6Qu

디코딩 = 안녕하세요! 지니PHP입니다.

09.3.3 RFC1738 URL 처리

PHP는 RFC1738 기반으로 URL을 인코딩, 디코딩하여 처리할 수 있습니다. RFC1738와 관련된 전용 함수들을 제공합니다.

| 내장 함수 |

```
string rawurlencode ( string $str )
```

내장 함수 `rawurlencode()`는 RFC1738 형식에 따라서 URL을 인코딩합니다.

| 내장 함수 |

```
string rawurldecode ( string $str )
```

내장 함수 `rawurldecode()`는 인코딩된 url 문자열을 디코딩 처리합니다.

예제 파일 | **string-27.php**

```
1  <?php
2      $url="http://www.hojin.io/?name=jiny&country=korea";
3      echo "URL = " . $url . "<br>";
4
5      $urlEncode = rawurlencode($url);
6      echo "인코딩 : " . $urlEncode . "<br>";
7
8      $urlDecode = rawurldecode($urlEncode);
9      echo "디코딩 : " . $urlDecode . "<br>";
10
11  ?>
```

화면 출력

URL = http://www.hojin.io/?name=jiny&country=korea
인코딩 : http%3A%2F%2Fwww.hojin.io%2F%3Fname%3Djiny%26country%3Dkorea
디코딩 : http://www.hojin.io/?name=jiny&country=korea

09.4 호스트 정보

PHP는 접속된 단말기의 정보들을 읽어서 처리할 수 있습니다. 시스템을 운영하면서 로그 기록, 통계 등을 위해서 클라이언트 단말기의 접속 정보 등을 처리할 수 있습니다.

09.4.1 IP 주소 확인

내장 함수 `gethostbyname()`은 지정한 인터넷 호스트 이름에 대하여 IPv4 주소 정보를 가져옵니다.

| 내장 함수 |

string **gethostbyname** (string \$hostname)

예제 파일 | **gethostbyname.php**

```
1  <?php
2      $ip = gethostbyname('www.hojin.io');
3      echo $ip;
4
5  ?>
```

화면 출력 93.184.216.34

| 내장 함수 |

array **gethostbyname_l** (string \$hostname)

내장 함수 `gethostbyname_l()`은 지정된 인터넷 호스트 이름에 해당하는 IPv4 주소 목록을 가져옵니다.

예제 파일 | **gethostbyname_l.php**

```
1  <?php
2      $hosts = gethostbyname_l('www.hojin.io');
3      print_r($hosts);
4
5  ?>
```

화면 출력

Array ([0] => 93.184.216.34)

09.4.2 호스트명

내장 함수 `gethostbyaddr()`은 `ip_address`로 지정된 인터넷 호스트의 호스트 이름을 반

환합니다.

| 내장 함수 |

```
string gethostbyaddr ( string $ip_address )
```

예제 파일 | **gethostbyaddr.php**

```
1  <?php
2      $hostname = gethostbyaddr($_SERVER['REMOTE_ADDR']);
3      echo $hostname;
4
5  ?>
```

| 내장 함수 |

```
string gethostname ( void )
```

내장 함수 `gethostname()`은 호스트 이름을 가져옵니다.

예제 파일 | **gethostname.php**

```
1  <?php
2      // 호스트 이름을 출력합니다.
3      echo gethostname();
4
5  ?>
```

화면 출력

ns.dojangshop.com

09.4.3 포트

내장 함수 `getservbyname()`은 인터넷 서비스 및 프로토콜과 관련된 포트 번호를 가져옵니다.

| 내장 함수 |

```
int getservbyname ( string $service , string $protocol )
```

예제 파일 | **getservbyname.php**

```
1  <?php
2      $services = array('http', 'ftp', 'ssh', 'telnet', 'imap',
3          'smtp', 'nickname', 'gopher', 'finger', 'pop3', 'www');
4
5      foreach ($services as $service) {
6          $port = getservbyname($service, 'tcp');
7          echo $service . ": " . $port . "<br />\n";
8      }
9
10 ?>
```

화면 출력

```
http: 80
ftp: 21
ssh: 22
telnet: 23
imap: 143
smtp: 25
nickname: 43
gopher: 70
finger: 79
pop3: 110
www: 80
```

09.5 DNS 정보

PHP는 DNS 정보를 확인할 수 있는 몇 개의 내장 함수를 지원합니다.

| 내장 함수 |

```
bool checkdnsrr ( string $host [, string $type = "MX" ] )
```

내장 함수 `checkdnsrr()`은 주어진 인터넷 호스트 이름 또는 IP 주소에 해당하는 DNS 레코드를 확인하면 됩니다.

예제 파일 | `checkdnsrr.php`

```
1  <?php
2      $email = "infohojin@naver.com";
3      $exp = "[a-z\ '0-9]+([._-][a-z\ '0-9]+)*@([a-z0-9]+([._-][a-z0-9]+)+)$";
4
5      if (ereg($exp,$email)) {
6          $emailAddr = explode("@",$email);
7          if (checkdnsrr($emailAddr[1],"MX")) {
8              echo "DNS 레코드 확인";
9          } else {
10             echo "DNS 레코드 확인 불가";
11         }
12     } else {
13         echo "이메일 형식이 아닙니다.";
14     }
15 }
16
17 ?>
```

화면 출력

DNS 레코드 확인

| 내장 함수 |

```
array dns_get_record ( string $hostname [, int $type = DNS_ANY [, array &$authns [, array &$addtl [, bool $raw = false ]]] ] )
```


내장 함수 `dns_get_record()`는 호스트 이름과 관련된 DNS 리소스 레코드를 가져옵니다.

예제 파일 | `dns_get_record.php`

```
1 <?php
2     $result = dns_get_record("php.net");
3     print_r($result);
4
5 ?>
```

화면 출력

```
Array ( [0] => Array ( [host] => php.net [type] => AAAA [ipv6] => 2a02:cb41::7
[class] => IN [ttl] => 300 ) [1] => Array ( [host] => php.net [type] => A
[ip] => 72.52.91.14 [class] => IN [ttl] => 300 ) [2] => Array ( [host] =>
php.net [type] => TXT [txt] => v=spf1 ip4:72.52.91.12 ip6:2a02:cb41::8
ip4:140.211.15.143 ?all [entries] => Array ( [0] => v=spf1 ip4:72.52.91.12
ip6:2a02:cb41::8 ip4:140.211.15.143 ?all ) [class] => IN [ttl] => 300 ) [3] =>
Array ( [host] => php.net [type] => MX [pri] => 0 [target] => php-smtp2.php.
net [class] => IN [ttl] => 30 ) [4] => Array ( [host] => php.net [type] => SOA
[mname] => ns1.php.net [rname] => admin.easydns.com [serial] => 1484930803
[refresh] => 16384 [retry] => 2048 [expire] => 1048576 [minimum-ttl] => 2560
[class] => IN [ttl] => 300 ) [5] => Array ( [host] => php.net [type] => NS
[target] => dns2.easydns.net [class] => IN [ttl] => 300 ) [6] => Array ( [host]
=> php.net [type] => NS [target] => dns3.easydns.org [class] => IN [ttl] =>
300 ) [7] => Array ( [host] => php.net [type] => NS [target] => dns1.easydns.
com [class] => IN [ttl] => 300 ) [8] => Array ( [host] => php.net [type] => NS
[target] => dns4.easydns.info [class] => IN [ttl] => 300 ) )
```

| 내장 함수 |

```
bool getmxrr ( string $hostname , array &$mxhosts [, array &$weight ] )
```

내장 함수 `getmxrr()`은 지정된 인터넷 호스트 이름에 해당하는 MX 레코드를 체크합니다.

아파치 함수

PHP는 웹 서버와 연동하여 웹 서비스를 제공합니다. 대표적인 웹 서버로는 아파치 웹 서버가 있습니다.

PHP에서는 아파치 웹 서버의 정보를 읽어서 처리할 수 있는 몇 가지 함수들을 제공합니다.

| 내장 함수 |

```
string apache_get_version ( void )
```

내장 함수 `apache_get_version()`은 아파치 버전을 확인합니다.

예제 파일 | `apache_get_version.php`

```
1  <?php
2      $version = apache_get_version();
3      echo "$version";
4  ?>
```

화면 출력

Apache/2.2.29 (Unix) mod_ssl/2.2.29 OpenSSL/1.0.1e-fips DAV/2 PHP/5.3.29

| 내장 함수 |

array **apache_get_modules** (void)

내장 함수 `apache_get_modules()`는 적재된 아파치 모듈 목록을 얻습니다

예제 파일 | **apache_get_modules.php**

```
1  <?php
2      print_r(apache_get_modules());
3
4  ?>
```

화면 출력

Array ([0] => core [1] => prefork [2] => http_core [3] => mod_so [4] => mod_authn_file [5] => mod_authn_dbm [6] => mod_authn_anon [7] => mod_authn_dbd [8] => mod_authn_default [9] => mod_authz_host [10] => mod_authz_groupfile [11] => mod_authz_user [12] => mod_authz_dbm [13] => mod_authz_owner [14] => mod_authz_default [15] => mod_auth_basic [16] => mod_auth_digest [17] => mod_dbd [18] => mod_dumpio [19] => mod_reqtimeout [20] => mod_ext_filter [21] => mod_include [22] => mod_filter [23] => mod_substitute [24] => mod_deflate [25] => mod_log_config [26] => mod_logio [27] => mod_env [28] => mod_expires [29] => mod_headers [30] => mod_ident [31] => mod_setenvif [32] => mod_version [33] => mod_ssl [34] => mod_mime [35] => mod_dav [36] => mod_status [37] => mod_autoindex [38] => mod_asis [39] => mod_info [40] => mod_cgi [41] => mod_dav_fs [42] => mod_vhost_alias [43] => mod_negotiation [44] => mod_dir [45] => mod_imagemap [46] => mod_actions [47] => mod_speling [48] => mod_userdir [49] => mod_alias [50] => mod_rewrite [51] => mod_php5 [52] => mod_ruid2)

| 내장 함수 |

string **apache_getenv** (string \$variable [, bool \$walk_to_top])

내장 함수 `apache_getenv()`는 아파치 서브 프로세스의 환경 변수를 가져옵니다. 아파치 2 버전 이상에서만 지원합니다.

예제 파일 | `apache_getenv.php`

```
1  <?php
2      // 아파치 환경 변수 SERVER_ADDR을 가져오는 방법.
3      $ret = apache_getenv("SERVER_ADDR");
4      echo $ret;
5
6  ?>
```

화면 출력

211.110.1.195

| 내장 함수 |

`bool apache_setenv (string $variable , string $value [, bool $walk_to_top = false])`

내장 함수 `apache_setenv()`는 아파치의 서브 프로세스의 환경 변수를 설정합니다. 하지만 아파치 환경 변수를 설정할 때 해당하는 `$_SERVER` 변수는 바뀌지 않습니다.

예제 파일 | `apache_setenv.php`

```
1  <?php
2      apache_setenv("EXAMPLE_VAR", "Example Value");
3
4      $ret = apache_getenv("EXAMPLE_VAR");
5      echo $ret;
6
7  ?>
```

화면 출력

Example Value

| 내장 함수 |

object **apache_lookup_uri** (string \$filename)

내장 함수 `apache_lookup_uri()`는 지정한 URI에 부분 요청을 수행하고 그에 대한 정보를 반환합니다.

예제 파일 | **apache_lookup_uri.php**

```
1  <?php
2      $info = apache_lookup_uri('apache_getenv.php');
3      print_r($info);
4
5      if (file_exists($info->filename)) {
6          echo 'file exists!';
7      }
8
9  ?>
```

화면 출력

```
stdClass Object ( [status] => 200 [the_request] => GET /jinysrc/apache_
lookup_uri.php HTTP/1.1 [method] => GET [mtime] => 0 [clength] => 0 [chunked]
=> 0 [content_type] => application/x-httpd-php [no_cache] => 0 [no_local_
copy] => 1 [unparsed_uri] => /jinysrc/apache_getenv.php [uri] => /jinysrc/
apache_getenv.php [filename] => /home/www/jinysrc/apache_getenv.php [allowed]
=> 0 [sent_bodyct] => 0 [bytes_sent] => 0 [request_time] => 1502701432 )
file exists!
```

| 내장 함수 |

string **apache_note** (string \$note_name [, string \$note_value])

내장 함수 `apache_note()`는 아파치 모듈 간의 통신입니다. 아파치의 요청 노트를 얻거나 설정합니다.

| 내장 함수 |

array **apache_request_headers** (void)

내장 함수 `apache_request_headers()`는 모든 HTTP 요청 헤더를 가져옵니다.

예제 파일 | **apache_request_headers.php**

```
1  <?php
2      $headers = apache_request_headers();
3
4      foreach ($headers as $header => $value) {
5          echo "$header: $value <br />\n";
6      }
7
8  ?>
```

화면 출력

```
Accept: text/html, application/xhtml+xml, image/jxr, */*
Accept-Language: ko
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like
Gecko
Accept-Encoding: gzip, deflate
Host: shop.jinyweb.com
Connection: Keep-Alive
Cookie: PHPSESSID=t48nid9kddv90e0cfps68oc663
```

| 내장 함수 |

array **apache_response_headers** (void)

내장 함수 `apache_response_headers()`는 모든 HTTP 응답 헤더를 가져옵니다.

예제 파일 | **apache_response_headers.php**

```
1  <?php
2      print_r(apache_response_headers());
3  ?>
```

화면 출력

Array ([X-Powered-By] => PHP/5.3.29)

| 내장 함수 |

bool **virtual** (string \$filename)

내장 함수 `virtual()`은 아파치 하위 요청을 처리합니다. 하위 요청을 실행할 때 헤더를 포함한 모든 버퍼를 종료하고 브라우저에 출력합니다.

`virtual()`은 `mod_include`의 `<!--#include virtual...-->`과 동일한 처리를 하는 아파치 함수입니다. 아파치 모듈로 설치했을 때만 지원합니다.

예제 파일 | **virtual.php**

```
1  <?PHP
2      $file = '../images/logo.jpg';
3      $file_info = apache_lookup_uri($file);
4      header('content-type: ' . $file_info -> content_type);
5      virtual($file);
6      die();
7
8  ?>
```

| 내장 함수 |

bool **apache_reset_timeout** (void)

내장 함수 `apache_reset_timeout()`은 아파치의 기본 동작 타이머는 300초입니다. 이 타이머 값을 초기화합니다.

`set_time_limit(0); ignore_user_abort(true)`와 같은 함수 또는 주기적으로 `apache_reset_timeout()` 함수를 호출하면 이론적으로 아파치를 무한 실행할 수 있습니다.

| 내장 함수 |

```
bool apache_child_terminate ( void )
```

내장 함수 `apache_child_terminate()`는 아파치 프로세스를 종료합니다. 보통 PHP 스크립트는 실행이 끝났을 때 자동으로 프로세스를 종료하기 위해서 아파치 프로세스 목록에 등록합니다.

스크립트에서 사용한 메모리는 내부적으로 재사용할 수 있도록 해제되지만, 한 번 사용된 메모리는 운영체제로는 반환하지 않습니다. 메모리를 많이 사용한 스크립트의 경우 본 함수를 통하여 스크립트를 종료할 수 있습니다.

11

통신

11.1 소켓 통신

소켓 통신이란 네트워크상에서 여러 다른 컴퓨터들의 프로세스 간의 통신 채널을 말합니다.

| 내장 함수 |

```
resource fsocketopen ( string $hostname [, int $port = -1 [, int &$errno [, string &$errstr [, float $timeout = ini_get("default_socket_timeout") ]]] )
```

내장 함수 `fsocketopen()`은 유닉스/리눅스 기반의 도메인 소켓을 엽니다. 도메인 서버와 포트를 응용하여 파일 포인트 형태로 반환합니다. 또한 접속 타임아웃도 함께 설정할 수 있습니다.

`fsocketopen()` 함수를 통하여 엽힌 포인터는 `fgets()`, `fgetss()`, `fputs()`, `fclose()`, `feof()` 등과 같이 파일 처리하는 것과 유사하게 사용할 수 있습니다.

예제 파일 | **fsockopen.php**

```
1  <?php
2      $fp = fsockopen("www.example.com", 80, $errno, $errstr, 30);
3      if (!$fp) {
4          echo "errstr ($errno)<br />\n";
5      } else {
6          $out = "GET / HTTP/1.1\r\n";
7          $out .= "Host: www.example.com\r\n";
8          $out .= "Connection: Close\r\n\r\n";
9          fwrite($fp, $out);
10
11         while (!feof($fp)) {
12             echo fgets($fp, 128);
13         }
14         fclose($fp);
15     }
16
17  ?>
```

| 내장 함수 |

resource pfsockopen (string \$hostname [, int \$port = -1 [, int &\$amp;errno [, string &\$amp;errstr [, float \$timeout = ini_get("default_socket_timeout")]]]])

내장 함수 pfsockopen()은 지속적인 인터넷 또는 Unix 도메인 소켓을 연결합니다.

예제 파일 | **pfsockopen.php**

```
1  <?php
2
3      $host = gethostbyaddr($_SERVER['REMOTE_ADDR']);
4
5      $host = 'www.example.com';
6      $service_uri = '/cgi/proACT';
7      $vars = 'code=23&act=TESTing';
8
9      //# HTTP 요청 헤더 구성
10     $header = "Host: $host\r\n";
```

```

11     $header .= "User-Agent: PHP Script\r\n";
12     $header .= "Content-Type: application/x-www-form-urlencoded\r\n";
13     $header .= "Content-Length: ".strlen($vars)."\r\n";
14     $header .= "Connection: close\r\n\r\n";
15
16     $fp = pfsockopen("ssl://" . $host, 443, $errno, $errstr);
17     if (!$fp) {
18         echo "errstr ($errno)<br/>\n";
19         echo $fp;
20     } else {
21         fputs($fp, "POST $service_uri HTTP/1.1\r\n");
22         fputs($fp, $header.$vars);
23         fwrite($fp, $out);
24
25         while (!feof($fp)) {
26             echo fgets($fp, 128);
27         }
28         fclose($fp);
29     }
30
31     ?>

```

11.2 AJAX

AJAX는 Asynchronous JavaScript XML의 약자입니다. AJAX는 페이지의 재로딩 없이 비동기/동기 방식으로 페이지의 일부분을 갱신할 수 있는 웹 기술입니다.

보통 웹 페이지는 url 주소에 따라 웹 서버에서 페이지를 로딩하여 화면을 출력합니다. 특정한 페이지로 이동하거나, FROM 요소 등을 통하여 데이터를 입력 시 url 주소가 변경이 됩니다. 이때 브라우저는 페이지를 다시 로딩하여 화면을 재구성하게 됩니다.

PHP와 같이 자바스크립트의 AJAX 방식을 혼용하여 사용할 수 있습니다. url로 페이지 전체를 다시 로딩하는 방식이 아닌 페이지 일부분만 로딩하여 내용을 변경할 수 있습니다.

AJAX는 인터넷 표준을 기반을 따릅니다.

- XMLHttpRequest object(서버와 비동기 방식으로 데이터 교환)
- JavaScript/DOM(정보 표시 및 상호작용)
- CSS
- XML

11.2.1 Javascript & JQUERY

AJAX를 사용하기 위해서는 자바스크립트의 XMLHttpRequest 기능을 사용해야 합니다. 자바스크립트의 XMLHttpRequest 코드들은 코드가 복잡하고 브라우저마다 상호 특성에 따라 다르게 동작할 수 있습니다.

요즘 들어 AJAX를 쉽게 처리하기 위해서 자바스크립트 라이브러리 등을 많이 이용하여 사용합니다. 대표적으로 JQuery가 인기가 높습니다. JQuery는 복잡한 자바스크립트의 코드들을 쉽게 처리할 수 있으며 AJAX 통신 코드 또한 포함되어 있습니다(JQuery는 <https://jquery.com/> 사이트를 참조). 또는 AJAX용으로 간결하게 제작된 AXIOS라는 라이브러리도 있습니다.

11.2.2 Javascript 예제

AJAX를 처리하기 위해서는 2개의 스크립트 파일이 필요로 합니다. 첫 번째 스크립트는 브라우저 접속 시 보여지는 기본 화면입니다. 두 번째 스크립트는 어떠한 동작이 있을 경우 AJAX로 처리되는 스크립트입니다.

예제 파일 | [ajax.php](#)

```
1 <html>
2 <head>
3   <script>
4     function ajax(str) {
5       if (str.length == 0) {
6         document.getElementById("txtHint").innerHTML = "";
7         return;
```

```

8         } else {
9             var xmlhttp = new XMLHttpRequest();
10            xmlhttp.onreadystatechange = function() {
11                if (this.readyState == 4 && this.status == 200) {
12                    document.getElementById("txtHint").innerHTML = this.
                        responseText;
13                }
14            };
15            xmlhttp.open("GET", "ajax_email.php?q=" + str, true);
16            xmlhttp.send();
17        }
18    }
19    </script>
20 </head>
21
22 <body>
23
24     <p><b>아래 이메일 주소를 입력해 주세요:</b></p>
25     <form>
26         email: <input type="text" onkeyup="ajax(this.value)">
27     </form>
28     <p>확인: <span id="txtHint"></span></p>
29 </body>
30 </html>

```

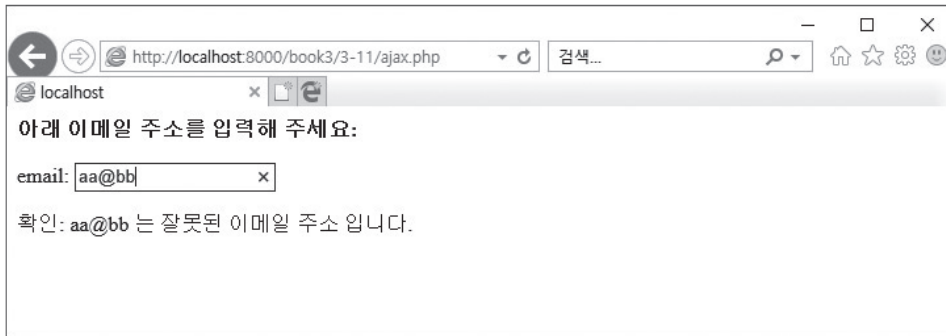
예제 파일 | [ajax_email.php](#)

```

1  <?php
2      $email = $_GET['q'];
3
4      if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
5          echo "$email 정상적인 이메일 주소입니다.";
6      } else {
7          echo "$email 는 잘못된 이메일 주소입니다.";
8      }
9
10  ?>

```

화면 출력



위의 예제는 전형적인 자바스크립트의 XMLHttpRequest 함수를 이용한 AJAX 처리입니다. 먼저 ajax.php로 접속 시 이메일 주소를 입력하는 폼이 하나 출력됩니다.

input 필드에 키를 한 글자씩 입력할 때마다 ajax_email.php 파일을 비동기 AJAX로 호출하여 이메일의 유효성을 체크합니다. 입력되는 이메일 문자는 GET 방식으로 전달합니다.

출력된 문자열 메시지를 자바스크립트의,

```
document.getElementById("txtHint").innerHTML = this.responseText;
```

처리로 DOM 내용을 갱신하여 출력합니다.

11.3 JQuery 예제

JQuery 라이브러리를 이용하면 좀 더 쉽게 AJAX를 처리할 수 있습니다. 다음 예제는 JQuery와 POST 방식의 전송 처리입니다.

JQuery를 사용하기 위해서는 라이브러리 파일을 스크립트로 삽입해야 합니다. 여기서는 CDN 방식으로 삽입해 보도록 하겠습니다.

예제 파일 | [ajax2.php](#)

```

1  <html>
2  <head>
3      <!-- JQuery CDN -->
4      <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/
        jquery.min.js"></script>
5
6      <script>
7          $(document).ready(function() {
8              $("#email").keydown( function() {
9                  // AJAX 호출
10                 $.ajax({
11                     url:'ajax_post.php',
12                     type:'post',
13                     data:$('form').serialize(),
14                     success:function(data) {
15                         $('#txtHint').html(data);
16                     }
17                 });
18             });
19
20         });
21
22     </script>
23 </head>
24
25 <body>
26     <p><b>아래 이메일 주소를 입력해 주세요:</b></p>
27     <form name='login' method='post' enctype='multipart/form-data'>
28         email: <input type="text" name="email" id="email">
29     </form>
30     <p>확인: <span id="txtHint"></span></p>
31 </body>
32 </html>

```

예제 파일 | [ajax_post.php](#)

```

1  <?php
2      $email = $_POST['email'];

```

```
3
4     if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
5         echo "$email 정상적인 이메일 주소입니다.";
6     } else {
7         echo "$email 는 잘못된 이메일 주소입니다.";
8     }
9
10  ?>
```

12

cURL

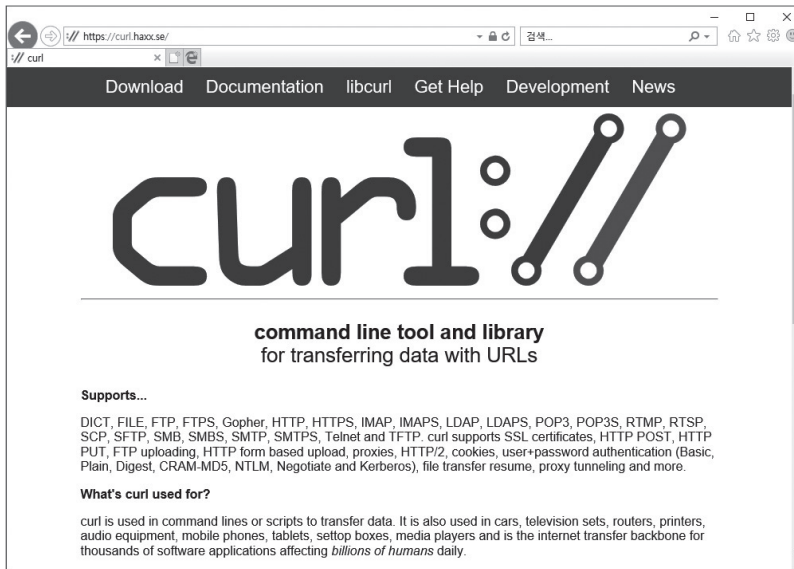
보통 웹 페이지들은 브라우저를 통하여 접속합니다. 또한 브라우저는 전송받은 HTML을 분석하고 자바스크립트를 실행한 결과를 화면에 그래픽 처리 결과로 보여줍니다.

cURL은 client URL의 약자입니다. cURL은 command line tool로 콘솔 창을 통하여 웹 사이트에 접속하고 결과를 받아올 수 있습니다. cURL은 다양한 프로토콜을 지원합니다. 그중 많이 사용하는 프로토콜로는 HTTP, FTP 등이 있습니다.

웹 URL을 접근할 때 fopen() 함수 등을 통하여 접근도 가능하나, 보안적인 측면 때문에 접속을 제한합니다. 하지만 cURL은 PHP의 allow_url_fopen 옵션 설정과 상관없이 동작합니다. 또한 Proxy, Cookie, Header를 쉽게 설정할 수 있습니다.

12.1 설치

cURL의 모듈, 라이브러리 및 설치 파일들은 공식 사이트 <https://curl.haxx.se/>에서 다운로드할 수 있습니다.



12.1.1 모듈 설정

cURL 확장 모듈이 설정되어 있지 않는다면 php.ini 파일을 직접 수정해야 합니다.

확장 모듈의 설치 경로를 php가 설치된 디렉터리로 변경, 주석을 해제합니다.

; On windows:

extension_dir = "C:\wphp-7.1.4-Win32-VC14-x86\ext"

확장 모듈 부분에서 cURL 모듈 부분을 설정, 주석을 해제합니다.

extension=php_curl.dll

php.ini를 변경 후에 내장 서버 또는 아파치 서버를 재시작하면 됩니다.

12.1.2 모듈 확인

PHP는 코드 소스상에서 cURL 기능을 사용할 수 있는 내장 함수들을 확장 기능으로 제공 합니다. 확장 기능이 설치되어 있는지 확인하는 방법은 콘솔상에서 명령어를 입력해 보면 쉽게 알 수 있습니다.

#] php --re curl

또는 phpinfo() 함수를 통해서 확인할 수 있습니다.

curl	
cURL support	enabled
cURL Information	7.53.1
Age	3
Features	
AsynchDNS	Yes
CharConv	No
Debug	No
GSS-Negotiate	No
IDN	Yes
IPv6	Yes
krb4	No
Largefile	Yes
libz	Yes
NTLM	Yes
NTLMWB	No
SPNEGO	Yes
SSL	Yes
SSPI	Yes
TLS-SRP	No
HTTP2	Yes
GSSAPI	No
KERBEROS5	Yes
UNIX_SOCKETS	No
PSL	No
Protocols	dict, file, ftp, ftps, gopher, http, https, imap, imaps, ldap, pop3, pop3s, rtsp, scp, sftp, smtp, smtps, telnet, tftp
Host	i386-pc-win32
SSL Version	OpenSSL/1.0.2k
ZLib Version	1.2.8
libSSH Version	libssh2/1.8.0

12.1.3 모듈 체크

프로그램 실행 시 cURL 미설치로 발생할 수 있는 오류를 사전에 방지하기 위하여 소스상에서 확장 모듈을 확인할 수 있습니다.

예제 파일 | [curl.php](#)

```
1  <?php
2      if (extension_loaded("curl")) {
3          echo "cURL extension is Loaded";
4      } else {
5          echo "cURL extension is not available";
6      }
7  ?>
```

화면 출력

cURL extension is Loaded

12.2 기본 동작

cURL 확장 모듈이 잘 설정되었다면 PHP cURL 함수들을 사용할 수 있습니다.

12.2.1 초기화

cURL을 사용하기 위해서는 먼저 cURL 세션의 초기화가 필요합니다. 세션 초기화를 위해서는 `curl_init()`를 사용합니다.

| 내장 함수 |

```
resource curl_init ([ string $url = NULL ] )
```

```
$ch = curl_init();
```

위와 같이 세션의 초기화 후에 `curl_setopt()`, `curl_exec()`를 통하여 cURL 세션을 실행할 수 있습니다. 모든 cURL 세션 후에는 `curl_close()`를 통하여 세션을 종료합니다.

예제 파일 | `curl_basic.php`

```
1  <?php
2
3      $ch = curl_init("http://www.example.com/");
4      $fp = fopen("example.txt", "w");
5
6      curl_setopt($ch, CURLOPT_FILE, $fp);
7      curl_setopt($ch, CURLOPT_HEADER, 0);
8
9      curl_exec($ch);
10     curl_close($ch);
11     fclose($fp);
```

```
12
13 ?>
```

위의 예제는 지정한 웹 사이트 URL로 접속하여 HTML 페이지를 받아 지정한 파일로 저장하는 예제입니다.

12.2.2 전송 옵션

| 내장 함수 |

```
bool curl_setopt ( resource $ch , int $option , mixed $value )
```

내장 함수 `curl_setopt()`는 cURL 전송을 위한 옵션을 설정합니다. 옵션은 `CURLOPT_***` 형태의 상수값으로 되어 있습니다.

| 내장 함수 |

```
bool curl_setopt_array ( resource $ch , array $options )
```

내장 함수 `curl_setopt_array()`는 cURL 전송을 위한 옵션을 배열로 설정할 수 있습니다.

```
$options = array(
    CURLOPT_URL => 'http://www.example.com/',
    CURLOPT_HEADER => false
);

curl_setopt_array($ch, $options);
```

`curl_setopt_array()` 함수를 사용 시 앞의 `curl_setopt()` 함수를 통하여 여러 번 호출하지 않고 옵션을 배열 그룹으로 전달할 수 있습니다.

| 내장 함수 |

```
void curl_reset ( resource $ch )
```

내장 함수 `curl_reset()`은 설정한 cURL 세션의 설정값을 리셋합니다.

```
$ch = curl_init();  
  
curl_setopt($ch, CURLOPT_URL, 'http://aaa.com/');  
  
// 옵션 설정을 모두 리셋  
curl_reset($ch);
```

12.2.3 전송 실행

설정한 cURL 세션 값 등을 실행하여 통신을 처리합니다. cURL 세션을 실행하기 위해서는 내장 함수 `curl_exec()`를 사용합니다.

| 내장 함수 |

```
mixed curl_exec ( resource $ch )
```

실행 성공 시 결과값을 반환합니다. 반환된 값을 받아 처리합니다.

```
$response = curl_exec($ch);
```

12.2.4 세션 종료

cURL 생성과 실행이 완료된 경우에는 세션을 종료합니다. 내장 함수 `curl_close()`는 cURL 세션을 종료합니다.

| 내장 함수 |

```
void curl_close ( resource $ch )
```

```
curl_close($ch);
```

12.3 공유 핸들

공유 핸들은 다수의 cURL 핸들에게 동일한 설정값을 적용한 핸들을 이용하여 설정값을 지정할 수 있는 기능입니다.

| 내장 함수 |

```
resource curl_share_init ( void )
```

내장 함수 `curl_share_init()`는 공유 cURL 세션 핸들을 초기화합니다.

| 내장 함수 |

```
bool curl_share_setopt ( resource $sh , int $option , string $value )
```

내장 함수 `curl_share_setopt()`는 공유 cURL 핸들에 옵션값을 설정합니다.

| 내장 함수 |

```
void curl_share_close ( resource $sh )
```

내장 함수 `curl_share_close()`는 공유 cURL 핸들을 종료합니다.

예제 파일 | `curl_share.php`

```
1  <?php
2      // cURL 공유 핸들을 초기화합니다.
3      // 쿠키 값과 같은 공유 옵션을 설정합니다.
4      $sh = curl_share_init();
5      curl_share_setopt($sh, CURLSHOPT_SHARE, CURL_LOCK_DATA_COOKIE);
6
7      $ch1 = curl_init("http://example.com/");
8      // 공유 핸들의 옵션을 통하여 ch1의 curl 핸들 옵션을 설정합니다.
9      curl_setopt($ch1, CURLOPT_SHARE, $sh);
10     curl_exec($ch1);
11
12     $ch2 = curl_init("http://php.net/");
13     // 공유 핸들의 옵션을 통하여 ch2의 curl 핸들 옵션을 설정합니다.
14     curl_setopt($ch2, CURLOPT_SHARE, $sh);
15     curl_exec($ch2);
16
17     // 공유 핸들을 종료합니다.
18     curl_share_close($sh);
19
20     curl_close($ch1);
21     curl_close($ch2);
22
23  ?>
```

12.4 멀티 핸들

멀티 CURL 처리를 위한 함수들을 지원합니다. 멀티 핸들은 1개의 작업별로 작업하던 cURL 작업 대신에 여러 개를 작업할 수 있도록 설정하고 실행하는 방법입니다.

| 내장 함수 |

```
resource curl_multi_init ( void )
```

내장 함수 `curl_multi_init()`는 멀티 핸들을 초기화합니다. 멀티 핸들은 여러 개의 cURL

핸들을 비동기 방식으로 처리할 수 있습니다.

| 내장 함수 |

```
int curl_multi_add_handle ( resource $mh , resource $ch )
```

내장 함수 `curl_multi_add_handle()`은 단일 cURL 핸들을 멀티 cURL 핸들에 추가합니다.

| 내장 함수 |

```
int curl_multi_exec ( resource $mh , int &$still_running )
```

내장 함수 `curl_multi_exec()`은 멀티 cURL 핸들에 속해 있는 하위 핸들을 실행합니다. 첫 번째 인자는 멀티 핸들, 두 번째 인자는 작업이 실행 도중인지 여부를 알려주는 플래그 참조 값입니다.

| 내장 함수 |

```
int curl_multi_select ( resource $mh [, float $timeout = 1.0 ] )
```

내장 함수 `curl_multi_remove_handle()`은 멀티 핸들에서 단일 핸들을 제거합니다.

| 내장 함수 |

```
void curl_multi_close ( resource $mh )
```

내장 함수 `curl_multi_close()`은 멀티 cURL 핸들을 종료합니다.

예제 파일 | [curl_multi.php](#)

```
1 <?php
```

```

2
3 // cURL 단일 핸들을 초기화합니다.
4 $ch1 = curl_init();
5 // 각각의 핸들에 옵션값을 설정합니다.
6 $url1 = "http://www.example.com/";
7 curl_setopt($ch1, CURLOPT_URL, $url1);
8 curl_setopt($ch1, CURLOPT_HEADER, 0);
9
10 // cURL 단일 핸들을 초기화합니다.
11 $ch2 = curl_init();
12 // 각각의 핸들에 옵션값을 설정합니다.
13 $url2 = "http://www.example.com/";
14 curl_setopt($ch2, CURLOPT_URL, $url2 );
15 curl_setopt($ch2, CURLOPT_HEADER, 0);
16
17 //멀티 핸들을 생성합니다.
18 $mh = curl_multi_init();
19
20 //멀티 핸들에 단일 핸들을 추가합니다.
21 curl_multi_add_handle($mh,$ch1);
22 curl_multi_add_handle($mh,$ch2);
23
24 // 멀티 실행 여부 체크 플래그
25 $active = null;
26
27 do {
28     // 멀티 핸들을 실행합니다.
29     $mrc = curl_multi_exec($mh, $active);
30 } while ($mrc == CURLM_CALL_MULTI_PERFORM);
31
32 while ($active && $mrc == CURLM_OK) {
33
34     // curl_multi 연결에 대한 작업을 대기합니다.
35     // 실패 시 -1을 반환
36     if (curl_multi_select($multi) == -1) {
37         // 마이크로 초 지연 실행
38         usleep(1);
39
40     } else {
41         do {
42             // 멀티 핸들을 실행합니다.

```

```

43         $mrc = curl_multi_exec($mh, $active);
44     } while ($mrc == CURLM_CALL_MULTI_PERFORM);
45
46     }
47
48 }
49
50 // 멀티 핸들에서 단일 핸들을 제거합니다.
51 curl_multi_remove_handle($mh, $ch1);
52 curl_multi_remove_handle($mh, $ch2);
53
54 // 멀티 cURL 핸들을 종료합니다.
55 curl_multi_close($mh);
56
57 ?>

```

| 내장 함수 |

string curl_multi_getcontent (resource \$ch)

내장 함수 `curl_multi_getcontent()`는 `CURLOPT_RETURNTRANSFER`가 설정된 경우 cURL 핸들의 내용을 반환합니다.

`CURLOPT_RETURNTRANSFER`가 특정 핸들에 대해 설정된 옵션 경우에 cURL 핸들의 내용을 문자열 형식으로 반환합니다.

예제 파일 | **curl_multi_getcontent.php**

```

1  <?php
2
3      $ch = curl_init('http://www.example.com/');
4      curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
5
6      // curl_exec()로 실행된 핸들에 curl_multi_getcontent()를 사용할 수 있습니다.
7      $result = curl_exec($ch);
8
9      // curl_multi_getcontent()는 curl_exec()와 같은 결과값을 반환합니다.

```

```

10     $content = curl_multi_getcontent($ch);
11
12     var_dump($result === $content);
13     echo $content;
14
15     curl_close($ch);
16
17     ?>

```

| 내장 함수 |

```
bool curl_multi_setopt ( resource $mh , int $option , mixed $value )
```

내장 함수 `curl_multi_setopt()`는 cURL 멀티 핸들에 대한 옵션을 설정합니다.

| 내장 함수 |

```
array curl_multi_info_read ( resource $mh [, int &$msgs_in_queue = NULL ] )
```

내장 함수 `curl_multi_info_read()`는 현재 전송의 정보를 읽어옵니다.

각각의 전송에서 메시지 또는 정보가 있는 경우에는 멀티 핸들로 요청해야 합니다. 메시지는 전송 오류 코드 또는 전송이 완료되었다는 정보만 표시됩니다.

만일 이 함수를 반복적으로 호출할 때는 매번 새로운 다른 결과값이 반환될 수 있습니다. FALSE 상태로 더 이상 반환되는 값을 얻을 수 없을 때까지 반복됩니다. `msgs_in_queue`가 가리키는 정수에는 이 함수가 호출 된 이후 남은 메시지 수가 포함됩니다.

예제 파일 | [curl_multi_info_read.php](#)

```

1  <?php
2
3      $urls = array(
4          "http://www.exsample.com/",
5          "http://www.php.net/"

```

```

6     );
7
8     $mh = curl_multi_init();
9
10    foreach ($urls as $i => $url) {
11        $conn[$i] = curl_init($url);
12        curl_setopt($conn[$i], CURLOPT_RETURNTRANSFER, 1);
13        curl_multi_add_handle($mh, $conn[$i]);
14    }
15
16    do {
17        $status = curl_multi_exec($mh, $active);
18        $info = curl_multi_info_read($mh);
19        if (false !== $info) {
20            var_dump($info);
21            echo "<br>";
22        }
23    } while ($status === CURLM_CALL_MULTI_PERFORM || $active);
24
25    foreach ($urls as $i => $url) {
26        $res[$i] = curl_multi_getcontent($conn[$i]);
27        curl_close($conn[$i]);
28    }
29
30    var_dump(curl_multi_info_read($mh));
31
32    ?>

```

화면 출력

```

array(3) { ["msg"]=> int(1) ["result"]=> int(0) ["handle"]=> resource(4) of
type (curl) }
array(3) { ["msg"]=> int(1) ["result"]=> int(52) ["handle"]=> resource(3) of
type (curl) }
bool(false)

```

| 내장 함수 |

```
string curl_multi_strerror ( int $errornum )
```

내장 함수 `curl_multi_strerror()`는 오류 코드를 설명하는 텍스트 오류 메시지를 반환합니다.

예제 파일 | `curl_multi_strerror.php`

```
1  <?php
2
3      $ch1 = curl_init("http://example.com");
4      $ch2 = curl_init("http://www.php.net/");
5
6      $mh = curl_multi_init();
7
8      curl_multi_add_handle($mh, $ch1);
9      curl_multi_add_handle($mh, $ch2);
10
11     // 멀티 실행 여부 체크 플래그
12     $active = null;
13
14     do {
15         $status = curl_multi_exec($mh, $active);
16         // Check for errors
17         if ($status > 0) {
18             // Display error message
19             echo "ERROR!\n " . curl_multi_strerror($status);
20         }
21     } while ($status === CURLM_CALL_MULTI_PERFORM || $active);
22
23     ?>
```

12.5 cURL 오류 처리

통신을 처리할 때는 예외적인 상황들이 자주 발생합니다. cURL을 처리할 때 발생할 수 있는 오류들을 처리할 수 있는 몇 가지 함수들을 지원합니다.

| 내장 함수 |

```
string curl_error ( resource $ch )
```

내장 함수 `curl_error()`는 cURL 오류가 발생된 경우 에러 문자열을 읽어옵니다.

| 내장 함수 |

```
int curl_errno ( resource $ch )
```

내장 함수 `curl_errno()`는 오류 코드를 반환합니다.

| 내장 함수 |

```
string curl_strerror ( int $errornum )
```

내장 함수 `curl_strerror()`는 오류 코드의 문자열을 반환합니다.

예제 파일 | **curl_error.php**

```
1  <?php
2      // 존재하지 않은 url의 세션을 초기화합니다.
3      $ch = curl_init('http://404.php.net/');
4      curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
5
6      if (curl_exec($ch) === false)
7      {
8          // 오류 발생 시 : 오류 메시지를 읽어옵니다.
9          echo "Curl error Message : ". curl_error($ch) . "<br>";
10
11         // 오류 넘버를 출력합니다.
12         $errNo = curl_errno ($ch);
13         echo "cURL Error No : ". $errNo;
14
15         echo " == ". curl_strerror($errNo);
16     }
```

```

17     } else {
18         echo 'success';
19     }
20
21     // 종료
22     curl_close($ch);
23
24     ?>

```

화면 출력

Curl error Message : Could not resolve host: 404.php.net
 cURL Error No : 6 == Couldn't resolve host name

12.6 cURL 그 외 함수

그 외 CURL을 처리하는 도움이 되는 몇 가지 관련 함수를 지원합니다.

| 내장 함수 |

```
array curl_version ([ int $age = CURLVERSION_NOW ] )
```

내장 함수 `curl_version()`은 cURL 버전 정보를 가지고 옵니다. 버전 정보를 배열값으로 반환합니다.

예제 파일 | `curl_version.php`

```

1  <?php
2      $version = curl_version();
3      print_r($version);
4
5  ?>

```

화면 출력

Array ([version_number] => 472321 [age] => 3 [features] => 2428829 [ssl_

```
version_number] => 0 [version] => 7.53.1 [host] => i386-pc-win32 [ssl_version]
=> OpenSSL/1.0.2k [libz_version] => 1.2.8 [protocols] => Array ( [0] => dict
[1] => file [2] => ftp [3] => ftps [4] => gopher [5] => http [6] => https [7]
=> imap [8] => imaps [9] => ldap [10] => pop3 [11] => pop3s [12] => rtsp [13]
=> scp [14] => sftp [15] => smtp [16] => smtps [17] => telnet [18] => tftp ) )
```

| 내장 함수 |

```
string curl_escape ( resource $ch , string $str )
```

내장 함수 `curl_escape()`는 주어진 문자열을 URL을 RFC 3986에 따라 인코딩합니다.

예제 파일 | `curl_escape.php`

```
1  <?php
2
3      $ch = curl_init();
4
5      // GET 파라미터로 사용되는 문자열 값을 RFC3986 인코딩합니다.
6      $location = curl_escape($ch, 'aaa / bbb');
7
8      $url = "http://example.com/test.php?location={$location}";
9      echo $url;
10
11  ?>
```

화면 출력

```
http://example.com/test.php?location=aaa%20%2F%20bbb
```

| 내장 함수 |

```
string curl_unescape ( resource $ch , string $str )
```

내장 함수 `curl_unescape()`는 URL을 RFC 3986에 따라 디코딩합니다.

예제 파일 | `curl_unescape.php`

```
1  <?php
2
3      $ch = curl_init();
4
5      // GET 파라미터로 사용되는 문자열 값을 RFC3986 인코딩합니다.
6      $location = curl_escape($ch, 'aaa / bbb');
7
8      $url = "http://example.com/test.php?location={$location}";
9      echo $url . "<br>";
10
11     echo curl_unescape($ch , $location);
12
13  ?>
```

화면 출력

```
http://example.com/test.php?location=aaa%20%2F%20bbb
aaa / bbb
```

| 내장 함수 |

mixed **curl_getinfo** (resource \$ch [, int \$opt])

내장 함수 `curl_getinfo()`는 cURL 세션 실행 성공 시 마지막 전송에 관련된 정보를 얻습니다.

- `CURLINFO_EFFECTIVE_URL` - 마지막 유효 URL
- `CURLINFO_HTTP_CODE` - 마지막으로 수신한 HTTP 코드
- `CURLINFO_FILETIME` - 검색된 문서의 원격 시간(`CURLOPT_FILETIME`이 활성화됨). -1이 반환되면 문서의 시간을 알 수 없습니다.
- `CURLINFO_TOTAL_TIME` - 마지막 전송에 대한 총 트랜잭션 시간(초)
- `CURLINFO_NAMELOOKUP_TIME` - 이름 확인이 완료 될 때까지의 시간(초)
- `CURLINFO_CONNECT_TIME` - 연결을 설정하는 데 걸리는 시간(초)
- `CURLINFO_PRETRANSFER_TIME` - 시작부터 파일 전송이 시작되기 직전까지

의 시간 (초)

- CURLINFO_STARTTRANSFER_TIME - 첫 번째 바이트가 전송 될 때까지의 초 단위 시간
- CURLINFO_REDIRECT_COUNT - CURLOPT_FOLLOWLOCATION 옵션을 사용하는 리디렉션의 수
- CURLINFO_REDIRECT_TIME - CURLOPT_FOLLOWLOCATION 옵션을 사용하여 최종 트랜잭션이 시작되기 전의 모든 리디렉션 단계의 시간(초)
- CURLINFO_REDIRECT_URL - CURLOPT_FOLLOWLOCATION 옵션을 사용 중지한 경우: 마지막 트랜잭션에서 찾은 리디렉션 URL을 수동으로 요청해야 합니다. CURLOPT_FOLLOWLOCATION 옵션을 사용하면 비어 있습니다. 이 경우 리디렉션 URL은 CURLINFO_EFFECTIVE_URL에서 사용할 수 있습니다.
- CURLINFO_PRIMARY_IP - 가장 최근 연결의 IP 주소
- CURLINFO_PRIMARY_PORT - 가장 최근 연결의 대상 포트
- CURLINFO_LOCAL_IP - 가장 최근 연결의 로컬 (소스) IP 주소
- CURLINFO_LOCAL_PORT - 가장 최근 연결의 로컬 (소스) 포트
- CURLINFO_SIZE_UPLOAD - 업로드된 총 바이트 수
- CURLINFO_SIZE_DOWNLOAD - 다운로드된 총 바이트 수
- CURLINFO_SPEED_DOWNLOAD - 평균 다운로드 속도
- CURLINFO_SPEED_UPLOAD - 평균 업로드 속도
- CURLINFO_HEADER_SIZE - 수신된 모든 헤더의 총 크기
- CURLINFO_HEADER_OUT - 요청 문자열이 전송되었습니다. 이 작업을 수행하려면 curl_setopt()를 호출하여 핸들에 CURLINFO_HEADER_OUT 옵션을 추가하십시오.
- CURLINFO_REQUEST_SIZE - 현재 HTTP 요청에 대해서만 발행된 요청의 총 크기
- CURLINFO_SSL_VERIFYRESULT - CURLOPT_SSL_VERIFYPEER를 설정하여 요청한 SSL 인증 확인 결과
- CURLINFO_CONTENT_LENGTH_DOWNLOAD - 다운로드 길이, Content-Length: 필드에서 읽음

- CURLINFO_CONTENT_LENGTH_UPLOAD - 지정된 업로드 크기
- CURLINFO_CONTENT_TYPE - Content-Type: 요청한 문서의 이름입니다. NULL은 서버가 유효한 Content-Type: 헤더를 보내지 않았음을 나타냅니다.
- CURLINFO_PRIVATE - 이전에 curl_setopt()의 CURLOPT_PRIVATE 옵션으로 설정한 이 cURL 핸들과 연관된 개인 데이터.
- CURLINFO_RESPONSE_CODE - 마지막 응답 코드
- CURLINFO_HTTP_CONNECTCODE - CONNECT 응답 코드
- CURLINFO_HTTPAUTH_AVAIL - 이전 응답에 따라 사용 가능한 인증 방법을 나타내는 비트 마스크
- CURLINFO_PROXYAUTH_AVAIL - 이전 응답에 따라 사용 가능한 프록시 인증 방법을 나타내는 비트 마스크
- CURLINFO_OS_ERRNO - 연결 실패로 인한 Errno입니다. 번호는 OS 및 시스템에 따라 다릅니다.
- CURLINFO_NUM_CONNECTS - 이전 전송을 위해 콜이 작성해야 하는 연결 수
- CURLINFO_SSL_ENGINES - OpenSSL 암호화 엔진 지원
- CURLINFO_COOKIELIST - 모든 알려진 쿠키
- CURLINFO_FTP_ENTRY_PATH - FTP 서버의 입력 경로
- CURLINFO_APPCONNECT_TIME - 시작부터 SSL/SSH 연결/핸드 셰이크가 원격 호스트에 완료될 때까지 걸린 시간(초)
- CURLINFO_CERTINFO - TLS 인증서 체인
- CURLINFO_CONDITION_UNMET - 충족되지 않은 시간 조건에 대한 정보
- CURLINFO_RTSP_CLIENT_CSEQ - 다음 RTSP 클라이언트 CSeq
- CURLINFO_RTSP_CSEQ_RECV - 최근 받은 CSeq
- CURLINFO_RTSP_SERVER_CSEQ - 다음 RTSP 서버 CSeq
- CURLINFO_RTSP_SESSION_ID - RTSP 세션 ID

예제 파일 | [curl_getinfo.php](#)

```
1 <?php
2
3 $ch = curl_init('http://www.example.com/');
```

```

4     curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
5
6     if ($response = curl_exec($ch)) {
7
8         $info = curl_getinfo($ch);
9         print_r($info);
10        echo "<br>";
11
12        switch ($http_code = curl_getinfo($ch, CURLINFO_HTTP_CODE)) {
13            case 200:
14                echo 'success <br>';
15                break;
16            default:
17                echo 'Unexpected HTTP code: ', $http_code, "\n";
18        }
19
20    } else {
21        // 오류 발생 시 : 오류 메시지를 읽어옵니다.
22        echo "Curl error Message : ". curl_error($ch) . "<br>";
23
24        // 오류 번호를 출력합니다.
25        echo "cURL Error No : ". curl_errno ($ch);
26
27    }
28
29    // 종료
30    curl_close($ch);
31    ?>

```

화면 출력

```

Array ( [url] => http://www.example.com/ [content_type] => text/html [http_
code] => 200 [header_size] => 311 [request_size] => 54 [filetime] => -1 [ssl_
verify_result] => 0 [redirect_count] => 0 [total_time] => 0.312 [namelookup_
time] => 0 [connect_time] => 0.156 [pretransfer_time] => 0.156 [size_
upload] => 0 [size_download] => 1270 [speed_download] => 4070 [speed_
upload] => 0 [download_content_length] => 1270 [upload_content_length] => -1
[starttransfer_time] => 0.312 [redirect_time] => 0 [redirect_url] => [primary_
ip] => 93.184.216.34 [certinfo] => Array ( ) [primary_port] => 80 [local_ip]
=> 172.30.1.3 [local_port] => 57793 )
success

```

| 내장 함수 |

```
resource curl_copy_handle ( resource $ch )
```

내장 함수 `curl_copy_handle()`은 cURL 핸들을 환경 설정과 함께 복사합니다.

예제 파일 | `curl_copy_handle.php`

```
1  <?php
2
3      $ch = curl_init();
4
5      curl_setopt($ch, CURLOPT_URL, 'http://www.example.com/');
6      curl_setopt($ch, CURLOPT_HEADER, 0);
7
8      // cURL handle를 복사합니다.
9      $ch2 = curl_copy_handle($ch);
10     curl_exec($ch2);
11
12     curl_close($ch2);
13     curl_close($ch);
14
15  ?>
```

12.7 POST 접속 응용

cURL의 기본 기능들을 이용하여 해당 사이트에 POST 형태로 접속할 수 있습니다. 다음은 POST 방식으로 웹 URL을 읽어오는 메서드 함수의 예제입니다.

```
public function _curl_post($url, $postfiled){
    $ch = curl_init();

    curl_setopt($ch, CURLOPT_URL,$url);
    curl_setopt($ch, CURLOPT_POST, 1);
```



```

    curl_setopt($ch, CURLOPT_POSTFIELDS,$postfild);

    // receive server response ...
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

    $server_output = curl_exec($ch);
    curl_close($ch);

    return $server_output;
}

```

12.8 파일 업로드

cURL을 응용하여 POST 파일 업로드도 가능합니다. 다음은 curl을 이용하여 파일을 업로드하는 예제 함수입니다.

```

function curl_upload($mode, $target_url, $path, $file){

    $file_name_with_full_path = realpath($file);

    $post = array('mode'=>$mode, 'path'=>$path, 'file_
    contents'=>'@'.$file_name_with_full_path);

    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL,$target_url);
    curl_setopt($ch, CURLOPT_POST,1);
    curl_setopt($ch, CURLOPT_POSTFIELDS, $post);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER,1);
    $result=curl_exec($ch);
    curl_close($ch);

    return $result;
}

```


13

외부 처리

PHP는 시스템의 외부 명령어를 실행하고, 결과값을 반환받을 수 있는 몇 개의 시스템 함수들을 제공합니다.

외부 명령 처리는 시스템의 커맨드 명령 창에서 실행하는 것과 동일한 처리를 의미합니다.

13.1 시스템 함수

PHP는 시스템의 명령을 호출하여 실행할 수 있습니다. PHP는 세 가지 방식으로 시스템을 호출하여 실행할 수 있습니다.

| 내장 함수 |

```
string exec ( string $command [, array &$output [, int &$return_var ] ] )
```

내장 함수 `exec()`는 시스템의 외부 명령을 실행합니다. `exec()`는 반환값으로는 처리 결과 문자열이 반환됩니다.

output 인수가 있으면 지정된 배열은 명령의 모든 출력 행을 배열로 채워집니다. 하지만 \n 같은 후행 공백은 이 배열에 포함되지 않습니다. 만일 output 변수가 초기화되지 않은 값인 경우에 배열 뒤에 추가로 연결됩니다.

예제 파일 | **exec.php**

```
1 <?php
2     $result = exec("dir",$output);
3     echo $result;
4     echo "<br>";
5     print_r($output);
6
7 ?>
```

화면 출력

```
2개 디렉터리 84,432,617,472바이트 남음
Array ( [0] => C 드라이브의 볼륨: windows [1] => 볼륨 일련번호: 447E-0DB0 [2] => [3]
=> C:\php-7.1.4-Win32-VC14-x86\sample1 디렉터리 [4] => [5] => 2017-08-17 오후
06:06
. [6] => 2017-08-17 오후 06:06
.. [7] => 2017-08-17 오후 06:04 95 exec.php [8] => 1개 파일 95바이트 [9] => 2개 디렉
터리 84,432,617,472바이트 남음
```

| 내장 함수 |

```
string system ( string $command [, int &$return_var ] )
```

내장 함수 system()은 외부 프로그램을 실행합니다. 실행 후 **결과를 출력**합니다. system()은 주어진 명령을 실행하고 결과를 출력한다는 점에서 함수의 C 버전과 같습니다.

예제 파일 | **system.php**

```
1 <?php
2     $result = system("dir",$output);
3     echo $result;
4
5 ?>
```

화면 출력

```
C 드라이브의 볼륨: windows 볼륨 일련번호: 447E-0DB0 C:\php-7.1.4-Win32-VC14-x86\
sample1 디렉터리 2017-08-17 오후 06:20
. 2017-08-17 오후 06:20
.. 2017-08-17 오후 06:04 95 exec.php 2017-08-17 오후 06:20 66 system.php 2개 파일
161바이트 2개 디렉터리 84,477,603,840바이트 남음 2개 디렉터리 84,477,603,840바이트 남음
```

| 내장 함수 |

```
string shell_exec ( string $cmd )
```

내장 함수 `shell_exec()`는 셸을 통해 명령을 실행합니다. 실행한 전체 출력을 문자열로 반환합니다.

예제 파일 | `shell_exec.php`

```
1 <?php
2     $output = shell_exec('ls -lart');
3     echo "<pre>$output</pre>";
4
5 ?>
```

13.2 프로세스

| 내장 함수 |

```
resource proc_open ( string $cmd , array $descriptorspec , array &$amp;pipes [, string
$cwd [, array $env [, array $other_options ]]])
```

내장 함수 `proc_open()`은 명령 실행과 입/출력에 대한 파일 포인터를 생성합니다. `proc_open()`은 `popen()`과 비슷한 점이 많지만 좀 더 유연한 제어를 할 수 있습니다.

`cmd`는 실행할 명령어입니다.

두 번째 인자 descriptorspec는 인덱스 타입의 배열입니다. 인덱스 키는 자식 프로세스에게 전달되는 방법을 설정합니다. 0은 stdin, 1은 stdout, 2는 stderr입니다.

pipes는 생성된 파이프의 PHP 끝 부분에 해당하는 파일 포인터의 인덱스 배열로 설정됩니다.

cwd는 명령을 실행하는 초기 작업 디렉터리입니다. 이 값은 상대 경로가 아닌 절대 경로 형태로 지정해야 합니다. 또는 기본 PHP가 설치된 작업 디렉터를 사용할 경우에는 NULL을 사용하면 됩니다.

env는 실행되는 명령어에 대한 환경 변수를 배열로 설정합니다.

other_options

추가 옵션을 지정할 수 있습니다. 현재 지원되는 옵션은 다음과 같습니다.

- **suppress_errors**(윈도우 전용): 이 함수가 TRUE로 설정된 경우 생성된 오류를 억제합니다.
- **bypass_shell**(윈도우 전용): TRUE로 설정하면 cmd.exe 셸을 무시합니다.

| 내장 함수 |

```
int proc_close ( resource $process )
```

내장 함수 `proc_open()`에 의해 열린 프로세스를 닫습니다. 프로세스의 종료 코드를 반환합니다.

- **command**: `proc_open()`에 전달된 명령 문자열입니다.
- **pid**: process id
- **running**: 프로세스가 아직 실행 중이면 TRUE, 종료된 경우 FALSE 값을 반환합니다.
- **signaled**: 자식 프로세스가 포착되지 않은 신호에 의해 종료된 경우 TRUE 값입니다. 윈도우에서는 항상 FALSE 값으로 설정됩니다.

- **stopped**: 자식 프로세스가 신호에 의해 중단 된 경우 TRUE 값 입니다. 윈도우에서는 항상 FALSE로 설정됩니다.
- **exitcode**: 실행 도중 프로세스가 반환한 종료 코드로 FALSE 값입니다. 첫 번째 호출만 실제 값을 반환하고, 다음 호출은 -1을 반환합니다.
- **termsig**: 자식 프로세스가 실행을 종료하게 만든 신호의 번호입니다
- **stopsig**: 중지된 경우 자식 프로세스가 실행을 멈추게 만든 신호의 번호입니다.

| 내장 함수 |

```
bool proc_nice ( int $increment )
```

내장 함수 `proc_nice()`는 현재 프로세스의 우선순위 변경합니다.

`proc_nice()`는 `increment`에 지정된 양만큼 현재 작업 중인 프로세스의 우선순위를 변경하게 됩니다. 양수 값의 증가는 현재 프로세스의 우선순위를 낮추고, 음수 값의 증가는 우선순위를 높입니다.

| 내장 함수 |

```
bool proc_terminate ( resource $process [, int $signal = 15 ] )
```

내장 함수 `proc_terminate()`는 `proc_open`에 의해 오픈된 프로세스를 종료합니다. `proc_open()` 함수를 통해서 생성된 프로세스를 종료합니다. `proc_terminate()` 호출 시 프로세서는 기다리지 않고 즉시 종료합니다.

예제 파일 | **proc.php**

```
1  <?php
2      $descriptorspec = array(
3          0 => array("pipe", "r"), // stdin
4          1 => array("pipe", "w"), // stdout
5          2 => array("file", "/tmp/error-output.txt", "a") // stderr
6      );
```

```

7
8     $cwd = '/tmp';
9     $env = array('some_option' => 'aeiou');
10
11     $process = proc_open('php', $descriptorspec, $pipes, $cwd, $env);
12     echo "proc status<br>";
13     print_r(proc_get_status($process));
14     echo "<br>";
15
16     if (is_resource($process)) {
17         // $pipes now looks like this:
18         // Any error output will be appended to /tmp/error-output.txt
19
20         echo "0 => writeable handle connected to child stdin <br>";
21         $php_code = "<?php echo 'hello' ?>";
22         fwrite($pipes[0], $php_code);
23         fclose($pipes[0]);
24
25         echo "1 => readable handle connected to child stdout <br>";
26         echo stream_get_contents($pipes[1]);
27         echo "<br>";
28         fclose($pipes[1]);
29
30         // It is important that you close any pipes before calling
31         // proc_close in order to avoid a deadlock
32         $return_value = proc_close($process);
33
34         echo "command returned $return_value <br>";
35     }
36
37     ?>

```

화면 출력

```

proc status
Array ( [command] => php [pid] => 28647 [running] => 1 [signaled] => [stopped]
=> [exitcode] => -1 [termsig] => 0 [stopsig] => 0 )
0 => writeable handle connected to child stdin
1 => readable handle connected to child stdout
hello
command returned 0

```


14

메일

PHP의 내장 메일 함수는 PHP를 이용하여 간단하게 메일을 전송할 수 있습니다. 메일을 발송하기 위해서는 메일 발송 SMTP 서버가 필요합니다.

14.1 SMTP

SMTP는 Simple Mail Transfer Protocol의 약자로 메일 발송 역할을 하는 서버입니다. 보통 메일을 발송하려면 메일 전송 서버와 수신 서버가 필요합니다.

PHP 내장 함수를 통하여 메일을 전송하기 위해서는 SMTP 발송 서버가 설치되어 있어야 합니다.

리눅스에서 대표적으로 사용하는 SMTP 서버로는 sendmail 데몬 서비스가 있습니다 (sendmail 데몬 서비스에 대한 설치 및 설정 정보는 리눅스 관련 서적 참조).

14.2 메일 작성

PHP에서는 내장 함수 `mail()`을 통하여 메일을 작성할 수 있습니다. 작성된 메일은 지정된 메일 서버로 전달되고, 메일 서버를 통하여 상대방에게 전달됩니다.

| 내장 함수 |

```
bool mail ( string $to , string $subject , string $message [, string $additional_headers  
[, string $additional_parameters ] ] )
```

`mail()` 함수는 메일 전송에 필요한 몇 개의 인자를 필요로 합니다.

to: 메일 수신 이메일

subject: 메일의 제목

message: 메일의 본문

additional_headers: 메일 헤더 부분

예제 파일 | `mail.php`

```
1  <?php
2      // 메일 메시지 본문
3      $message = "내용1\r\n 내용2\r\n 내용3";
4
5      // 메일 내용이 길어진 경우 wordwrap() 함수를 통하여 넘길 수 있습니다.
6      $message = wordwrap($message, 70, "\r\n");
7
8      $to = "infohojin@naver.com";
9      $subject = "메일 테스트";
10
11     $header = 'From: infohojin@naver.com' . "\r\n" .
12         'Reply-To: infohojin@naver.com' . "\r\n" .
13         'X-Mailer: PHP/' . phpversion();
14
15     // Send
16     mail($to, $subject, $message, $headers);
17
18  ?>
```

15

오류 처리 함수

PHP는 오류에 대한 처리를 할 수 있는 몇 가지 함수들을 제공합니다.

15.1 오류

PHP는 다양한 오류의 정의와 메시지 코드를 지원합니다. 내장 함수 `error_reporting()`는 PHP 에러를 정의합니다.

| 내장 함수 |

```
int error_reporting ([ int $level ] )
```

PHP에서 사용되는 대표적인 에러 코드와 기호는 다음과 같습니다.

- 1: `E_ERROR` 에러를 출력하고 스크립트의 실행을 중단합니다. 메모리 할당 에러 등의 복구가 힘든 문제의 에러를 의미합니다.

- 2: E_WARNING 경고를 출력하지만 스크립트는 정상적으로 실행됩니다.
- 4: E_PARSE 컴파일 타임 구문 분석 오류 메시지입니다.
- 8: E_NOTICE 뭔가 에러를 감지했으나 출력은 하지 않습니다.
- 16: E_CODE_ERROR PHP 코어에 의하여 생성된 에러 메시지입니다.
- 32: E_CODE_WARNING PHP 코어에 의하여 생성된 에러 메시지입니다.

예제 파일 | **error_report.php**

```
1  <?php
2
3      // Turn off all error reporting
4      error_reporting(0);
5
6      // Report simple running errors
7      error_reporting(E_ERROR | E_WARNING | E_PARSE);
8
9      // Reporting E_NOTICE can be good too (to report uninitialized
10     // variables or catch variable name misspellings ...)
11     error_reporting(E_ERROR | E_WARNING | E_PARSE | E_NOTICE);
12
13     // Report all errors except E_NOTICE
14     error_reporting(E_ALL & ~E_NOTICE);
15
16     // Report all PHP errors (see changelog)
17     error_reporting(E_ALL);
18
19     // Report all PHP errors
20     error_reporting(-1);
21
22     // Same as error_reporting(E_ALL);
23     ini_set('error_reporting', E_ALL);
24
25  ?>
```

15.2 오류 출력

PHP에서 발생한 오류에 대한 정보를 가져오고 로그를 출력할 수 있습니다.

| 내장 함수 |

```
bool error_log ( string $message [, int $message_type = 0 [, string $destination [,  
string $extra_headers ]]] )
```

내장 함수 `error_log()`는 메시지를 오류 처리 루틴으로 전달합니다. 지정한 타입에 따라서 메시지를 출력할 곳을 지정할 수 있습니다.

- 타입0: `php.ini`에 지정된 시스템에 에러를 출력합니다.
- 타입1: 지정한 이메일로 에러 메시지를 출력합니다.
- 타입2: 호스트, IP 주소의 PHP 디버깅으로 출력합니다. 디버깅 출력은 `remote debugging` 설정이 되어 있어야 합니다.
- 타입3: 지정한 파일로 출력합니다.

예제 파일 | `error_log.php`

```
1  <?php
2      // 지정한 로그 파일에 에러를 출력합니다.
3      $errFile = "./my-errors.log";
4      error_log("You messed up!", 3, $errFile);
5
6  ?>
```

| 내장 함수 |

```
array error_get_last ( void )
```

내장 함수 `error_get_last()`는 마지막으로 발생한 오류를 가지고 옵니다.

예제 파일 | [error_report.php](#)

```
1 <?php
2     echo $a;
3     print_r(error_get_last());
4
5 ?>
```

| 내장 함수 |

void error_clear_last (void)

내장 함수 `error_clear_last()`는 가장 최근 오류를 지웁니다.

예제 파일 | [error_clear_last.php](#)

```
1 <?php
2     var_dump(error_get_last());
3     error_clear_last();
4     var_dump(error_get_last());
5
6     @$a = $b;
7
8     var_dump(error_get_last());
9     error_clear_last();
10    var_dump(error_get_last());
11
12 ?>
```

| 내장 함수 |

bool trigger_error (string \$error_msg [, int \$error_type = E_USER_NOTICE])

내장 함수 `trigger_error()`는 사용자 수준의 오류/경고/통지 메시지를 생성합니다.

예제 파일 | `trigger_error.php`

```
1 <?php
2     trigger_error("Cannot divide by zero", E_USER_ERROR);
3
4 ?>
```

콘솔 출력

[Mon Aug 21 16:59:44 2017] ::1:57186 [500]: /trigger_error.php - **Cannot divide by zero** in C:\php-7.1.4-Win32-VC14-x86\trigger_error.php on line 3

15.3 역추적

PHP는 역추적에 관련된 몇 개의 내장 함수들을 지원합니다.

| 내장 함수 |

```
array debug_backtrace ([ int $options = DEBUG_BACKTRACE_PROVIDE_OBJECT  
[, int $limit = 0 ] ] )
```

내장 함수 `debug_backtrace()`는 역추적을 실행합니다.

예제 파일 | `debug_backtrace.php`

```
1 <?php
2     function test($str)
3     {
4         echo "Hello World! $str<br>";
5         var_dump(debug_backtrace());
6     }
7
8     test('jiny');
9
10
11 ?>
```

콘솔 출력

```
Hello World! jiny
array(1) { [0]=> array(4) { ["file"]=> string(47) "C:\php-7.1.4-Win32-VC14-
x86\debug_backtrace.php" ["line"]=> int(8) ["function"]=> string(4) "test"
["args"]=> array(1) { [0]=> string(4) "jiny" } } }
```

| 내장 함수 |

```
void debug_print_backtrace ([ int $options = 0 [, int $limit = 0 ] ] )
```

내장 함수 `debug_print_backtrace()`는 역추적을 출력합니다.

예제 파일 | `debug_print_backtrace.php`

```
1  <?php
2
3      function a() {
4          b();
5      }
6
7      function b() {
8          c();
9      }
10
11     function c() {
12         debug_print_backtrace();
13     }
14
15     a();
16
17  ?>
```

화면 출력

```
#0 c() called at [C:\php-7.1.4-Win32-VC14-x86\debug_print_backtrace.php:8]
#1 b() called at [C:\php-7.1.4-Win32-VC14-x86\debug_print_backtrace.php:4]
#2 a() called at [C:\php-7.1.4-Win32-VC14-x86\debug_print_backtrace.php:15]
```


15.4 오류 핸들

PHP는 오류 처리 핸들에 관련된 몇 가지 함수들을 지원합니다.

| 내장 함수 |

```
mixed set_error_handler ( callable $error_handler [, int $error_types = E_ALL | E_STRICT ] )
```

내장 함수 `set_error_handler()`는 사용자 정의 오류 처리기 함수를 설정합니다.

| 내장 함수 |

```
bool restore_error_handler ( void )
```

내장 함수 `restore_error_handler()`는 이전 오류 처리 함수를 복원합니다.

| 내장 함수 |

```
callable set_exception_handler ( callable $exception_handler )
```

내장 함수 `set_exception_handler()`는 사용자 정의 예외 처리 함수를 설정합니다.

| 내장 함수 |

```
bool restore_exception_handler ( void )
```

내장 함수 `restore_exception_handler()`는 이전에 정의된 예외 핸들러 함수를 복원합니다.

16

함수

PHP는 수많은 함수들을 포함하고 있습니다. 또한 다양한 함수들을 통하여 PHP 응용프로그램을 개발합니다.

함수들은 내부적으로 제공 함수와 사용자가 직접 작성하여 생성한 함수로 구분할 수 있습니다. 또한 함수 이름은 중복해서 사용할 수 없습니다. PHP는 내부적으로 함수를 관리하고 처리할 수 있는 별도의 특수 함수들을 제공합니다.

16.1 함수 목록

함수를 작성하거나 사용할 때 함수의 이름은 매우 중요합니다. 함수는 PHP 내에서 중복해서 사용할 수 없습니다. 만일 중복된 함수명을 생성하면 오류를 발생합니다.

또한 존재하지 않은 함수를 호출할 때도 PHP는 오류를 발생합니다. PHP는 지정한 이름의 함수가 존재하는지 확인할 수 있는 특별한 함수를 제공합니다.

| 내장 함수 |

```
bool function_exists ( string $function_name )
```

내장 함수 `function_exists()`는 스크립트 내에 주어진 이름의 함수에 정의되어 있는지를 확인합니다.

예제 파일 | [function_exists.php](#)

```
1  <?php
2      if (function_exists('str_replace')) {
3          echo "함수가 정의되어 있습니다.<br />\n";
4      } else {
5          echo "없는 함수가 있습니다.<br />\n";
6      }
7
8  ?>
```

화면 출력

함수가 정의되어 있습니다.

| 내장 함수 |

```
array get_defined_functions ([ bool $exclude_disabled = FALSE ] )
```

내장 함수 `get_defined_functions()`는 정의된 모든 함수의 목록을 배열로 반환합니다.

예제 파일 | [get_defined_functions.php](#)

```
1  <?php
2      $arr = get_defined_functions();
3      print_r($arr);
4
5  ?>
```

16.2 함수 인자

우리가 함수를 이용하는 것은 코드의 중복되는 처리들을 블록화하는 것입니다. 함수는 처리를 위한 기본값의 매개변수와 반환값들을 가지고 있습니다.

PHP는 유연한 함수의 매개변수 처리를 지원합니다. 이와 관련하여 매개변수를 처리할 수 있는 몇 가지 함수들을 제공합니다.

| 내장 함수 |

```
int func_num_args ( void )
```

내장 함수 `func_num_args()`는 함수에 전달된 인수의 개수를 반환합니다.

예제 파일 | `func_num_args.php`

```
1  <?php
2      function foo()
3      {
4          $numargs = func_num_args();
5          echo "Number of arguments: $numargs\n";
6      }
7
8      foo(1, 2, 3);
9
10 ?>
```

화면 출력

Number of arguments: 3

위의 실험에서 사용자 정의 함수의 매개변수를 지정하지 않았습니다. 그리고 3개의 값을 전달하여 함수를 호출했습니다. `func_num_args()`를 통하여 전달되는 인자의 개수를 확인할 수 있습니다.

| 내장 함수 |

mixed **func_get_arg** (int \$arg_num)

내장 함수 `func_get_arg()`는 인수 목록에서 항목 반환합니다.

예제 파일 | **func_get_arg.php**

```
1  <?php
2      function foo()
3      {
4          $numargs = func_num_args();
5          echo "Number of arguments: $numargs <br>";
6          if ($numargs >= 2) {
7              echo "Second argument is: " . func_get_arg(1) . " <br>";
8          }
9      }
10
11      foo(1, 2, 3);
12
13  ?>
```

화면 출력

Number of arguments: 3

Second argument is: 2

| 내장 함수 |

array **func_get_args** (void)

내장 함수 `func_get_args()`는 함수의 인수 목록을 구성하는 배열을 반환합니다.

예제 파일 | **func_get_args.php**

```
1  <?php
2      function foo()
3      {
```

```

4     $numargs = func_num_args();
5     echo "Number of arguments: $numargs <br>";
6     if ($numargs >= 2) {
7         echo "Second argument is: " . func_get_arg(1) . "<br>";
8     }
9
10    $arg_list = func_get_args();
11    for ($i = 0; $i < $numargs; $i++) {
12        echo "Argument $i is: " . $arg_list[$i] . "<br>";
13    }
14 }
15
16 foo(1, 2, 3);
17
18 ?>

```

화면 출력

```

Number of arguments: 3
Second argument is: 2
Argument 0 is: 1
Argument 1 is: 2
Argument 2 is: 3

```

16.3 콜백 호출

| 내장 함수 |

mixed **call_user_func_array** (callable \$callback , array \$param_arr)

내장 함수 `call_user_func_array()`는 매개변수 배열을 사용하여 콜백 호출합니다.

예제 파일 | `call_user_func_array.php`

```

1  <?php
2      // 함수 호출
3      function foobar($arg, $arg2) {

```

```

4      echo __FUNCTION__, " got $arg and $arg2 <br>";
5  }
6
7      // 함수명
8      // 매개변수 배열
9      call_user_func_array("foobar", array("one", "two"));
10
11     // 객체 메서드 호출
12     class foo {
13         function bar($arg, $arg2) {
14             echo __METHOD__, " got $arg and $arg2<br>";
15         }
16     }
17
18     $foo = new foo;
19     // 인스턴스, 메서드 배열
20     // 매개변수 배열
21     call_user_func_array(array($foo, "bar"), array("three", "four"));
22
23     ?>

```

화면 출력

```

foobar got one and two
foo::bar got three and four

```

| 내장 함수 |

mixed **call_user_func** (callable \$callback [, mixed \$parameter [, mixed \$...]])

내장 함수 call_user_func()는 매개변수에 의해 콜백 호출 처리합니다.

예제 파일 | call_user_func.php

```

1  <?php
2      function barber($type)
3      {
4          echo "You wanted a $type haircut <br>";
5      }

```



```

6
7     call_user_func('barber', "mushroom");
8     call_user_func('barber', "shave");
9
10  ?>

```

화면 출력

You wanted a mushroom haircut
 You wanted a shave haircut

16.4 메서드 호출

| 내장 함수 |

mixed **forward_static_call** (callable \$function [, mixed \$parameter [, mixed \$...]])

내장 함수 forward_static_call()은 정적 메서드를 호출합니다.

예제 파일 | forward_static_call.php

```

1  <?php
2
3      class A
4      {
5          const NAME = 'A';
6          public static function test() {
7              $args = func_get_args();
8              echo static::NAME, " ".join(', ', $args). " <br>";
9          }
10     }
11
12     class B extends A
13     {
14         const NAME = 'B';
15
16         public static function test() {
17             echo self::NAME, "<br>";

```

```

18         forward_static_call(array('A', 'test'), 'more', 'args');
19         forward_static_call( 'test', 'other', 'args');
20     }
21 }
22
23 B::test('foo');
24
25 function test() {
26     $args = func_get_args();
27     echo "C ".join(', ', $args)." <br>";
28 }
29
30 ?>

```

화면 출력

```

B
B more,args
C other,args

```

| 내장 함수 |

mixed **forward_static_call_array** (callable \$function , array \$parameters)

내장 함수 `forward_static_call_array()`는 정적 메서드를 호출합니다. 인수는 배열로 전달합니다.

예제 파일 | `forward_static_call_array.php`

```

1  <?php
2
3      class A
4      {
5          const NAME = 'A';
6          public static function test() {
7              $args = func_get_args();
8              echo "Class == ".static::NAME, " ".join(', ', $args)." <br>";
9          }
10     }

```

```

11
12     class B extends A
13     {
14         const NAME = 'B';
15
16         public static function test() {
17             echo "Class : ". self::NAME, "<br>";
18
19             // A클래스 test 메서드 호출
20             forward_static_call_array(array('A', 'test'), array('more',
21                                     'args'));
22
23             // 함수 호출
24             forward_static_call_array( 'test', array('other', 'args'));
25         }
26     }
27     B::test('foo');
28
29     function test() {
30         $args = func_get_args();
31         echo "function call = ".join(', ', $args)." <br>";
32     }
33
34     ?>

```

화면 출력

```

Class : B
Class == B more,args
function call = other,args

```

16.5 틱 실행

| 내장 함수 |

```
bool register_tick_function ( callable $function [, mixed $arg [, mixed $... ] ] )
```

내장 함수 `register_tick_function()`은 틱에서 실행될 함수를 등록합니다.

예제 파일 | **register_tick_function.php**

```
1  <?php
2      declare(ticks=1);
3
4      register_tick_function('my_function', true);
5
6      $object = new my_class();
7      register_tick_function(array(&$object, 'my_method'), true);
8
9  ?>
```

| 내장 함수 |

void unregister_tick_function (string \$function_name)

내장 함수 `unregister_tick_function()`은 틱에서 실행될 함수를 해제합니다.

17

HTML & FORM

HTML은 크게 화면을 구성하는 태그와 사용자의 입력을 받아 서버로 전송하는 FORM 구성 요소로 구분할 수 있습니다.

FORM 구성 요소들은 서버로 데이터를 전송하는데, PHP는 이런 전송된 데이터를 받아서 처리할 수 있습니다.

17.1 FORM

HTML의 FORM 태그는 폼 요소들을 감싸고 있는 바디 부분과 그 안에 들어가는 요소들의 항목으로 구분할 수 있습니다.

17.1.1 FORM 태그

HTML에서 <form></form>으로 마크업된 코드를 form 태그라고 합니다. 폼 태그는 폼 요소들을 포함하는 바디와 같습니다.

```
<form>
  폼요소들...
</form>
```

폼 요소들은 폼 태그로 둘러싸여 있어야 합니다.

폼 태그는 몇 개의 속성 설정을 할 수 있습니다.

name

폼의 이름을 지정합니다.

method

메서드 속성은 입력된 데이터의 전송 방식을 설정합니다. POST 방식 또는 GET 방식으로 지정할 수 있습니다. 만일 POST 방식으로 설정했다면 서버 쪽 PHP에서는 슈퍼 변수 \$_POST 배열을 통해서 데이터를 가져올 수 있습니다.

GET 방식으로 설정했다면 슈퍼 변수 \$_GET 배열을 통해서 데이터를 읽어올 수 있습니다.

action

액션 속성은 해당 폼을 처리하는 PHP 스크립트의 url과 파일명을 지정합니다. 폼이 서밋 되면 액션에서 지정한 속성으로 페이지가 이동되면서 데이터를 처리합니다.

enctype="multipart/form-data"

파일 업로드와 같이 바이너리 데이터를 전송할 경우에 추가되는 속성입니다.

17.1.2 Action

폼 태그의 속성 중 action은 서밋 클릭 시 처리되는 스크립트 페이지입니다. 전형적인 HTML 코딩에서는 직접 url과 스크립트명을 입력합니다.

```
<form method="post" action=".test.php">
```

하지만 form 요소를 작성하면서 처리되는 action 파일명을 직접 지정하는 방식은 불편합니다. 또한 잘못 이름을 타이핑하여 오동작을 발생할 수도 있을 것입니다.

이런 경우 슈퍼 변수 \$_SERVER["PHP_SELF"] 값을 통하여 다음과 같이 편리하게 사용하기도 합니다.

```
<form method="post" action="$_SERVER['PHP_SELF']">
```

\$_SERVER["PHP_SELF"]은 슈퍼 글로벌 변수입니다. 즉 action 값으로 현재 실행하고 있는 PHP 스크립트의 파일명을 가리키고 있습니다.

action="" 부분에 \$_SERVER["PHP_SELF"]를 입력하는 것은 서밋 버튼 클릭 시 다른 페이지로 이동하지 말고, 다시 자기 자신으로 다시 호출하여 데이터를 처리하라는 의미입니다.

하지만 이렇게 편리하게 사용한 \$_SERVER["PHP_SELF"] 테크닉은 요즘 들어 보안적으로 문제가 발생되고 있습니다. 이러한 셀프 페이지 방식은 동일한 페이지 form에서 오류 메시지를 출력하기 좋습니다.

웹 페이지는 불특정 다수의 사람들에게 노출되는 특성이 있습니다. 그중에서는 정상적인 사용자 이외에 해커 및 블랙유저 또한 있을 것입니다. PHP의 서버 전역 변수 \$_SERVER["PHP_SELF"]의 값은 해커의 이해 노출될 수 있습니다.

PHP_SELF를 사용하면 해커들을 슬래시(/)를 입력한 후에 XSS(Cross Site Scripting) 명령을 입력하여 정상 실행을 방해할 수 있습니다. XSS는 웹 응용프로그램에서 일반적으로 발견되는 컴퓨터 보안적 취약점 유형입니다. XSS를 사용하면 공격자가 다른 사용자가 본 웹 페이지에 클라이언트 측 스크립트를 삽입할 수 있습니다.

예를 들어 현재의 스크립트 test.php에서 폼 액션 값을 다음과 같이 설정했다고 생각해 봅시다.

```
<form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
```

이 페이지가 정상적으로 서밋했다면 현재의 페이지 test.php일 것입니다.

```
<form method="post" action="test.php">
```

하지만 해커가 브라우저 주소에 `http://www.도메인.com/test.php%22%3E%3Cscript%3Ealert('hacked')%3C/script%3E`처럼 입력했다면 어떻게 될까요?

특수문자를 풀어서 보면은 위의 주소는 다음과 같습니다.

```
<form method="post" action="test.php/"><script>alert('hacked')</script>
```

즉, 폼 태그 뒤에 추가로 자바스크립트 코드가 더 들어 있습니다. 변경된 폼 요소는 자바스크립트를 실행하고 페이지가 해킹될 수도 있습니다.

`$_SERVER["PHP_SELF"]`의 보안적인 부분 때문에 직접 변수를 사용하지 않고 `htmlspecialchars()` 함수를 같이 이용하여 처리합니다.

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
```

`htmlspecialchars()` 함수는 HTML의 특수문자들을 변환 처리합니다. 특수문자란 `<`, `>` 그리고 `<` 와 `>`를 말합니다.

이런 특수문자는 HTML 또는 자바스크립트 코드 등에서 인젝션 공격을 당할 수 있습니다. 또한 사이트가 위협에 노출될 수 있습니다.

다시 한번 예를 들면,

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
```


htmlspecialchars() 함수를 같이 사용하면

```
<form method="post" action="test.php/quote"><script>alert('hacked')</script></form>
```

형태로 변경이 되기 때문에 좀 더 안전한 실행이 가능할 것입니다.

17.1.3 서밋

서밋(submit)은 form에서 입력된 데이터를 action에서 지정한 url로 전송을 처리하는 승인 버튼입니다. 즉, 클라이언트 브라우저에서 사용자 데이터 값을 입력 후에 서밋 버튼을 클릭하면 서버 쪽으로 데이터를 전송합니다.

기본적으로 서밋을 지정하는 양식은 다음과 같습니다.

```
<input type='submit' value='확인'>
```

input 태그로 타입을 submit으로 설정하면 됩니다. 또한 버튼의 이름을 value 속성값으로 설정하면 됩니다.

또한 서밋 버튼을 클릭 시 자바스크립트를 실행하여 처리할 수도 있습니다.

```
<input type='submit' value='확인' onclick='\"javascript:check_submit()\"' >
```

서버로 데이터 값을 전달하기 전에 자바스크립트를 통하여 필수 항목이나 유효성 등을 확인한 후에 처리할 수 있습니다.

타입 형식을 image로 설정할 경우 이미지 버튼으로 사용할 수 있습니다.

```
<input type='image' src='img/submit.png' >
```

17.1.4 리셋

리셋(reset)은 서밋의 반대로 입력한 폼의 값들을 모두 초기화합니다.

```
<input type='reset' value='지우기'>
```

17.2 요소

HTML은 폼 태그 안에 다양한 요소들을 삽입하여 데이터를 입력받을 수 있습니다.

폼의 요소 값을 입력받아 처리할 때는 프로그램 동작 오류를 줄이기 위해서 필수 체크와 유효성 값을 같이 처리하는 것이 필요합니다.

필수 체크

폼 입력은 사용자가 입력한 데이터를 기반으로 서버 PHP에서 데이터를 처리합니다. 만일 사용자가 실수로 PHP에서 처리해야 하는 값을 누락할 수도 있습니다.

이런 경우 다음에 처리할 유효성 체크 부분과 정상적인 프로그램 로직 처리가 불가능해질 수도 있습니다. 물론, 폼을 처리하기 전에 자바스크립트 등으로 체크한다고 해도 외부에서 잘못된 경로로 처리 PHP로 접근하여 오류를 발행하거나, 해킹될 수도 있을 것입니다.

따라서 폼으로 입력을 받은 데이터 값 중에 반드시 필요로 하는 값 등은 필수 체크를 중복으로 하는 것이 좋습니다.

값을 입력받을 때 다음과 같이 empty() 함수를 응용하여,

```
if (empty($_POST["name"])) {  
    참: 비어 있습니다...  
} else {  
    거짓: 비어 있지 않습니다...  
}
```

패턴으로 필수 체크를 하는 것이 좋습니다.

유효성

폼의 유효성 데이터 값을 체크하는 것은 매우 중요합니다. 의미 없는 값을 입력하여 프로그램의 오류를 발생시키거나, 해킹 등의 원인이 될 수도 있습니다.

따라서 정확한 처리 로직과 보안을 위해서 반드시 데이터를 처리하기 전에 PHP에서도 한 번 더 데이터를 검사하는 작업을 하는 것이 좋습니다.

PHP의 필터링 함수의 기능을 이용하면 좀 더 복잡한 데이터의 유효성을 쉽게 확인할 수 있습니다.

17.2.1 input 텍스트

HTML의 input Text 요소는 폼 요소로 가장 사용 빈도가 높은 요소입니다. TEXT는 사용자로부터 다양한 문자열을 입력받을 수 있습니다. 입력되는 값이 문자열일 수도 있고, 또한 문자를 가장한 숫자일 수도 있습니다. 그 외에도 이메일, 이름, url 등 모두 문자열에 속하기 때문에 다양한 입력값들이 올 수 있습니다.

```
<input type="text" name="name">
```

예제 파일 | form_text.php

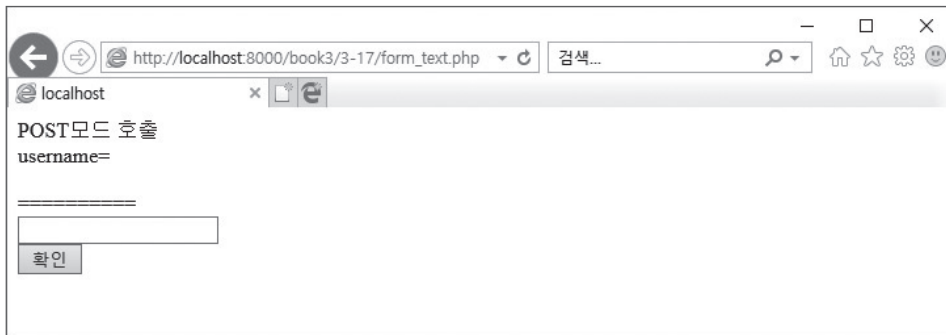
```
1  <?php
2      // 서밋 처리 시 데이터 값 확인
3      if ($_SERVER["REQUEST_METHOD"] == "POST") {
4          echo "POST 모드 호출 <br>";
5          echo "username= <br>" . $_POST['username'];
6          echo "<br>=====<br>";
7      }
8
9      // HTML form을 출력합니다.
10
11     echo "<form name='jiny' method='post' action='". $_SERVER['PHP_
12         SELF']."'>";
13
14     echo "<input type=\"text\" name=\"username\">";
```

```

14
15     echo "<br>";
16     echo "<input type='submit' value='확인'>";
17     echo "</form>";
18
19 ?>

```

화면 출력



다양한 타입 속성

HTML5의 input은 좀 더 많은 다양한 타입 속성을 지원합니다. 타입 속성을 이용하면 데이터 입력 시 발생할 수 있는 오류들을 먼저 브라우저 차원에서 필터링할 수 있습니다.

email
 url
 search
 tel
 number
 date
 datetime
 month
 week
 time
 range
 color

필수 체크

HTML5에는 폼의 필수 입력을 확인하는 속성인 **required**입니다. 하지만 이 속성은 브라우저에서만 동작하는 속성입니다. 브라우저 이외의 방식으로 접속할 경우에는 필수 체크를 할 수 없습니다. 이런 경우 PHP 소스상에서 다시 한번 필수 체크해주는 것이 중요합니다.

입력값 제한

HTML5는 브라우저에서 입력 문자열의 길이를 제한할 수 있습니다. `input` 태그의 속성으로 `maxlength="값"`을 추가하면 됩니다.

`maxlength`의 입력 제한 동작은 브라우저에서만 동작합니다. 브라우저를 제외한 방법으로 접속하여 데이터를 전송할 때는 `maxlength` 적용받지 않습니다.

입력 패턴

HTML5의 입력 패턴 속성을 통하여 데이터 형식을 제한할 수 있습니다. 패턴 속성과 정규표현 방법을 통하여 입력되는 데이터를 필터링합니다.

```
<input type="password" name="passwd" pattern="[0~9A-Za-z]">
```

최대, 최소

HTML5에서는 `min` 또는 `max` 속성값을 통하여 입력되는 데이터의 최소치와 최대치를 지정할 수도 있습니다.

초기값 지정

`input`의 텍스트 속성을 사용 시 사용자에게 보여주는 초기값 및 기본값을 미리 출력해서 보여줄 수 있습니다. 초기값은 약간 흐린 회색 계열의 문자로 출력함으로써 사용자 입력값과 구분합니다.

```
<input type="text" name="username" placeholder="이름을 입력해 주세요">
```

자동 완성 추천

검색 엔진이나 특정 값을 입력할 때 하단에 추천하는 단어들을 나타내는 경우를 본 적이 있을 것입니다. HTML5의 autocomplete 기능을 이용하면 쉽게 구현할 수 있습니다.

```
<input type="text" name="country" placeholder="국가를 입력해 주세요"
autocomplete="on" list="local">
<datalist id="local">
  <option value="대한민국">
  <option value="미국">
  <option value="일본">
</datalist>
```

17.2.2 hidden

text 속성은 브라우저 화면에 사용자 입력 박스를 출력합니다. 그와 반대로 hidden 속성은 서버로 데이터를 전송하는 값이지만 화면에는 출력되지 않습니다. 즉, 숨겨진 text 속성과 같습니다.

```
<input type="hidden" name="mode" value="editup">
```

17.2.3 textarea

HTML teatarea 요소는 input 요소와 달리 많은 양의 텍스트를 입력받을 수 있습니다.

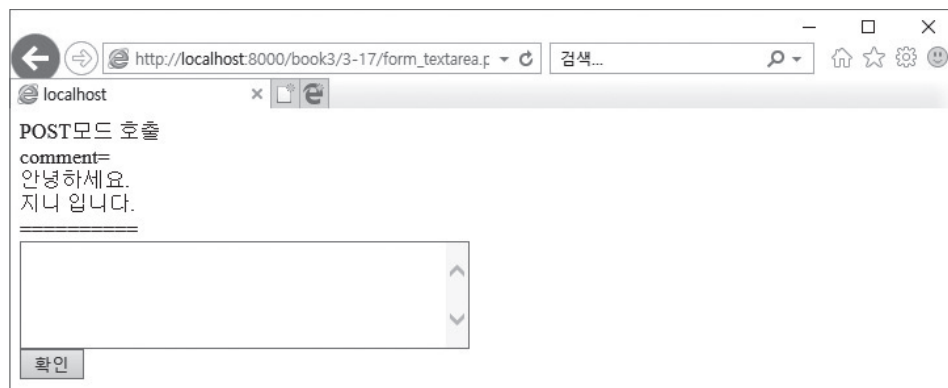
```
<textarea name="comment" rows="5" cols="40"></textarea>
```

textarea는 웹 페이지에서 글을 작성하는 본문 등 많이 이용합니다. textarea는 많은 양과 다양한 입력을 고려하여 데이터 안에 숨겨진 기호나 패턴 등을 복합적으로 판별해야 합니다.

예제 파일 | form_textarea.php

```
1  <?php
2      // 서밋 처리 시 데이터 값 확인
3      if ($_SERVER["REQUEST_METHOD"] == "POST") {
4          echo "POST 모드 호출 <br>";
5
6          // 다음 줄 n\를 <br>로 변환하여 출력함.
7          echo "comment= <br>" . nl2br($_POST['comment']);
8          echo "<br>=====<br>";
9      }
10
11     // HTML form을 출력합니다.
12
13     echo "<form name='jiny' method='post' action='". $_SERVER['PHP_
14         SELF']."'>";
15
16     echo "<textarea name=\"comment\" rows=\"5\" cols=\"40\"></textarea>";
17
18     echo "<br>";
19     echo "<input type='submit' value='확인'>";
20     echo "</form>";
21 ?>
```

화면 출력



POST모드 호출

comment=

안녕하세요.

지니 입니다.

확인

17.2.4 select

select 요소는 input 요소 다음으로 가장 많이 사용하는 html form 구성입니다. select 는 드롭 다운 형태로 여러 개의 데이터 값 중에 1개 또는 다수의 데이터를 선택할 수 있습니다.

```
<select name="이름">
    <option value="옵션값1">옵션명1</option>
    <option value="옵션값2">옵션명2</option>
    ...
</select>
```

예제 파일 | **form_select1.php**

```
1  <?php
2      // 서밋 처리 시 데이터 값 확인
3      if ($_SERVER["REQUEST_METHOD"] == "POST") {
4          echo "POST 모드 호출 <br>";
5          echo "country = " . $_POST['country'] . "<br>";
6          echo "=====<br>";
7      }
8
9      // HTML form을 출력합니다.
10
11     echo "<form name='jiny' method='post' action='".$_SERVER['PHP_
12         SELF']."'>";
13
14     $body = "<select name=\"country\">";
15     $body .= "<option value=\"kr\">대한민국</option>";
16     $body .= "<option value=\"jp\">일본</option>";
17     $body .= "<option value=\"cn\">중국</option>";
18     $body .= "</select>";
19
20     // 기본 선택 값을 selected로 처리
21     if ($_POST['country'])
22         $body = str_replace(
23             "value=\"".$_POST['country'].\"",
24             "value=\"".$_POST['country']."\" selected",
```

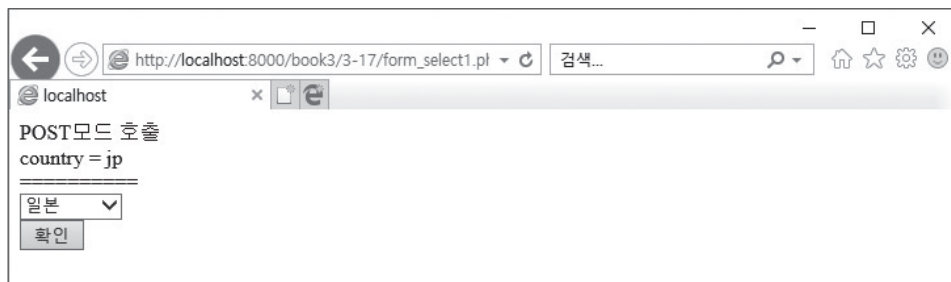


```

24         $body);
25
26     echo $body;
27
28     echo "<br>";
29     echo "<input type='submit' value='확인'>";
30     echo "</form>";
31
32 ?>

```

화면 출력



select는 기본적으로 선택한 1개의 값을 지정할 수 있습니다. 하지만 여러 개의 항목 선택이 필요한 경우에는 `<select multiple name="이름[]">` 형태로 이름을 배열 형태로 `multiple` 속성을 사용하면 됩니다.

예제 파일 | form_select2.php

```

1  <?php
2      // 서밋 처리 시 데이터 값 확인
3      if ($_SERVER["REQUEST_METHOD"] == "POST") {
4          echo "POST모드 호출 <br>";
5          $COUNTRY = $_POST['country'];
6
7          for($i=0;$i<count($COUNTRY);$i++) {
8              echo $i . "=" . $COUNTRY[$i] . "<br>";
9          }
10         echo "=====<br>";
11     }
12
13     // HTML form을 출력합니다.

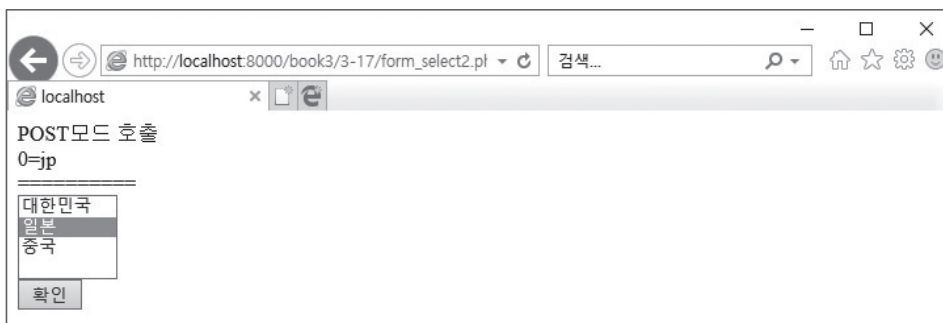
```

```

14
15     echo "<form name='jiny' method='post' action='".$_SERVER['PHP_
        SELF'].>";
16
17     $body = "<select name=\"country[]\" multiple>";
18     $body .= "<option value=\"kr\">대한민국</option>";
19     $body .= "<option value=\"jp\">일본</option>";
20     $body .= "<option value=\"cn\">중국</option>";
21     $body .= "</select>";
22
23     // 기본 선택 값을 selected로 처리
24     if ($_POST['country']) {
25         $COUNTRY = $_POST['country'];
26
27         for($i=0;$i<count($COUNTRY);$i++) {
28             $body = str_replace(
29                 "value=\"".$_COUNTRY[$i].\"",
30                 "value=\"".$_COUNTRY[$i].\" \" selected",
31                 $body);
32
33         }
34     }
35
36     echo $body;
37
38     echo "<br>";
39     echo "<input type='submit' value='확인'>";
40     echo "</form>";
41
42     ?>

```

화면 출력



17.2.5 checkbox

폼 요소 중 체크박스는 어떠한 항목에 동의 등을 구할 때 많이 사용하는 구성 요소입니다.

```
<input type='checkbox' name='이름'>
```

위와 같이 HTML 코드 입력하면 form의 데이터는 슈퍼 변수 \$_GET[] 또는 \$_POST[] 배열 변수를 통해서 값을 가져올 수 있습니다.

이때 지정한 name="이름" 값의 배열의 선택 키 값으로 사용됩니다. 만일 체크박스의 초기값을 "선택된"으로 하고자 할 때는 checked 키워드를 추가하면 됩니다.

체크박스가 선택된 값은 on 문자열 값으로 전달됩니다.

예제 파일 | **form_check1.php**

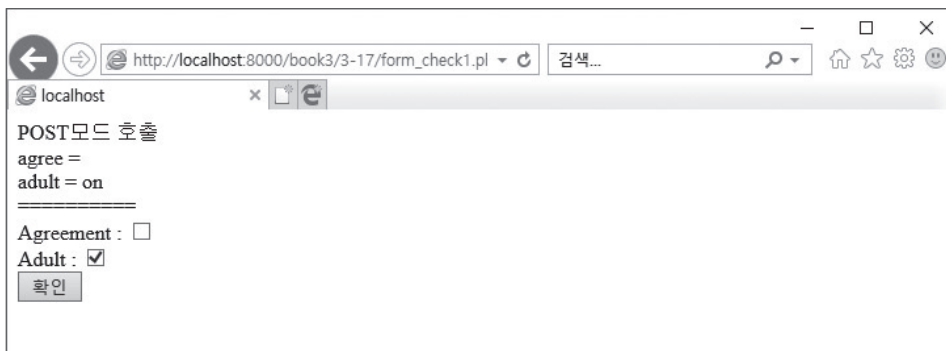
```
1  <?php
2      // 서버 및 처리 시 데이터 값 확인
3      if ($_SERVER["REQUEST_METHOD"] == "POST") {
4          echo "POST모드 호출 <br>";
5          echo "agree = " . $_POST['agree'] . "<br>";
6          echo "adult = " . $_POST['adult'] . "<br>";
7          echo "=====<br>";
8      }
9
10     // HTML form을 출력합니다.
11
12     echo "<form name='jiny' method='post' action='". $_SERVER['PHP_
13         SELF']. ">";
14
15     echo "Agreement : ";
16     if ($_POST['agree']) {
17         echo "<input type='checkbox' name='agree' checked>";
18     } else {
19         echo "<input type='checkbox' name='agree'>";
20     }
21     echo "<br>";
```

```

21
22     echo "Adult : ";
23     if ($_POST['adult']) {
24         echo "<input type='checkbox' name='adult' checked>";
25     } else {
26         echo "<input type='checkbox' name='adult'>";
27     }
28
29     echo "<br>";
30     echo "<input type='submit' value='확인'>";
31     echo "</form>";
32
33 ?>

```

화면 출력



위의 예제1은 체크박스가 하나에 1개의 이름으로 각각 할당되어 있습니다. 다음 예제는 체크박스 이름을 배열 형태로 지정한 후에 value 속성을 이용하여 값을 할당합니다.

예제 파일 | form_check2.php

```

1 <?php
2     // 서밋 처리 시 데이터 값 확인
3     if ($_SERVER["REQUEST_METHOD"] == "POST") {
4         echo "POST모드 호출 <br>";
5         $TID = $_POST['TID'];
6
7         for($i=0;$i<count($TID);$i++) {
8             echo $i . "=" . $TID[$i] . "<br>";

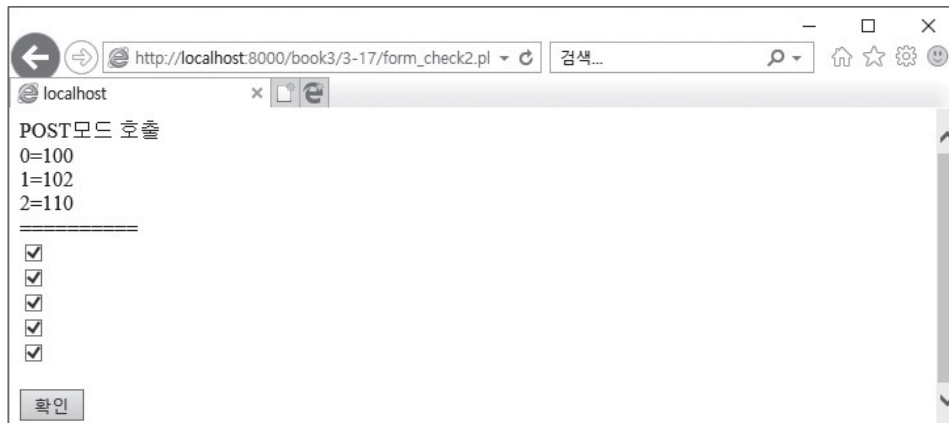
```

```

9         }
10
11         echo "=====<br>";
12     }
13
14     // HTML form을 출력합니다.
15
16     echo "<form name='jiny' method='post' action='".$_SERVER['PHP_
17     SELF']."'>";
18
19     // 체크 값의 이름이 배열 형태로 지정
20     echo "<input type='checkbox' name='TID[]' value='100' checked> <br>";
21     echo "<input type='checkbox' name='TID[]' value='102' checked> <br>";
22     echo "<input type='checkbox' name='TID[]' value='105' checked> <br>";
23     echo "<input type='checkbox' name='TID[]' value='107' checked> <br>";
24     echo "<input type='checkbox' name='TID[]' value='110' checked> <br>";
25
26     echo "<br>";
27     echo "<input type='submit' value='확인'>";
28     echo "</form>";
29     ?>

```

화면 출력



예제2에서 체크박스는 TID라는 이름의 배열로 작성되어 있습니다. 또한 각각의 체크박스는 value 속성을 통하여 고유값들을 가지고 있습니다.

서밋을 통하여 PHP로 값이 전달된 경우 배열로 체크박스는 배열 형태로 값을 전달받게 됩니다. 선택한 체크박스의 값만을 배열화하여 전달합니다.

이러한 유형은 리스트 목록 등에서 여러 개의 데이터를 선택하여 FORM으로 전달하는 용도로 많이 사용됩니다.

17.2.6 radio box

HTML의 라디오 요소는 여러 개의 선택 값 중에 하나만 입력받을 수 있는 요소입니다.

```
<input type="radio" name="sex" value="female">Female  
<input type="radio" name="sex" value="male">Male
```

라디오 버튼의 name="이름" 부분은 동일한 값을 가집니다. 하지만 속성 값은 각각의 다른 값을 갖게 됩니다.

예제 파일 | form_radio.php

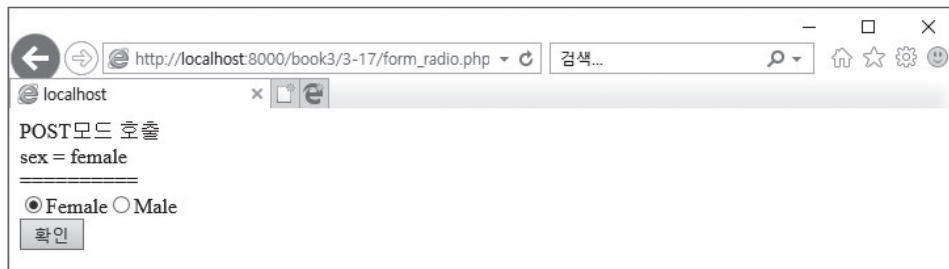
```
1  <?php  
2      // 서밋 처리 시 데이터 값 확인  
3      if ($_SERVER["REQUEST_METHOD"] == "POST") {  
4          echo "POST모드 호출 <br>";  
5          echo "sex = " . $_POST['sex'] . "<br>";  
6          echo "=====<br>";  
7      }  
8  
9      // HTML form을 출력합니다.  
10  
11     echo "<form name='jiny' method='post' action='".$_SERVER['PHP_  
12         SELF']."'>";  
13  
14     if ($_POST['sex'] == "female") {  
15         echo "<input type=\"radio\" name=\"sex\" value=\"female\"  
16             checked>Female";  
17         echo "<input type=\"radio\" name=\"sex\" value=\"male\">Male";  
18     } else if ($_POST['sex'] == "male") {
```

```

18     echo "<input type=\"radio\" name=\"sex\" value=\"female\">Female";
19     echo "<input type=\"radio\" name=\"sex\" value=\"male\"
    checked>Male";
20
21 } else {
22     echo "<input type=\"radio\" name=\"sex\" value=\"female\">Female";
23     echo "<input type=\"radio\" name=\"sex\" value=\"male\">Male";
24 }
25
26 echo "<br>";
27 echo "<input type='submit' value='확인'>";
28 echo "</form>";
29
30 ?>

```

화면 출력



17.2.7 file

메일 작성이나 게시판에 글과 함께 첨부하는 파일 등을 지정하여 서버로 전송할 수 있습니다. 클라이언트 브라우저에서 서버로 파일을 전송하기 위해서는 form의 type="file"을 사용하면 됩니다.

```
<input type="file" name="upload">
```

HTML에서 파일 전송을 위해서는 전송 메서드 타입이 POST로 되어 있어야 합니다. 또한 바이너리 데이터를 함께 전송하기 위해서 속성 enctype="multipart/form-data"가 설정되어야 합니다.

HTML의 file 요소를 선택하면 컴퓨터는 서버로 전송할 파일을 선택하는 팝업 창을 브라우저가 실행시킵니다.

또한 PHP에서 파일을 업로드하여 전송하기 위해서는 약간의 php.ini 환경 설정이 필요합니다. php.ini 파일에서 file_uploads 부분을 ON으로 변경해야 합니다.

file_uploads = On

또한 파일 용량 제한도 가능합니다. 보다 자세한 것은 환경 설정 부분을 참조하면 됩니다.

예제 파일 | **form_upload.php**

```
1  <?php
2      // 서밋 처리 시 데이터 값 확인
3      if ($_SERVER["REQUEST_METHOD"] == "POST") {
4          echo "POST 모드 호출 <br>";
5          print_r($_FILES['upload']);
6          echo "<br>";
7
8          // 업로드 디렉터리 지정
9          $target_dir = "./uploads";
10         if (!is_dir($target_dir)) {
11             mkdir($target_dir);
12         }
13
14         $target_file = $target_dir . "/" . basename($_FILES["upload"]
15         ["name"]);
16         echo "저장파일명 = ". $target_file . "<br>";
17
18         $uploadOk = true;
19
20         // 업로드할 파일이 존재하는지 검사
21         if (file_exists($target_file)) {
22             echo "이미 존재하는 파일명입니다.";
23             $uploadOk = false;
24         }
25
26         // 파일 사이즈 제한
27         if ($_FILES["upload"]["size"] > 500000) {
28             echo "파일 사이즈가 너무 큼니다.";
29         }
30     }
```

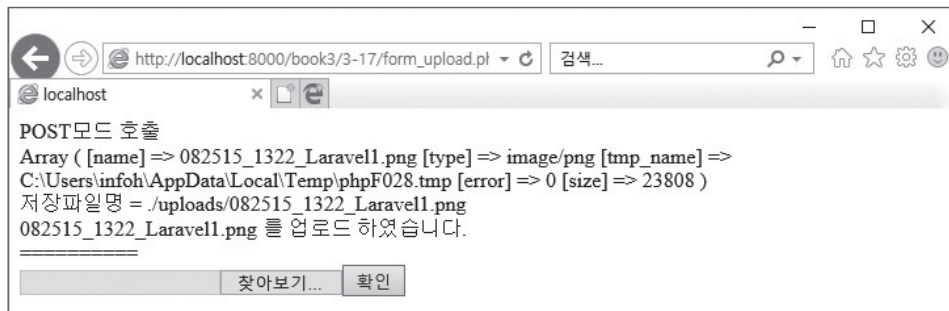


```

28         $uploadOk = false;
29     }
30
31     // 파일 등록 및 복사
32     if ($uploadOk) {
33         if (move_uploaded_file($_FILES["upload"]["tmp_name"], $target_
34             file)) {
35             echo basename( $_FILES["upload"]["name"]). "를 업로드
36                 하였습니다.";
37         } else {
38             echo "파일 업로드 오류";
39         }
40     } else {
41         echo "죄송합니다. 파일 등록을 할 수 없습니다.";
42     }
43
44     echo "<br>=====<br>";
45
46     // HTML form을 출력합니다.
47
48     echo "<form name='jiny' method='post' enctype='multipart/form-data\"
49         action='\"".$_SERVER['PHP_SELF']."'>";
50
51     echo "<input type='file\" name='\"upload\" id='\"upload\"\">";
52     echo "<input type='submit' value='확인'>";
53     echo "</form>";
54
55     ?>

```

화면 출력



위의 파일 업로드 실험은 클라이언트의 웹 브라우저에서 서버로 파일을 전송합니다.

form의 file 요소로 파일을 선택하여 전송합니다. 서버의 PHP에서는 \$_FILES 배열 변수로 파일의 정보를 받아들일 수 있습니다.

\$_FILES 배열에는 파일의 관련된 다양한 값의 정보를 가지고 있습니다. 또한 전송받은 파일은 임시폴더 안에 저장됩니다. move_uploaded_file() 함수를 통하여 실제 저장 장소로 이동합니다.

파일을 이동 시에는 파일의 정보와 저장 디렉터리, 용량 등을 사전에 체크하여 필터링할 수 있습니다.

17.3 다운로드

서버상에 있는 파일을 사용자 PC에 다운로드할 수 있도록 링크를 생성할 수 있습니다. 서버상의 파일을 간단하게 다운로드하는 방법은 header() 함수를 통하여 브라우저에 header를 설정하여 전송하는 방법입니다.

| 내장 함수 |

```
void header ( string $string [, bool $replace = true [, int $http_response_code ] ] )
```

내장 함수 header() 함수는 사용자 브라우저로 데이터를 전송합니다. header() 함수는 다른 HTML 코드보다 우선적으로 실행되어야 합니다.

예제 파일 | download.php

```
1  <?php
2
3      $server_filename= "./sample.csv";
4
5      // $server_filename에서 파일명 부분만 추출합니다.
6      $filename = substr($server_filename, strpos($server_filename, '/') + 1);
```

```

7
8 // 서버의 파일명을 확인합니다.
9 // 파일이 존재하지 않거나 읽을 수가 없을 때 처리를 중단합니다.
10 if (!file_exists($server_filename) || !is_readable($server_filename)) {
11     die("파일을 읽을 수 없습니다.");
12 } else {
13     // 파일 사이즈를 체크합니다.
14     if (($filesize = filesize($server_filename)) == 0) {
15         die("파일 크기가 0 바이트 입니다.\n");
16     } else {
17         if (($fp = @fopen($server_filename, 'rb')) === false) {
18             die("파일을 읽을 수 없습니다.\n");
19         }
20     }
21 }
22
23
24 // UTF-8 파일명이 깨지지 않도록 하기 위해서
25 // 파일명에 사용할 수 없는 문자를
26 // 모두 제거합니다. 또는 다른 안전한 문자로 치환합니다. (RFC2231/5987 표준)
27 $illegal = array('\', '/', '<', '>', '{', '}', ':', ';', '|', '"',
28     '~', '`', '@', '#', '$', '%', '^', '&', '*', '?');
29 $replace = array('_', '(', ')', '(', ')', '-', ',', '_ ', ' ', '_ ',
30     '\ ', '_ ', '_ ', '_ ', '_ ', '_ ', '_ ', ' ', '');
31 $filename = str_replace($illegal, $replace, $filename);
32 $filename = preg_replace('/([\x00-\x1f\x7f\xff]+)/', '', $filename);
33
34 // 유니코드 공백 문자를 일반 공백 문자(0x20)로 변경을 합니다.
35 $filename = trim(preg_replace('/[\pZ\pC]+/u', ' ', $filename));
36
37 // 중복된 점, 대체 문자를 정리합니다.
38 $filename = trim($filename, '._-');
39 $filename = preg_replace('/__+/', '_', $filename);
40 if ($filename === '') {
41     die("파일명 이름 변환처리 실패");
42 }
43
44 // 브라우저의 User-Agent 값 처리
45 USER_AGENT = isset($_SERVER['HTTP_USER_AGENT']) ? $_SERVER['HTTP_
46     USER_AGENT'] : '';

```

```

45 $old_ie = (bool)preg_match('#MSIE [3-8]\.#', $USER_AGENT);
46
47 // 브라우저별 헤더 파일명을 정리합니다.
48 if (preg_match('/^[a-zA-Z0-9_-]+$/ ', $filename)) {
49     // 파일명에 숫자와 영문으로만 된 경우
50     // 브라우저 상관없이 파일명을 헤더로 처리를 합니다.
51     $header = 'filename="' . $filename . '"';
52
53 } elseif ($old_ie || preg_match('#Firefox/(\d+)\.#', $USER_AGENT,
54 $matches)
55     && $matches[1] < 5) {
56     // IE 9 미만 , Firefox 5 미만의 경우 처리.
57     $header = 'filename="' . rawurlencode($filename) . '"';
58
59 } elseif (preg_match('#Chrome/(\d+)\.#', $USER_AGENT, $matches)
60     && $matches[1] < 11) {
61     // Chrome 11 미만의 경우.
62     $header = 'filename=' . $filename;
63
64 } elseif (preg_match('#Safari/(\d+)\.#', $USER_AGENT, $matches)
65     && $matches[1] < 6) {
66     // Safari 6 미만의 경우.
67     $header = 'filename=' . $filename;
68
69 } elseif (preg_match('#Android #', $USER_AGENT, $matches)) {
70     // 안드로이드 브라우저의 경우.
71     $header = 'filename="' . $filename . '"';
72
73 } else {
74     // RFC2231/5987 표준 준수
75     $header = "filename*=UTF-8'" . rawurlencode($filename) .
76     ' ; filename="' . rawurlencode($filename) . '"';
77 }
78
79 // 캐시 처리
80 // 캐시 처리를 위해서 Cache-Control, Expires 등의 헤더를 적용합니다.
81 if (!$expires) {
82     // 캐싱 금지
83     if ($old_ie) {
84         // 익스플로러 8 이하 버전
85         // SSL 사용시 no-cache 및 pragma 헤더 오류를 발생

```

```

85         header('Cache-Control: private,
86             must-revalidate,
87             post-check=0,
88             pre-check=0');
89         header('Expires: Sat, 01 Jan 2000 00:00:00 GMT');
90     } else {
91         header('Cache-Control: no-store,
92             no-cache,
93             must-revalidate,
94             post-check=0,
95             pre-check=0');
96         header('Expires: Sat, 01 Jan 2000 00:00:00 GMT');
97     }
98
99     } else {
100         // 캐싱 허용
101         header('Cache-Control: max-age=' . (int)$expires);
102         header('Expires: ' . gmdate('D, d M Y H:i:s', time() + (int)$expires)
103             ' GMT');
104     }
105
106     // 다운로드 이어받기 헤더 처리
107     // Range 헤더 자동 감지, Accept-Ranges 헤더 자동 생성
108     if (isset($_SERVER['HTTP_RANGE']) && preg_match('/^bytes=(\d+)-/',
109         $_SERVER['HTTP_RANGE'], $matches)) {
110         $range_start = $matches[1];
111
112         if ($range_start < 0 || $range_start > $filesize) {
113             header('HTTP/1.1 416 Requested Range Not Satisfiable');
114             return false;
115         }
116
117         header('HTTP/1.1 206 Partial Content');
118         header('Content-Range: bytes ' . $range_start . '-' . ($filesize - 1) .
119             '/' . $filesize);
120         header('Content-Length: ' . ($filesize - $range_start));
121     } else {
122         $range_start = 0;
123         header('Content-Length: ' . $filesize);

```

```

123
124     }
125
126     // 그 외 헤더를 전송한다.
127     header('Accept-Ranges: bytes');
128     header('Content-Type: application/octet-stream');
129     header('Content-Disposition: attachment; ' . $header);
130
131     // 출력 버퍼를 초기화.
132     // 대용량 파일 다운로드 시 메모리 누수를 방지합니다.
133     while (ob_get_level()) {
134         ob_end_clean();
135     }
136
137     // 파일을 64KB 단위로 분할 전송
138     // readfile() 함수의 메모리 누수 방지
139     $block_size = 16 * 1024;
140     $speed_sleep = $speed_limit > 0 ? round(($block_size / $speed_limit /
141         1024) * 1000000) : 0;
142
143     $buffer = '';
144     if ($range_start > 0) {
145         fseek($fp, $range_start);
146         $alignment = (ceil($range_start / $block_size) * $block_size) -
147             $range_start;
148         if ($alignment > 0) {
149             $buffer = fread($fp, $alignment);
150             echo $buffer;
151             unset($buffer);
152             flush();
153         }
154     }
155
156     while (!feof($fp)) {
157         $buffer = fread($fp, $block_size);
158         echo $buffer;
159         unset($buffer);
160         flush();
161         usleep($speed_sleep);

```

```
162     }  
163  
164     fclose($fp);  
165  
166 ?>
```


18

스크립트

18.1 종료

내장 함수 `exit()`는 스크립트를 종료합니다.

| 내장 함수 |

```
void exit ([ string $status ] )
```

`status`가 문자열이면 종료 직전의 문자열을 출력합니다. `status`가 정수일 때는 값이 화면에 출력되지는 않습니다. 정수값은 0~255 사이의 값을 가집니다. 만일 0의 값은 정상적인 종료를 의미합니다.

예제 파일 | `exit.php`

```
1  <?php
2
3      //일반적인 프로그램 종료
4      exit;
```

```

5      exit();
6      exit(0);
7
8      // 문자열을 출력하고 종료합니다.
9      exit("finish");
10
11     //exit with an error code
12     exit(1);
13     exit(0376); //octal
14
15     ?>

```

| 내장 함수 |

```
void die ( [ string $status ] );
```

내장 함수 `die()`는 스크립트를 종료합니다. `exit()` 함수와 동일한 동작을 수행합니다.

예제 파일 | `die.php`

```

1  <?php
2      die ("데이터베이스 접속 실패");
3  ?>

```

| 내장 함수 |

```
void register_shutdown_function ( callable $callback [, mixed $parameter [, mixed $... ] ] )
```

스크립트 실행이 끝나거나 `exit()`가 호출된 후에 실행될 콜백을 등록합니다.

예제 파일 | `register_shutdown_function.php`

```

1  <?php
2      function shutdown() {
3          // 스크립트가 완료되기 전에 마지막 작업을 수행할 수 있습니다.
4          echo 'Script executed with success', PHP_EOL;

```

```

5     }
6
7     register_shutdown_function('shutdown');
8
9     ?>

```

화면 출력

Script executed with success

18.2 코드 실행

내장 함수 eval()은 문자열로 된 소스 코드를 실행합니다. eval() 함수는 해킹의 용도로 사용될 수 있기 때문에 주의하여 사용해야 합니다.

| 내장 함수 |

```
mixed eval ( string $code )
```

예제 파일 | eval.php

```

1  <?php
2      $string = 'cup';
3      $name = 'coffee';
4
5      // 작은따옴표는 일반 텍스트로 문자를 인식합니다.
6      // $string, $name는 변수명 자체를 화면에 출력합니다.
7      $str = 'This is a $string with my $name in it.';
8      echo $str. "<br>";
9
10     // PHP 코드 처리됩니다.
11     $code = "\$aaa = \"$str\"";
12     eval($code);
13
14     echo $aaa;
15 ?>

```

화면 출력

This is a \$string with my \$name in it.
This is a cup with my coffee in it.

18.3 접속 상태

PHP 스크립트는 내부적으로 세 가지의 접속 상태가 존재합니다.

0 – NORMAL

1 – ABORTED

2 – TIMEOUT

18.3.1 connection_status

내장 함수 `connection_status()`는 연결 상태 반환합니다.

| 내장 함수 |

```
int connection_status ( void )
```

NORMAL 상태는 보통 스크립트를 실행하고 정상적인 동작을 할 때입니다.

예제 파일 | `connection_status.php`

```
1  <?php
2      $status = connection_status();
3      if ($status!=0) {
4          die;
5      } else {
6          echo $status."<br>";
7          echo "접속상태 입니다.";
8      }
9
10  ?>
```

화면 출력

0

접속상태 입니다.

18.3.2 ignore_user_abort

ABORTED 상태는 스크립트를 동작하고 있는 중에 접속 클라이언트의 사용자가 임의적으로 중지 버튼을 누르거나 접속을 차단했을 때의 상태입니다. 또는 시간 제한에 의하여 타임아웃 상태가 된 경우입니다.

기본적으로 ABORTED 상태가 되면 스크립트는 중단합니다. 하지만 ABORTED 상태가 되어도 실행 요청한 스크립트가 지속적으로 동작하여 정상 수행을 다 마치길 바랄 때도 있을 것입니다.

이런 경우 PHP.ini 설정의 ignore_user_abort를 변경하거나 아파치와 같은 웹 서버에서 설정을 변경할 수 있으며, 또는 PHP 소스상에서 ignore_user_abort() 함수를 통하여 설정할 수도 있습니다.

ignore_user_abort() 함수는 클라이언트 연결 끊기가 스크립트 실행을 중단해야 하는지 여부를 설정합니다.

| 내장 함수 |

```
int ignore_user_abort ([ bool $value ] )
```

예제 파일 | ignore_user_abort.php

```
1  <?php
2      echo 'PHP connection';
3
4      ignore_user_abort(true);
5      set_time_limit(0);
6
7      while (1) {
```

```

8
9      // 접속상태 실패
10     if (connection_status() != CONNECTION_NORMAL)
11     {
12         break;
13     }
14
15     // 10초간 지연
16     sleep(10);
17 }
18
19 ?>

```

18.3.3 connection_aborted

보통 스크립트가 종료될 때 `register_shutdown_function()`을 통하여 종료 처리를 할 수 있습니다. 하지만 클라이언트의 연결이 끊겨서 스크립트를 종료해야 할 경우도 있습니다.

| 내장 함수 |

```
int connection_aborted ( void )
```

이런 경우 `connection_aborted()` 함수를 통하여 별도의 처리를 수행할 수 있습니다.

| 내장 함수 |

```
bool set_time_limit ( int $seconds )
```

내장 함수 `set_time_limit()`는 최대 실행 시간을 제한합니다.

18.4 지연 함수, 실행 시간 설정

18.4.1 지연

내장 함수 `sleep()`은 딜레이를 적용합니다. 입력되는 값은 초 단위로 입력합니다.

| 내장 함수 |

```
int sleep ( int $seconds )
```

예제 파일 | `sleep.php`

```
1  <?php
2
3      // 현재 시간
4      echo date('h:i:s') . "<br>";
5
6      // 10초간 지연
7      sleep(10);
8
9      // 시간 다시 표시
10     echo date('h:i:s') . "<br>";
11
12  ?>
```

화면 출력

09:01:09

09:01:19

| 내장 함수 |

```
void usleep ( int $micro_seconds )
```

내장 함수 `usleep()`은 마이크로 초 단위로 지연합니다.

예제 파일 | **usleep.php**

```
1  <?php
2
3      // 현재 시간
4      echo date('h:i:s') . "<br>";
5
6      // 2초간 대기합니다.
7      usleep(2000000);
8
9      // 시간 다시 표시
10     echo date('h:i:s') . "<br>";
11
12  ?>
```

화면 출력

09:05:09

09:05:11

18.4.2 실행 시간

스크립트의 실행의 최대 시간을 설정할 수 있습니다. 작업이 많이 필요한 스크립트의 경우 실행 시간을 조정할 수 있습니다.

| 내장 함수 |

```
void set_time_limit ( int $seconds )
```

예제 파일 | **set_time_limit-01.php**

```
1  <?php
2      // 스크립트 최대 실행 시간 100초
3      set_time_limit(100);
4      echo "스크립트 시간 100초<br>";
5
6      while ($i<=50) {
7          echo "sleep=$i , ";
```



```

8         sleep(1);
9         $i++;
10    }
11    ?>

```

화면 출력

스크립트 시간 100초

sleep= , sleep=1 , sleep=2 , sleep=3 , sleep=4 , sleep=5 , sleep=6 , sleep=7 , sleep=8 , sleep=9 , sleep=10 , sleep=11 , sleep=12 , sleep=13 , sleep=14 , sleep=15 , sleep=16 , sleep=17 , sleep=18 , sleep=19 , sleep=20 , sleep=21 , sleep=22 , sleep=23 , sleep=24 , sleep=25 , sleep=26 , sleep=27 , sleep=28 , sleep=29 , sleep=30 , sleep=31 , sleep=32 , sleep=33 , sleep=34 , sleep=35 , sleep=36 , sleep=37 , sleep=38 , sleep=39 , sleep=40 , sleep=41 , sleep=42 , sleep=43 , sleep=44 , sleep=45 , sleep=46 , sleep=47 , sleep=48 , sleep=49 , sleep=50 ,

기본 설정값은 30초 또는 php.ini에 정의된 max_execution_time 값입니다. 만일, 지정한 값보다 시간을 초과할 때는 치명적인 오류를 반환합니다.

스크립트를 실행 도중에 set_time_limit()를 실행하면 제한 시간 카운터를 0에서 다시 시작합니다. 예로 기본 timeout이 30초이고, 25초 되는 시점에 set_time_limit(20)과 같이 호출하게 되면 총 시간은 45초 동안 실행됩니다.

예제 파일 | [set_time_limit-02.php](#)

```

1  <?php
2
3      set_time_limit(20);
4
5      while ($i<=10) {
6          echo "i=$i ";
7          sleep(100);
8          $i++;
9      }
10
11  ?>

```

화면 출력

i=0 i=1 i=2 i=3 i=4 i=5 i=6 i=7 i=8 i=9 i=10

18.5 스크립트 정보

PHP 스크립트의 정보를 읽어올 수 있습니다. 이와 관련된 몇 가지 내장 함수들을 지원합니다.

| 내장 함수 |

```
string get_current_user ( void )
```

내장 함수 `get_current_user()`는 현재 PHP 스크립트를 실행하고 있는 소유자(owner)를 확인할 수 있습니다.

예제 파일 | `get_current_user.php`

```
1 <?php
2     echo "스크립트 owner: " . get_current_user();
3 ?>
```

화면 출력

스크립트 owner: infoh

| 내장 함수 |

```
int getlastmod ( void )
```

내장 함수 `getlastmod()`는 현재 스크립트의 최종 수정 일자를 확인할 수 있습니다. 스크립트 파일 일자를 확인하여 업데이트와 같은 갱신 처리할 때 등 유용하게 사용할 수 있습니다.

예제 파일 | `getlastmod.php`

```
1 <?php
2
3     $scriptDate = date ("F d Y H:i:s.", getlastmod());
```

```

4     echo "최종 수정일: " . $scriptDate . "<br>";
5
6     if (date ("Y-m-d", getlastmod()) <= "2017-10-03") {
7         echo "스크립트가 최신 버전이 아닙니다. 업그레이드를 해주세요<br>";
8     } else {
9         echo "최신<br>";
10    }
11
12 ?>

```

화면 출력

최종 수정일: June 21 2017 08:04:13.
 스크립트가 최신 버전이 아닙니다. 업그레이드를 해주세요

| 내장 함수 |

```
int getmygid ( void )
```

내장 함수 getmygid()는 PHP 스크립트 소유자의 GID를 가지고 옵니다.

19

정보

PHP는 내장 함수를 통하여 PHP의 버전 및 설정 상태 등의 다양한 정보를 읽어서 처리할 수 있습니다.

19.1 시스템

내장 함수 `phpversion()`은 현재 PHP 버전을 가져옵니다.

| 내장 함수 |

```
string phpversion ([ string $extension ] )
```

예제 파일 | `phpversion.php`

```
1  <?php
2      echo phpversion();
3  ?>
```

화면 출력

7.1.4

| 내장 함수 |

```
mixed version_compare ( string $version1 , string $version2 [, string $operator ] )
```

내장 함수 `version_compare()`는 2개의 PHP 버전 번호를 비교합니다.

예제 파일 | `version_compare.php`

```
1  <?php
2      if (version_compare(PHP_VERSION, '7.0.0') >= 0) {
3          echo 'I am at least PHP version 7.0.0, my version: ';
4          echo PHP_VERSION."<br>";
5      }
6
7      if (version_compare(PHP_VERSION, '5.3.0') >= 0) {
8          echo 'I am at least PHP version 5.3.0, my version: ';
9          echo PHP_VERSION."<br>";
10     }
11
12  ?>
```

화면 출력

I am at least PHP version 7.0.0, my version: 7.1.4
I am at least PHP version 5.3.0, my version: 7.1.4

| 내장 함수 |

```
string zend_version ( void )
```

내장 함수 `zend_version()`은 Zend 엔진의 버전을 가져옵니다.

예제 파일 | **zend_version.php**

```
1 <?php
2     echo "Zend engine version: " . zend_version();
3 ?>
```

화면 출력

Zend engine version: 3.1.0

| 내장 함수 |

```
int zend_thread_id ( void )
```

내장 함수 `zend_thread_id()`는 현재 스레드의 고유 ID를 반환합니다. 이 함수는 PHP가 ZTS(Zend Thread Safety) 지원 및 디버그 모드(`--enable-debug`)로 빌드된 경우에만 사용할 수 있습니다.

예제 파일 | **zend_thread_id.php**

```
1 <?php
2     $thread_id = zend_thread_id();
3     echo 'Current thread id is: ' . $thread_id;
4 ?>
```

| 내장 함수 |

```
string php_uname ([ string $mode = "a" ] )
```

내장 함수 `php_uname()`은 PHP가 실행되는 운영체제 정보를 반환합니다.

예제 파일 | **php_uname.php**

```
1 <?php
2     echo php_uname() . "<br>";
3     echo PHP_OS;
4 ?>
```

화면 출력

Windows NT LAPTOP-M0820HEF 10.0 build 15063 (Windows 10) i586
WINNT

19.2 정보 함수

PHP 환경 설정과 정보들을 내장 함수를 통하여 읽어올 수 있습니다.

| 내장 함수 |

```
bool phpinfo ([ int $what = INFO_ALL ] )
```

내장 함수 `phpinfo()`는 PHP의 설정에 관한 정보를 출력합니다. 전반적인 PHP의 설치 환경 정보를 웹 페이지를 통하여 출력할 수 있습니다. 또는 지정 모듈만 선택하여 출력할 수도 있습니다.

- `INFO_GENERAL`
- `INFO_CREDITS`
- `INFO_CONFIGURATION`
- `INFO_MODULES`
- `INFO_ENVIRONMENT`
- `INFO_VARIABLES`
- `INFO_LICENSE`
- `INFO_ALL`

예제 파일 | `phpinfo.php`

```
1 <?php
2
3     // 전체 정보를 출력합니다.
4     phpinfo();
5
```



```

6      // 지정한 모듈 정보만 출력합니다.
7      phpinfo(INFO_MODULES);
8
9      ?>

```

| 내장 함수 |

```
string get_cfg_var ( string $option )
```

내장 함수 `get_cfg_var()`은 PHP의 설정 옵션값을 확인할 수 있습니다.

`get_cfg_var()` 함수는 컴파일하여 PHP를 설치했을 때 또는 아파치 등의 웹 서버에서 정보를 읽어 참조했을 때는 값을 반환하지 않습니다.

예제 파일 | `get_cfg_var.php`

```

1  <?php
2      // php.ini 값을 확인합니다.
3      echo "SMTP Server = ". get_cfg_var("SMTP") . "<br>";
4      echo "SMTP Server = ". get_cfg_var("smtp_port") . "<br>";
5
6      ?>

```

화면 출력

```

SMTP Server = localhost
SMTP Server = 25

```

| 내장 함수 |

```
string getenv ( string $varname )
```

내장 함수 `getenv()`는 환경 변수 값을 읽어올 수 있습니다.

예제 파일 | **cfg-07.php**

```
1 <?php
2     $ip = getenv('REMOTE_ADDR');
3     echo "접속 주소 : $ip <br>";
4
5     // 글로벌 환경 변수 값을 통하여도 읽어들 수 있습니다.
6     $ipAddr = $_SERVER['REMOTE_ADDR'];
7     echo "접속 주소 : $ipAddr <br>";
8
9 ?>
```

| 내장 함수 |

bool putenv (string \$setting)

내장 함수 putenv()는 환경 변수 값을 설정합니다.

예제 파일 | **cfg-08.php**

```
1 <?php
2     print "env is: " . $_ENV["USER"] . "<br>";
3
4     putenv("USER=jiny");
5     print "getenv is: " . getenv("USER") . "<br>";
6
7 ?>
```

화면 출력

env is:
getenv is: jiny

19.3 ini

PHP의 환경 설정 파일 ini에 대한 정보를 읽거나 설정할 수 있습니다. 이와 관련된 몇 개

의 내장 함수들을 제공합니다.

| 내장 함수 |

```
string ini_get ( string $varname )
```

내장 함수 `ini_get()`은 구성 옵션의 값을 가져옵니다.

예제 파일 | `ini_get.php`

```
1  <?php
2      echo 'display_errors = ' . ini_get('display_errors') . "<br>";
3      echo 'register_globals = ' . ini_get('register_globals') . "<br>";
4      echo 'post_max_size = ' . ini_get('post_max_size') . "<br>";
5      echo 'post_max_size+1 = ' . (ini_get('post_max_size')+1) . "<br>";
6      echo 'post_max_size in bytes = ' . ini_get('post_max_size');
7
8  ?>
```

화면 출력

```
display_errors =
register_globals =
post_max_size = 8M
post_max_size+1 = 9
post_max_size in bytes = 8M
```

| 내장 함수 |

```
array ini_get_all ([ string $extension [, bool $details = true ] ] )
```

내장 함수 `ini_get_all()`은 모든 구성 옵션을 가져옵니다.

예제 파일 | `ini_get_all.php`

```
1  <?php
2      print_r(ini_get_all("pcre"));
```

```

3     print_r(ini_get_all());
4
5     ?>

```

| 내장 함수 |

```
string ini_set ( string $varname , string $newvalue )
```

내장 함수 `ini_set()`는 구성 옵션의 값을 설정합니다.

예제 파일 | `ini_set.php`

```

1  <?php
2      echo ini_get('display_errors');
3
4      if (!ini_get('display_errors')) {
5          ini_set('display_errors', '1');
6      }
7
8      echo ini_get('display_errors');
9
10  ?>

```

화면 출력

```
1
```

| 내장 함수 |

```
void ini_restore ( string $varname )
```

내장 함수 `ini_restore()`는 구성 옵션의 값을 복원합니다.

예제 파일 | `ini_restore.php`

```

1  <?php
2      $setting = 'y2k_compliance';

```

```

3
4     echo 'Current value for \'' . $setting . '\': ' . ini_get($setting),
      "<br>";
5
6     ini_set($setting, ini_get($setting) ? 0 : 1);
7     echo 'New value for \'' . $setting . '\': ' . ini_get($setting), "<br>";
8
9     ini_restore($setting);
10    echo 'Original value for \'' . $setting . '\': ' . ini_get($setting),
      "<br>";
11  ?>

```

화면 출력

Current value for 'y2k_compliance':
 New value for 'y2k_compliance':
 Original value for 'y2k_compliance':

| 내장 함수 |

string **php_ini_loaded_file** (void)

내장 함수 `php_ini_loaded_file()`은 `php.ini` 파일의 경로를 검색합니다.

예제 파일 | `php_ini_loaded_file.php`

```

1  <?php
2      $inipath = php_ini_loaded_file();
3
4      if ($inipath) {
5          echo 'Loaded php.ini: ' . $inipath;
6      } else {
7          echo 'A php.ini file is not loaded';
8      }
9  ?>

```

화면 출력

Loaded php.ini: C:\php-7.1.4-Win32-VC14-x86\php.ini

| 내장 함수 |

```
string php_ini_scanned_files ( void )
```

내장 함수 `php_ini_scanned_files()`는 ini 디렉터리에서 파싱된 .ini 파일 목록을 반환합니다.

19.4 프로세스

PHP가 동작되는 프로세스에 대한 정보와 처리를 할 수 있습니다. 이와 관련된 몇 가지 내장 함수들을 제공합니다.

| 내장 함수 |

```
string cli_get_process_title ( void )
```

내장 함수 `cli_get_process_title()`은 현재 프로세스 제목을 반환합니다.

예제 파일 | `cli_get_process_title.php`

```
1  <?php
2      echo "Process title: " . cli_get_process_title() . "\n";
3
4  ?>
```

화면 출력

Process title: 명령 프롬프트 - php -S localhost:8000

| 내장 함수 |

```
bool cli_set_process_title ( string $title )
```

내장 함수 `cli_set_process_title()`은 프로세스 제목을 설정합니다.

예제 파일 | `cli_set_process_title.php`

```
1  <?php
2      $title = "jiny PHP Script";
3
4      cli_set_process_title($title);
5      echo "Process title: " . cli_get_process_title() . "\n";
6
7  ?>
```

화면 출력

Process title: jiny PHP Script

| 내장 함수 |

`int getmypid (void)`

내장 함수 `getmypid()`는 현재 PHP의 프로세스 ID를 확인할 수 있습니다.

예제 파일 | `cfg-05.php`

```
1  <?php
2      echo getmypid();
3
4  ?>
```

화면 출력

3868

| 내장 함수 |

`int getmyuid (void)`

내장 함수 `getmyuid()`는 현재 PHP의 UID를 확인할 수 있습니다.

예제 파일 | [cfg-06.php](#)

```
1 <?php
2     echo getmyuid();
3
4 ?>
```

| 내장 함수 |

int **getmyinode** (void)

내장 함수 `getmyinode()`는 현재 스크립트의 inode 값을 구합니다. inode는 유닉스 계통에서 사용하는 자료구조입니다. 각각의 파일은 1개의 inode 값을 갖고 있습니다.

예제 파일 | [getmyinode.php](#)

```
1 <?php
2     echo getmyinode();
3
4 ?>
```

| 내장 함수 |

string **uniqid** ([string \$prefix = "" [, bool \$more_entropy = false]])

내장 함수 `uniqid()`는 고유 ID를 생성합니다. 마이크로 초 단위로 현재 시각에 따라 접두사를 포함한 식별자를 생성합니다.

예제 파일 | [uniqid.php](#)

```
1 <?php
2     // 고유 ID 값을 생성합니다.
3     printf("uniqid(): %s <br>", uniqid());
4
```



```

5 // 접두사를 붙여서 고유 ID 값을 생성합니다.
6 $prefix = "php_";
7 printf("uniqid('$prefix'): %s <br>", uniqid($prefix));
8
9 // more_entropy
10 printf("uniqid('', true): %s <br>", uniqid('', true));
11
12 ?>

```

화면 출력

```

uniqid(): 599934c16f714
uniqid('php_'): php_599934c16f71a
uniqid('', true): 599934c16f71b2.39586057

```

19.5 리소스

PHP 동작 상태에 대한 메모리 등 리소스 등을 확인할 수 있습니다. 이와 관련된 몇 가지 내장 함수를 제공합니다.

| 내장 함수 |

```
array get_resources ([ string $type ] )
```

내장 함수 `get_resources()`는 활성화된 리소스를 반환합니다.

예제 파일 | `get_resources.php`

```

1 <?php
2 $fp = tmpfile();
3 var_dump(get_resources());
4
5 ?>

```

화면 출력

```
array(1) { [2]=> resource(2) of type (stream) }
```

| 내장 함수 |

```
array getrusage ([ int $who = 0 ] )
```

내장 함수 `getrusage()`는 현재 리소스 사용량을 가지고 옵니다.

예제 파일 | `getrusage.php`

```
1 <?php
2     $dat = getrusage();
3     print_r($dat);
4
5 ?>
```

화면 출력

```
Array ( [ru_majflt] => 3686 [ru_maxrss] => 12852 [ru_utime.tv_usec] => 31250
[ru_utime.tv_sec] => 0 [ru_stime.tv_usec] => 78125 [ru_stime.tv_sec] => 0 )
```

| 내장 함수 |

```
int memory_get_usage ([ bool $real_usage = false ] )
```

내장 함수 `memory_get_usage()`는 PHP에 할당된 메모리 양을 반환합니다.

예제 파일 | `memory_get_usage.php`

```
1 <?php
2     // 시작 전 메모리
3     echo memory_get_usage() . "<br>";
4
5     // 메모리 할당
6     $a = str_repeat("Hello", 4242);
7     echo memory_get_usage() . "<br>";
8
9     // 메모리 해제
10    unset($a);
11    echo memory_get_usage() . "<br>";
```

```
12
13  ?>
```

화면 출력

```
344128
368704
344128
```

| 내장 함수 |

```
int memory_get_peak_usage ([ bool $real_usage = false ] )
```

내장 함수 `memory_get_peak_usage()`는 PHP가 할당한 메모리 피크치를 반환합니다.

19.6 include

내장 함수 `get_included_files()`는 `include` 또는 `require` 파일의 이름을 가진 배열을 반환합니다.

| 내장 함수 |

```
array get_included_files ( void )
```

예제 파일 | [get_included_files.php](#)

```
1  <?php
2
3      include 'test1.php';
4      include_once 'test2.php';
5      require 'test3.php';
6      require_once 'test4.php';
7
8      $included_files = get_included_files();
```

```

9
10     foreach ($included_files as $filename) {
11         echo "$filename\n";
12     }
13
14     ?>

```

| 내장 함수 |

```
string set_include_path ( string $new_include_path )
```

내장 함수 `set_include_path()`는 `include_path` 구성 옵션을 설정합니다.

예제 파일 | `set_include_path.php`

```

1  <?php
2      set_include_path(' /usr/lib/pear');
3
4      // 또는 ini_set()를 사용할 수도 있습니다.
5      ini_set('include_path', ' /usr/lib/pear');
6
7      ?>

```

| 내장 함수 |

```
void restore_include_path ( void )
```

내장 함수 `restore_include_path()`는 `include_path` 구성 옵션의 값을 복원합니다.

예제 파일 | `restore_include_path.php`

```

1  <?php
2
3      echo get_include_path();
4      set_include_path('/inc');
5

```

```

6     echo get_include_path();
7     restore_include_path();
8
9     ?>

```

| 내장 함수 |

```
string get_include_path ( void )
```

내장 함수 `get_include_path()`는 현재 `include_path` 구성 옵션을 가져옵니다.

예제 파일 | [get_include_path.php](#)

```

1  <?php
2      echo get_include_path();
3      echo ini_get('include_path');
4
5  ?>

```

19.7 extention

내장함수 `extension_loaded()`를 확장 프로그램이 로드되었는지 확인합니다. 콘솔상에서 `php -m` 옵션을 통하여 설치된 모든 확장 모듈을 확인할 수 있습니다.

| 내장 함수 |

```
bool extension_loaded ( string $name )
```

예제 파일 | [extension_loaded.php](#)

```

1  <?php
2      if (extension_loaded('curl')) {
3          echo "모듈 설치됨";
4      } else {

```

```

5      echo "모듈이 설치되지 않았습니다.";
6    }
7
8    ?>

```

화면 출력

모듈 설치됨

| 내장 함수 |

array **get_loaded_extensions** ([bool \$zend_extensions = false])

내장 함수 `get_loaded_extensions()`를 컴파일된 모든 모듈명을 배열로 반환합니다.

예제 파일 | `get_loaded_extension.php`

```

1  <?php
2      print_r(get_loaded_extensions());
3
4  ?>

```

화면 출력

```

Array ( [0] => Core [1] => bcmath [2] => calendar [3] => ctype [4] => date [5]
=> filter [6] => hash [7] => iconv [8] => json [9] => mcrypt [10] => SPL [11]
=> pcre [12] => readline [13] => Reflection [14] => session [15] => standard
[16] => mysqlnd [17] => tokenizer [18] => zip [19] => zlib [20] => libxml [21]
=> dom [22] => PDO [23] => Phar [24] => SimpleXML [25] => xml [26] => wddx [27]
=> xmlreader [28] => xmlwriter [29] => cli_server [30] => curl )

```

| 내장 함수 |

array **get_extension_funcs** (string \$module_name)

내장 함수 `get_extension_funcs()`는 모듈의 함수 이름을 가진 배열을 반환합니다.

예제 파일 | `get_extension_funcs.php`

```
1  <?php
2      print_r(get_extension_funcs("xml"));
3
4  ?>
```

19.8 그 외

| 내장 함수 |

`string php_sapi_name (void)`

내장 함수 `php_sapi_name()`은 웹 서버와 PHP 사이의 인터페이스 유형을 반환합니다.

예제 파일 | `php_sapi_name.php`

```
1  <?php
2      $sapi_type = php_sapi_name();
3      echo $sapi_type;
4  ?>
```

화면 출력

cli-server

| 내장 함수 |

`array getopt (string $options [, array $longopts [, int &$optind]])`

내장 함수 `getopt()`는 콘솔상에서 명령행 인수 목록 옵션을 가지고 옵니다. 콘솔 모드를 이용한 응용프로그램을 개발할 때 인수 분석용으로 매우 유용합니다.

예제 파일 | **getopt.php**

```
1  <?php
2      $options = getopt("f:hp:");
3      var_dump($options);
4
5  ?>
```

화면 출력

```
C:\php-7.1.4-Win32-VC14-x86>php getopt.php -fvalue -h
array(2) {
    ["f"]=>
    string(5) "value"
    ["h"]=>
    bool(false)
}
```

| 내장 함수 |

```
array get_defined_constants ([ bool $categorize = false ] )
```

내장 함수 `get_defined_constants()`는 정의된 모든 상수를 이름과 값을 가진 연관 배열 형태로 반환합니다.

예제 파일 | **get_defined_constants.php**

```
1  <?php
2      print_r(get_defined_constants());
3  ?>
```

| 내장 함수 |

```
bool get_magic_quotes_gpc ( void )
```

내장 함수 `get_magic_quotes_gpc()`는 자동 따옴표 기능 활성화 상태를 체크할 수 있습니다.

예제 파일 | [cfg-10.php](#)

```
1  <?php
2      if (get_magic_quotes_gpc()) {
3          echo "자동 따옴표 기능 활성화";
4      } else {
5          echo "자동 따옴표 기능 비활성화";
6      }
7
8  ?>
```

화면 출력

자동 따옴표 기능 비활성화

| 내장 함수 |

```
bool phpcredits ([ int $flag = CREDITS_ALL ] )
```

내장 함수 `phpcredits()`는 PHP에 대한 크레딧을 출력합니다.

예제 파일 | [phpcredits.php](#)

```
1  <?php
2      phpcredits(CREDITS_GENERAL);
3
4  ?>
```


20

오토로드

PHP는 소스 파일을 나누어 작성할 수 있습니다. 분리된 파일은 include/require와 같은 전처리 명령을 통하여 소스를 삽입, 결합할 수 있습니다.

소스의 파일을 분리하여 작성하는 방법은 소스 코드들을 간결화하고, 기능별로 코드들을 분리할 수 있기 때문입니다. 비슷하게 분리된 파일들은 유지보수 및 재사용을 하는 데 있어서 매우 유용합니다.

분리된 파일들은 동작하는 데 있어서 상호 의존관계가 발생합니다. 예로, aaa.php 파일을 실행하는 데 필요한 함수가 bbb.php에 있다면 2개 파일은 서로 의존하게 됩니다. 만일 aaa.php 파일이 의존관계에 있는 bbb.php 파일을 include하지 않으면 의존성 실패로 스크립트 파일은 오류를 발생합니다.

파일을 분리하는 것은 의존성만 잘 관리한다면 매우 유용한 소스 관리 방법입니다. 하지만 너무 많은 파일로 소스가 분리되어 있을 경우 모든 의존성을 체크하여 처리하는 것은 힘이 듭니다.

```
<?php
    include ('파일...');
    include ('파일...');
    include ('파일...');
    ...
    ...
    include ('파일...');
    include ('파일...');
```

소스 상단에 많은 include문을 포함할 것입니다. 이러한 복잡한 의존관계를 쉽게 해결하기 위해서 PHP는 오토로드라는 기능을 적용했습니다. 오토로드는 PHP에서 의존성이 있는 소스를 실행할 때 먼저 실행되어 소스 결합을 처리하게 됩니다.

20.1 클래스 의존성

클래스는 인스턴스를 생성하거나 정적으로 클래스를 사용하기 전에 반드시 클래스가 정의되어 있어야 합니다. 또한 클래스를 상속받을 때도 상속되는 부모의 클래스는 반드시 사용 전에 정의되어 있어야 합니다.

이처럼 PHP가 클래스를 사용하기 전에 연관된 클래스의 관계를 클래스 의존성이라고 합니다. 프로그램이 크고 클래스의 관계가 복잡할수록 클래스의 의존성을 확인하는 것은 매우 중요합니다.

Basic.class.php 파일에,

```
<?php
    class basic {
        function show () {
            return "클래스입니다.";
        }
    }
?>
```

클래스를 생성한 후에,

Test.php

```
<?php
// 클래스 파일을 삽입합니다.
include "basic.class.php";

$basic = new basic;

echo $basic->show();
?>
```

파일에서 사전에 클래스 파일을 삽입합니다.

20.2 클래스 의존성 체크 함수

기존에는 클래스를 사용하면 모든 클래스 파일을 미리 사전에 include 또는 require하여 파일을 삽입했습니다. 하지만 작성한 모든 클래스들이 하나에 파일에 전부 사용되지는 않습니다.

PHP는 클래스의 인스턴스를 선언하거나 정적 호출 시 클래스의 의존성을 사전에 검사할 수 있는 함수를 제공합니다. 클래스 의존성을 체크할 수 있는 함수를 통하여 필요한 클래스 파일을 실시간으로 include 또는 require함으로써 메모리를 관리하고, 복잡한 의존성 관계를 쉽게 처리할 수 있습니다.

이렇게 클래스 파일을 자동으로 확인해서 처리하는 기능을 오토로드 기능이라고 합니다.

20.3 클래스 파일 삽입

PHP 5로 버전업이 되면서 php가 클래스 또는 인터페이스를 호출할 때 자동으로 실행되는 함수가 있습니다.

```
<?php
    function __autoload($className){
        include $className.".class.php";
    }
    $basic = new Basic;
?>
```

__autoload() 함수는 위의 예처럼 new basic 형태로 클래스를 선언할 때, __autoload 함수가 실행되고 클래스 이름이 __autoload의 \$className으로 인자를 전달하게 됩니다. __autoload 함수는 전달된 클래스명 인자를 이용하여 클래스 파일을 include하여 정상적으로 클래스가 선언되도록 합니다.

프로젝트가 커질수록 클래스를 관리하고 무결성을 유지하면서 클래스 파일을 include하기란 쉽지 않습니다. 이런 자동 로딩 기능을 이용하면 클래스 기반으로 코딩하는 데 매우 편리할 것입니다.

PHP 5.1.2로 업그레이드되면서 기존 __autoload는 spl_autoload_register() 함수를 이용하여 보다 더 유연하게 오토로드 처리를 할 수 있게 되었습니다.

최근에는 spl_autoload_register() 함수를 더 많이 쓰는 것 같습니다. spl_autoload_register() 함수는 기존 __autoload 함수 사용법이 같습니다.

```
function spl_autoload_register($className){
    include $className.".class.php";
}
```

위와 같은 형태로 동일하게 사용하면 됩니다.

오토로드 기능을 매번 파일 상단에 선언해서 사용하기란 불편합니다. 또한 오토로드 코드들이 각각의 파일마다 중복될 것입니다. 이런 경우 autoload.php 형태로 별도의 오토로드 처리 파일을 만들어서 사용하면 편리합니다.

autoload.php 파일 생성

```
<?
    // 오토로드 처리를 위한 파일

    // 클래스 파일을 구분하기 위한 사용자 정의 확장자를 사용하면,
    // php 실행 파일과 클래스 파일을 좀 더 쉽게 구분할 수 있습니다.
    $classExt = ".class.php";

    // 클래스를 호출하면 클래스명이 인자로 전달됩니다.
    function spl_autoload_register($className){
        $classFilePath = __DIR__ . "/" . $className . $classExt;

        // 클래스 파일이 존재하는지 검사를 합니다.
        // 클래스 파일이 없는 상태에서 require 등을 통하여 삽입을 한다고 하면,
        // 에러를 발생할 수 있습니다.
        if(is_readable($classFilePath)){

            // 클래스 파일을 불러옵니다.
            require $classFilePath;

        }

    }

?>
```

이렇게 미리 만들어 놓은 클래스 오토로드 처리 파일을 프로그램 작성 시 먼저 한 번 불러 오고, 클래스를 선언하여 사용하면 자동으로 php가 선언되지 않은 클래스 파일을 찾아 불러오고 사용할 수 있도록 처리합니다.

```
<?php
    // 오토로드 처리를 위한 파일을 불러옵니다.
    require autoload.php;

    // 오토로드 파일을 통하여 jiny 클래스가 정의된 클래스 파일을 불러와
    // 클래스를 오류 없이 선언하여 사용할 수 있습니다.
    $jiny = new jiny;
?>
```

20.4 PSR-4 Autoloading

PHP Framework Interop Group(PHP-FIG)에서는 PSR-0 autoloading standard를 제안했습니다. 다음과 같은 규칙을 적용하여 오토로드를 처리하도록 권장하고 있습니다.

- 네임스페이스와 클래스명의 구조는 \<vender Name>\(<namespace>\)*<Classname> 형식을 따릅니다.
- 네임스페이스는 서브 네임스페이스를 포함할 수 있습니다.
- 네임스페이스 구분자는 파일을 불러오기 위한 디렉터리 구분자입니다.
- 클래스명에 포함된 _ 글자는 디렉터리 구분자로 사용된다.
- 네임스페이스와 클래스 파일을 불러올 때 .php를 확장자로 불러옵니다.
- 벤더, 네임스페이스, 클래스는 대소문자를 구분한다.

```
<?php
function autoload($className) {

    $className = ltrim($className, '\\');
    $fileName = "";
    $namespace = "";

    if ($lastNsPos = strpos($className, '\\')) {
```



```

        $namespace = substr($className, 0, $lastNsPos);
        $className = substr($className, $lastNsPos + 1);
        $fileName = str_replace('\\', DIRECTORY_SEPARATOR, $namespace)
            . DIRECTORY_SEPARATOR;

        $fileName .= str_replace('_', DIRECTORY_SEPARATOR, $className) .
            '.php';
        require $fileName;
    }

    ?>

```

20.5 컴포저

PHP의 컴포저(composer)는 오토로드와 의존성을 관리할 수 있는 도구입니다. npm이나 apt, pip 같은 것과 유사합니다. 컴포저는 PSR-4 AutoLoader 제안과 PSR를 준수하여 패키지 의존성을 관리합니다.

컴포저를 PHP에서 사용하기 위해서는 PHP 5.3.2 이상의 버전이 필요로 합니다. 또한 컴포저는 멀티 플랫폼을 지원하므로 윈도우, 리눅스, 맥OS 등에서도 사용이 가능합니다.

20.5.1 컴포저 설치

컴포저는 <https://getcomposer.org>에 접속하면 설치 파일을 다운로드할 수 있습니다. 컴포저는 무료로 다운로드해서 설치할 수 있습니다.

20.5.2 구성 설정

컴포저를 통하여 패키지들을 설치하면 자동적으로 의존성과 관련된 설정 파일들이 생성됩니다. 의존성 설정 파일은 json 타입으로 작성되어 있습니다.

composer.json 파일은 컴포저를 이용하여 의존성 및 설치된 패키지를 관리할 수 있는 설정 파일입니다. 컴포저를 통하여 패키지의 의존성이 설치되면 기존 composer.json 파일과 별개로 composer.lock 파일이 추가로 생성됩니다. composer.lock 파일은 컴포저의 목록, 버전 등 세부적인 정보들을 담고 있습니다.

또한 composer.lock 파일은 컴포저가 패키지들의 최신 버전과 별개로 composer.lock에 명시된 버전으로 다운로드 및 사용하게 됩니다. 이는 컴포저가 패키지를 최신 버전으로 업데이트하여 패키지의 버전 차이로 인해 발생할 수 있는 오류를 줄일 수 있습니다.

20.5.3 컴포넌트

기존의 PHP는 다양한 해결 문제점들을 개발자들이 중복으로 개발하느라 많은 시간을 보냈습니다. 그리고 이렇게 개발, 공개된 소스들은 자신의 홈페이지나 블로그에 공개하는 방식으로 다른 개발자들과 공유했습니다.

개발자들의 서로 다른 공개 방식 및 사이트들을 검색하고 찾아서 자신의 프로젝트에 설치하는 것 또한 개발자로서는 힘든 과정 중 하나였습니다.

PHP는 이러한 분산되어 있는 PHP 소스들을 컴포저 의존성 설치 툴과 더불어 패키지스트라는 통합 관리 사이트를 통하여 체계적으로 공유된 소스들을 관리하고 설치할 수 있는 서비스가 출현했습니다.

PHP 개발자들은 공동의 인터페이스와 코드 스타일로 소스를 코딩하고, 패키지스트 사이트를 통하여 배포, 컴포저를 이용한 설치 과정을 한 번에 할 수 있도록 체계화되고 있습니다. 이러한 공통된 규약과 작업 등으로 지금까지 PHP는 다양한 컴포넌트를 찾아보고, 빠른 시간 안에 문제점을 해결할 수 있게 되었습니다.

컴포넌트들은 작은 단위로 정확한 문제점 해결과 작은 코드로 되어 있기 때문에 빠르고 안정적으로 테스트하면서 적용할 수 있습니다. 또한 프레임워크를 적용하여 사용하는 데 확장성과 향후 유지보수를 편리하게 도와줍니다.

20.5.4 패키지스트

패키지스트(<https://packagist.org>) 사이트는 PHP 컴포넌트를 등록 관리하는 사이트입니다. 패키지스트에는 다양한 PHP 패키지들이 등록되어 있고 검색하여 필요한 패키지를 컴포저를 통하여 설치, 사용할 수 있습니다.

컴포넌트는 대부분 배포하는 사람의 벤더명과 패키지명으로 구성합니다.

벤더명/패키지명

이러한 구조들은 서로 패키지들이 중첩되지 않고 각각의 관리를 하기 위함입니다. 동일한 패키지스트 사이트는 수많은 PHP 개발자들이 만들어 놓은 패키지 컴포넌트를 검색할 수 있는 것과 더불어 자기 자신도 새로운 패키지 컴포넌트를 생성하여 다른 개발자들과 공유할 수 있습니다.

20.5.5 패키지 다운로드

패키지 및 의존성 관리 도구인 컴포저를 이용하면 패키지스트에 등록된 다양한 PHP 컴포넌트를 다운로드 및 설치할 수 있습니다.

패키지스트 사이트에서 필요한 패키지를 검색해 봅니다. 검색 창에서 키워드를 입력하면 관련된 다양한 패키지들을 찾을 수 있습니다.

패키지를 검색하면 간략한 패키지 설명과 함께 다운로드 수와 별점이 표기됩니다. 다운로드 수와 별점은 해당 패키지 컴포넌트가 얼마나 인기가 있고, 많은 사람들이 사용하는지 가늠해 볼 수 있습니다.

하지만 이러한 평점들은 단순한 수치고, 많은 테스트와 자신에게 맞는 패키지를 설치하여 사용하면 됩니다.

필요한 패키지를 찾았으면 콘솔 창에서 composer 명령을 통하여 패키지를 설치할 수 있습니다.

```
composer require jiny/routes
```

패키지를 다운로드하면 자동으로 최상의 디렉터리에 /vender 폴더가 하나 생성됩니다. /vender 폴더는 다운로드한 컴포넌트 패키지들을 저장, 관리하는 폴더입니다.

/vender 폴더 아래에 각각의 컴포넌트의 벤더명으로 폴더를 생성하고, 패키지명으로 폴더를 생성합니다.

/vender / 패키지벤더명 / 패키지명

또한 composer.json과 composer.lock 파일을 생성하고 관리해 줍니다.

20.5.6 사설 저장소

모든 PHP의 컴포넌트를 위의 패키지리스트와 같은 서비스 사이트를 이용해야만 하는 것은 아닙니다. 자체적인 컴포넌트 저장소를 만들어서 개인적으로 제작한 사설 컴포넌트를 관리하고 컴포저를 통하여 설치할 수 있습니다.

개인적인 내부 서버를 통하여 컴포넌트를 관리하기 위해서는 컴포저에게 내부 서버명과 접속 정보를 함께 제공해야 합니다. 다음과 같은 형태로 접속 정보를 auth.json 파일을 생성하여 관리할 수 있습니다. auth.json 파일은 composer.json과 같은 파일 위치에 있습니다.

```
{
  "패키지명":{
    "서버도메인": {
      "username":"접속아이디",
      "password":"접속패스워드"
    }
  }
}
```

또는 콘솔상에서 composer 명령을 입력할 때 직접 입력하거나 --global 명령을 통하여 글로벌 정보로 저장할 수 있습니다.

composer config 패키지명.도메인 아이디 패스워드

또는

composer config --global 패키지명.도메인 아이디 패스워드

--global로 접속 정보를 저장하면 ~/.composer/auth.json 파일에 저장됩니다.

20.5.7 패키지 사용

컴포저를 통하여 컴포넌트 패키지를 다운로드하면 /vender 폴더에 자동적으로 저장됩니다. 컴포저는 모든 패키지들의 의존성과 호환이 되는 PSR-4 기준의 오토로더를 생성, 제공합니다.

자동적으로 제공되는 오토로더를 프로그램 상단에 require 명령을 통하여 삽입하여 패키지들을 사용하면 됩니다.

```
<php
    require "vender/autoload.php";
```

설치된 패키지를 통하여 클래스의 인스턴스를 생성할 때는 패키지의 벤더명과 네임스페이스를 함께 사용해야 합니다.

만일 네임스페이스명이 길면 use 명령을 통하여 별칭 또는 현재의 네임스페이스 경로를 변경할 수도 있습니다.

```
$obj = new \jiny\routes( );
```

A

addslashes	141	array_key_exists	42
addslashes	140	array_keys	28
apache_child_terminate	329	array_map	60
apache_getenv	324	array_merge	32
apache_get_modules	324	array_merge_recursive	32
apache_get_version	323	array_multisort	53
apache_lookup_uri	326	array_pad	63
apache_note	326	array_pop	37
apache_request_headers	327	array_product	65
apache_reset_timeout	328	array_push	37
apache_response_headers	327	array_rand	67
apache_setenv	325	array_reduce	61
array	23	array_replace	67
array_change_key_case	69	array_replace_recursive	68
array_chunk	35	array_reverse	36
array_column	28	array_search	42
array_combine	33	array_shift	39
array_count_values	43	array_slice	34
array_diff	75	array_splice	40
array_diff_assoc	77	array_sum	65
array_diff_key	81	array_udiff	76
array_diff_uassoc	80	array_udiff_assoc	78
array_diff_ukey	81	array_udiff_uassoc	79
array_fill	25	array_uintersect	73
array_fill_keys	25	array_uintersect_assoc	74
array_filter	59	array_uintersect_uassoc	74
array_flip	29	array_unique	44
array_intersect	70	array_unshift	38
array_intersect_assoc	70	array_values	27
array_intersect_key	71	array_walk	54
array_intersect_uassoc	71	array_walk_recursive	55
array_intersect_ukey	72	arsort	49
		asort	48

B~C

base64_decode	315
base64_encode	314
basename	257
cal_days_in_month	200
cal_from_jd	200
cal_info	201
call_user_func	384
call_user_func_array	383
cal_to_jd	202
chdir	239
checkdate	187
checkdnsrr	320
chgrp	250
chmod	247
chop	89
chown	245
chr	109
chroot	239
chunk_split	114
clearstatcache	262
cli_get_process_title	438
cli_set_process_title	438
closedir	240
compact	30
connection_aborted	422
connection_status	420
convert_cyr_string	125
convert_uudecode	124
convert_uuencode	124
copy	284
count	62
crc32	130
crypt	133
curl_close	345
curl_copy_handle	360
curl_errno	353
curl_error	353
curl_escape	355
curl_exec	344

curl_getinfo	356
curl_init	342
curl_multi_add_handle	347
curl_multi_close	347
curl_multi_exec	347
curl_multi_getcontent	349
curl_multi_info_read	350
curl_multi_init	346
curl_multi_select	347
curl_multi_setopt	350
curl_multi_strerror	351
curl_reset	344
curl_setopt	343
curl_setopt_array	343
curl_share_close	345
curl_share_init	345
curl_share_setopt	345
curl_strerror	353
curl_unescape	355
curl_version	354
current	47

D~E

date	178
date_default_timezone_get	216
date_default_timezone_set	215
date_parse	168
date_parse_from_format	180
date_sun_info	176
date_sunrise	177
date_sunset	177
debug_backtrace	375
debug_print_backtrace	376
die	418
dir	232
dirname	234
disk_free_space	244
disk_total_space	244
dns_get_record	320
each	26

easter_date	208	filter_input	18
easter_days	208	filter_input_array	19
echo	134	filter_list	3
end	45	filter_var	4
error_clear_last	374	filter_var_array	19
error_get_last	373	flock	281
error_log	373	fnmatch	262
error_reporting	371	fopen	263
escapeshellarg	150	forward_static_call	385
escapeshellcmd	150	forward_static_call_array	386
eval	419	fpassthru	267
exec	363	fprintf	137
exit	417	fputcsv	294
explode	113	fread	273
extension_loaded	445	frenchtojd	207
extract	31	fscanf	288
		fseek	274
		fsockopen	331
F		fstat	289
fclose	265	ftell	274
feof	267	ftruncate	290
fflush	281	func_get_arg	382
fgetc	268	func_get_args	382
fgetcsv	295	func_num_args	381
fgets	269	function_exists	380
fgetss	270	fwrite	276
file	286		
fileatime	254	G	
filectime	255	get_cfg_var	433
file_exists	252	get_current_user	426
file_get_contents	286	getcwd	234
filegroup	251	getdate	170
fileinode	262	get_defined_constants	448
filemtime	254	get_defined_functions	380
fileowner	246	getenv	433
fileperms	248	get_extension_funcs	446
file_put_contents	286	gethostbyaddr	318
filesize	271	gethostbyname	317
filetype	253	gethostbyname_l	317
filter_has_var	17	gethostname	318
filter_id	3		

get_html_translation_table	147	ini_set	436
get_included_files	443	is_dir	227
get_include_path	445	is_executable	249
getlastmod	426	is_file	252
get_loaded_extensions	446	is_link	258
get_magic_quotes_gpc	448	is_readable	278
get_meta_tags	148	is_uploaded_file	284
getmxrr	321	is_writable	278
getmygid	427	jddayofweek	202
getmyinode	440	jdmonthname	203
getmypid	439	jdtofrrench	207
getmyuid	439	jdtogregorian	205, 206
getopt	447	jdtojewish	206
get_resources	441	jdtojulian	204
getrusage	442	jdtonix	205
getservbyname	319	jewishtojd	206
gettimeofday	171	json_decode	160
glob	233	json_encode	158
gmdate	168	juliantojd	205
gmmktime	174		
gmstrftime	171		
gregoriantojd	202, 206		
		key	47
		krsort	49
		ksort	48
		lcfirst	119
		lchgrp	260
		lchown	261
		levenshtein	104
		link	260
		linkinfo	260
		list	24
		localeconv	152
		localtime	172
		ltrim	90
		mail	370
		md5	129
		md5_file	130
		memory_get_peak_usage	443
		memory_get_usage	442
		metaphone	106

microtime	173	quoted_printable_decode	125
mkdir	229	quoted_printable_encode	125
mktime	174	quotemeta	149
money_format	122		
natcasesort	52		
natsort	51		
next	45		
nl2br	142		
nl_langinfo	152		
number_format	120		
O~Q		R	
opendir	240	random_bytes	126
ord	110	range	64
parse_ini_file	290	rawurldecode	316
parse_ini_string	291	rawurlencode	315
parse_str	116	readdir	242
parse_url	311	readfile	275
pclose	288	readlink	259
pfsockopen	332	realpath	236
phpcredits	449	realpath_cache_get	236
phpinfo	432	realpath_cache_size	237
php_ini_loaded_file	437	register_shutdown_function	418
php_ini_scanned_files	438	register_tick_function	387
php_sapi_name	447	rename	238
php_undef	431	reset	45
phpversion	429	restore_error_handler	377
popen	288	restore_exception_handler	377
preg_match	300	restore_include_path	444
preg_quote	302	rewinddir	243
preg_replace	301	rmdir	237
preg_split	302	rsort	49
prev	45	rtrim	89
print	135		
printf	136, 138		
proc_close	366		
proc_nice	367		
proc_open	365		
proc_terminate	367		
putenv	434		
		S	
		serialize	164
		set_error_handler	377
		set_exception_handler	377
		set_include_path	444
		setlocale	151
		set_time_limit	422, 424
		sha1	131
		sha1_file	132
		shell_exec	365
		shuffle	66
		similar_text	103

sleep	423	str_word_count	87
sort	48	substr	88
soundex	103	substr_compare	101
sscanf	139	substr_count	86
stat	256	substr_replace	108
strcasecmp	100	symlink	259
strcmp	97	sys_get_temp_dir	280
strcoll	98	system	364
strcspn	95		
strftime	180		
str_getcsv	297	T~Z	
stripcslashes	141	tempnam	280
stripos	93	time	169
stripslashes	141	timezone_version_get	215
strip_tags	143	tmpfile	279
str_ireplace	108	touch	255
stristr	91	trigger_error	374
strlen	84	trim	89
strnatcasecmp	102	uasort	56
strnatcmp	102	ucfirst	119
strncmp	99	ucwords	119
str_pad	111	uksort	57
strpbrk	96	umask	247
strpos	92	uniqid	440
strptime	186	unixtojd	205
strrchr	92	unlink	283
str_repeat	111	unregister_tick_function	388
str_replace	107	unserialize	165
strrev	117	urldecode	313
stripos	94	urlencode	313
str_rot13	134	usleep	423
strrpos	93	usort	58
str_shuffle	126	version_compare	430
str_split	114	vfprintf	138
strspn	94	virtual	328
strstr	91	vprintf	137
strtok	112	vsprintf	139
strtolower	118	wordwrap	115
strtotime	175	zend_thread_id	431
strtoupper	118	zend_version	430
strtr	107		

넘버쓰리 PHP

응용 · 실전 끝장내기

초판 1쇄 발행 | 2018년 1월 31일

지은이 | 이호진

펴낸이 | 김범준

기획/책임편집 | 서현

교정교열 | 홍성신

편집디자인 | 한지혜

표지디자인 | 김민정

발행처 | 비제이퍼블릭

출판신고 | 2009년 05월 01일 제300-2009-38호

주소 | 경기도 고양시 덕양구 통일로 140 삼송테크노밸리 B동 229호

주문/문의 | 02-739-0739 팩스 | 02-6442-0739

홈페이지 | <http://bjpublic.co.kr> 이메일 | bjpublic@bjpublic.co.kr

가격 | 28,000원

ISBN | 979-11-86697-52-8

한국어판 © 2018 비제이퍼블릭

이 책은 저작권법에 따라 보호받는 저작물이므로 무단 전재와 무단 복제를 금지하며,
내용의 전부 또는 일부를 이용하려면 반드시 저작권자와 비제이퍼블릭의 서면 동의를 받아야 합니다.

잘못된 책은 구입하신 서점에서 교환해드립니다.