Instrukcja 3

1. Obsługa plików.

PHP, tak jak i inne języki programowania, posiada pełne wsparcie operacji na plikach zarówno tekstowych jak i binarnych. Dzięki odpowiednim funkcjom możemy otwierać, tworzyć, zapisywać, odczytywać i zamykać wskazany plik. Wskazany plik może być zarówno dostępny jako lokalny (na serwerze) jak i zdalny (na innym serwerze/komputerze; zamiast ścieżki lokalnej podaje się kompletny adres do pliku).
Lista podstawowych funkcji obsługujących pliki:

mixed fopen(string $nazwa_pliku, string $tryb, bool $uzyj_zawartej_sciezki = false, resource contex) – funkcja otwiera plik o nazwie $nazwa_pliku i zwraca do niego tzw. uchwyt (handle) typu resource (zmienna specjalna). Parametr $tryb jest obowiązkowy; określa on w jaki sposób będzie użytkowany otwarty plik. Opcjonalny parametr $uzyj_zawartej_sciezki można zmienić na true – wtedy plik, prócz aktualnego katalogu, będzie szukany także w katalogach znajdujących się w zmiennej środowiskowej INCLUDE_PATH. Ostatni parametr pozwala na dodawanie kontekstu dla otwieranego pliku (dodawanie jego właściwości, której domyślnie plik nie posiada, np. autora pliku).

Lista dostępnych flag (dla zmiennej $tryb):
'r' – otwiera plik tylko do odczytu; ustawia wskaźnik na początku pliku;
'r+' - otwiera plik to odczytu i zapisu; ustawia wskaźnik na początku pliku;
'w' – otwiera plik tylko do zapisu; ustawia wskaźnik na początku pliku i jeżeli plik istnieje to kasuje jego zawartość do zera (czyści jego zawartość); jeżeli nie istnieje to próbuje go utworzyć
'w+' - otwiera plik do zapisu i odczytu; ustawia wskaźnik na początku pliku i jeżeli plik istnieje to kasuje jego zawartość do zera (czyści jego zawartość); jeżeli nie istnieje to próbuje go utworzyć
'a' – otwiera plik tylko do zapisu; umieszcza wskaźnik na końcu pliku; jeżeli plik nie istnieje próbuje go utworzyć
'a+' - otwiera plik do zapisu i odczytu; umieszcza wskaźnik na końcu pliku; jeżeli plik nie istnieje próbuje go utworzyć
'x' – tworzy i otwiera plik tylko do zapisu; umieszcza wskaźnik na początku pliku; jeżeli plik już istnieje zamiast uchwytu do pliku zwracana jest wartość false i generowany błąd – ostrzeżenie
"x+' - tworzy i otwiera plik do zapisu i odczytu; umieszcza wskaźnik na początku pliku; jeżeli plik już istnieje zamiast uchwytu do pliku zwracana jest wartość false i generowany błąd – ostrzeżenie
'c' – otwiera plik tylko do zapisu; ustawia wskaźnik na początku pliku; jeżeli plik już istnieje nie jest obcinany (jego zawartość pozostaje) ani nie zostaje zgłoszony jakikolwiek błąd;
'c+' - otwiera plik do odczytu i zapisu; ustawia wskaźnik na początku pliku; jeżeli plik już istnieje nie jest obcinany (jego zawartość pozostaje) ani nie zostaje zgłoszony jakikolwiek błąd;

Domyślnie plik otwierany jest w trybie tekstowym. Trzeba pamiętać, że każdy z systemów operacyjnych posiada różne znaczniki końca linii (końca pliku). Przykładowo system Unix oraz Unix-like kończą linie gdy napotkają znak '\n', systemy rodziny Windows po sekwencji „\r\n” (dwa znaki), a system Unix MacOS (Apple) po '\r'. Jeżeli plik będzie posiadał złe zakodowanie końca linii może to doprowadzić do niekontrolowanych zachowań. W systemach Windows można użyć dodatkowej flagi 't' (od słowa tłumaczenie), dzięki której znak '\n' będzie zamieniany na znaki „\r\n”. Z kolei przy użyciu flagi 'b' żaden ze znaków nie zostanie podmieniony (przetłumaczony). Zarówno flagę 't' jak i 'b' używa się na końcu (jako ostatni znak) zmiennej $tryb.

Przykłady użycia:

$plik = fopen(„domysly_plik.txt”, „r”);

$plik2 = fopen(„plik.txt”, „r+b”);
$plik3 = fopen(„http://php.net”, „r”);
$plik4 = fopen(„plik.txt”, „ct”);

mixed fgets(resource $plik, int $dlugosc_bufora) – pobiera i zwraca całą linię ze wskazanego pliku $plik (uchwyt do pliku); opcjonalnie można podać parametr $dlugosc_bufora – wtedy odczytana zostanie tylko określona ilość bajtów (np. 4096 – czyli 4 kilobajty co odpowiada najczęściej podstawowej wielkości klastra danych w systemie alokacji plików). Na wyjściu funkcji dostajemy część zawartości pliku (string) lub false, jeżeli napotkany zostanie błąd lub koniec pliku (EOF – End Of File). Funkcja jest binarnie bezpieczna (można czytać nią pliki binarne).

mixed fread(resource $plik, int $dlugosc) – podobnie jako poprzednia funkcja czyta dane z pliku; w jej przypadku parametr $dlugosc jest obowiązkowy – funkcja czyta tylko tyle bajtów ile zostanie jej zadane; zwraca wartość typu string (zawartość czytanej części pliku) lub false w przypadku błędu. Funkcja jest binarnie bezpieczna (można czytać nią pliki binarne)

mixed fwrite(resource $plik, string $zawartosc_do_zapisu, int $dlugosc) – funkcja zapisuje $zawartosc_do_zapisu do pliku wskazanego zmienna $plik (uchwyt). Parametr $dlugosc jest opcjonalny – jeżeli został podany to zapis skończy się gdy zapisana zostanie określona ilość bajtów. Funkcja zwraca ilość zapisanych bajtów (int) lub false w przypadku wystąpienia błędu.

bool unlink(string $nazwa_pliku, resource $context) – funkcja kasuje plik podany w parametrze $nazwa_pliku. Opcjonalny parametr $context dodaje dodatkowy kontekst do wywoływanego pliku.

bool feof(resourse $plik) – sprawdza czy doczytany został koniec pliku $plik (true gdy osiągnięty został koniec pliku; false w przeciwnym razie)

bool fclose(resource $plik) – zamyka aktualnie otwarty plik $plik; plik musi być wcześniej otwarty! (np.poprzez funkcję fopen())

bool flock(resource $plik, int $operation, int &$wouldblock) – zakłada blokadę na $plik; blokada określana jest poprzez $operation jako odpowiednio ustawione bity. System blokady plików zapewnia odpowiednią ochronę odczytu/zapisu pliku w środowisku, w którym wielu klientów/aplikacji może żądać np. prawa zapisu do jednego pliku. Zamknięcie i otworzenie odbywa się w sposób arbitralny – niezależny od platformy (poza małymi wyjątkami z rodziną Windows). Trzeci parametr jest opcjonalny – jego wartość ustawiana jest na 1 gdy funkcja będzie blokować dostęp do pliku (nie działa w systemie Windows); funkcja zwraca true w przypadku sukcesu lub false w przypadku porażki.
Możliwości ustawienia zmiennej $operation:
LOCK_SH – wskazany plik jest zamykany jest na wspólne czytanie (shared read)
LOCK_EX – wskazany plik jest zamykany na wyłączność do zapisu (exclusive write)
LOCK_UN – plik jest odblokowywany (gdy wcześniej użyte zostało LOCK_SH, LOCK_EX)
LOCK_NB – dodatkowe ustawienie (do korzystania łączenie z LOCK_SH lub LOCK_EX); jeżeli zostanie ustawione to funkcja flock() NIE ZABLOKUJE wykonywania skryptu w przypadku gdy zostanie wykryta blokada na dany plik (nie działa na systemie Windows).

Przykład:

```
<?php

$fp = fopen("/tmp/lock.txt", "r+");

if (flock($fp, LOCK_EX)) {
    ftruncate($fp, 0);      // przycięcie pliku
```

```
    fwrite($fp, "Write something here\n");
    fflush($fp);            // wyczyszczenie wyjście przed zdjęciem blokady
    flock($fp, LOCK_UN);    // zdjęcie blokady
} else {
    echo "Nie można zablokować pliku!";
}

fclose($fp);

?>
```

```
<?php
$fp = fopen('/tmp/lock.txt', 'r+');

if(!flock($fp, LOCK_EX | LOCK_NB)) {
    echo 'Nie można utworzyć blokady';
    exit(-1);
}

/* ... */

fclose($fp);
?>
```

mixed fgetss(resource $plik, int $length, string $dozwolone_znaczniki_html) – funkcja pobiera linię wskazanego pliku $plik (lub liczbę bajtów podanych w zmiennej $length), wycina znaczniki HTML oraz kod PHP i zwraca treść wspomnianej linii (fragmentu pliku); jako dodatkowy parametr można podać znaczniki HTML, które mają być pozostawione w treści

bool file_exists(string $nazwa_pliku) – sprawdza czy wskazany plik istnieje (true jeżeli istnieje, w przeciwnym wypadku false).

Podane powyżej funkcje są najczęściej wykorzystywane do podstawowej obsługi plików; PHP posiada o wiele więcej funkcji do obsługi plików – ich opis i zastosowanie można znaleźć pod adresem http://www.php.net/manual/en/ref.filesystem.php .

Zadanie:
Proszę dodać do tworzonej przykładowej aplikacji możliwość zapisu osób w plikach (za pomocą funkcji). Zadanie proszę wykonać zarówno poprzez zapis osób/funkcji w osobnych plikach (np. jedna osoba – jeden plik) jak i zapis do jednego pliku (trzeba odpowiednio sformatować plik by dane się nie pomieszały). Proszę dodać do osób nowe pola – login oraz hasło; Wskazane jest by osoby z pliku mogły się logować na stronę poprzez swój login/hasło lub aby same mogły się usunąć z listy.

2. Obsługa katalogów.

Poza obsługą plików PHP oferuje również podstawową obsługę katalogów. Skrypt może sprawdzić ścieżkę, z której wywołany jest skrypt, może zmienić aktualnie wybrany katalog (trzeba jednak pamiętać, że katalog poza katalogiem wywołania może nie być dostępny dla skryptu; dlatego dobrym nawykiem jest używać katalogów w obrębie katalogu skryptu, np. <katalog>wykonania>/podkatalog) sprawdzić zawartość danego katalogu czy utworzyć nowy katalog.

bool chdir(string $katalog) – zmienia aktualnie wybrany katalog na katalog podany w zmiennej

$katalog. WAŻNE! Polecenie to nie zmieni aktualnie wybranego głównego katalogu (tzw. root directory. Jeżeli ścieżka katalogu głównego wygląda tak „/kat_glowny”, to po wykonaniu chdir(„katalog/drugi”) ścieżka zmieni się na: „/kat_glowny/katalog/drugi”.

bool chroot(string $katalog) – funkcja zmienia katalog główny na podany w $katalog. W przeciwieństwie do poprzedniej funkcji zmienia CAŁĄ ścieżkę; biorąc pod uwagę poprzedni przykład to po wydaniu polecenia chroot(„katalog/drugi”) zmieni się ona na: „/katalog/drugi”.

string getcwd(void) – zwraca ścieżkę do aktualnie wybranego katalogu.

bool mkdir(string $sciezkaDoKatalogu, int $tryb = 0777, bool $rekursywnie = false, resource $context) – funkcja tworzy nowy katalog zawarty w $sciezkaDoKatalogu (należy podać pełną ścieżkę + nazwę nowego katalogu). Dodatkowo można ustawić poprzez zmienną $tryb prawa do nowo utworzonego katalogu (domyślnie ustawiane są pełne prawa dla wszystkich); jeżeli zmienimy $rekursywnie na true można utworzyć kilka katalogów jednocześnie (podane w $sciezkaDoKatalogu, np. „/to/ito/jest/nowyKatalog”). Ostatni parametr umożliwia dodanie dodatkowego kontekstu do tworzonego katalogu.

bool rmdir(string $nazwaKatalogu, resource $context) – usuwa katalog podany w $nazwaKatalogu; katalog usuwany jest z katalogu głównego.

array scandir(string $katalog, int $sortowanie = SCANDIR_SORT_ASCENDING, resource $context) – zwraca w postaci tablicy listę zawartości katalogu (katalogi oraz pliki) podanego w zmiennej $katalog; opcjonalnie można zmienić kolejność pojawiania się elementów na liście poprzez $sortowanie (domyślnie wedle alfabetu).

Pełna lista operacji na katalogach http://www.php.net/manual/en/ref.dir.php .

Zadanie:
Proszę rozwijany przykład zmodyfikować tak, by plik/pliki tworzone przez skrypt znalazły się w osobnym katalogu (np. databases); dotworzyć osobny skrypt umożliwiający zarządzanie tworzeniem/kasowaniem kopii zapasowych plików w różnych katalogach.

3. Data i czas.

Od wersji 5.2.0 PHP posiada obsługę daty i czasu poprzez wygodną predefiniowaną klasę DateTime (pełny opis klasy pod adresem http://www.php.net/manual/en/class.datetime.php). Jedną z najważniejszych funkcji klasy jest oczywiście pobranie aktualnego czasu i daty; można to osiągnąć poprzez następujące polecenia:

```
$datetime = new DateTime();
echo $datetime->format(„Y-m-d”);
echo $datetime->format(„H:i:s”);
```

Pierwsze echo wydrukuje aktualną datę w kolejności rok-miesiąc-dzień; trzecia linia wydrukuje aktualną godzinę w formacie godzina:minuta:sekunda; oczywiście zarówno znaki '-' oraz ':' są opcjonalne – można stosować dowolne separatory pomiędzy kolejnymi składowymi daty i czasu. Przy tak utworzonej klasie DateTime() można zauważyć, że wyświetlana godzina jest błędna dla naszej strefy czasowej (domyślnie wyświetlany jest czas UTC - *Universal Time Clock*). Niedogodność tę można wyeliminować na dwa sposoby. Pierwszym z nich jest zmiana deklaracji samej zmiennej:

```
$datetime = new DateTime('NOW', new DateTimeZone('Europe/Warsaw'));
```

Tym razem podane zostały dwa parametry. Pierwszy z nich ('NOW') określa, że ma zostać pobrany aktualny czas; drugi natomiast tworzy nowy obiekt klasy DateTimeZone odpowiedzialnej za odpowiednie ustawienie strefy czasowej; dla Polski należy wybrać wskazaną strefę czasową ('Europe/Warsaw'); pełna lista stref czasowych znajduje się na stronie dokumentacji: http://www.php.net/manual/en/timezones.php .
Inny sposobem jest wykorzystanie wcześniejszego kodu, jednak przed wyświetleniem godziny zmianę strefy wykonuje się odpowiednią funkcją:

```
$datetime = new DateTime();
echo $datetime->format(„H:i:s”);
$datetime->setTimezone('Europe/Warsaw');
echo $datetime->format(„H:i:s”);
```

Pierwsze echo wyświetli godzinę w formacie UTC; drugie echo wyświetli już odpowiednią godzinę dla Polski.
Klasa nie musi wyświetlać nam tylko aktualnej daty; przy jej pomocy można formatować datę i czas zarówno przeszłą jak i przyszłą. Przykładowo:

```
$datetime = new DateTime(„2010-01-28T15:00:00”);
```

utworzy nam zmienną ze wskazanym czasem. Można również:

```
$datetime = new DateTime(„2010-01-28T15:00:00+02:00”);
```

co będzie równoważne z nadaniem strefy czasowej +02:00 (środkowoeuropejska strefa czasowa).
Konstruktor klasy przyjmuje także znacznik czasu systemu UNIX:

```
$datetime = new DateTime(„@946684800 ”);
```

Przy formatowaniu czasu i daty można używać następujących znaków charakterystycznych:

a) dni

d	dzień miesiąca, reprezentowany dwu cyfrowo wraz z przewodnim zerem (np. 01, 09, 12);
D	tekstowa, trzyliterowa reprezentacja dnia (np. Mon, Tue)
j	dzień miesiąca bez przewodniego zera (np. 1, 4, 10, 20)
l	pełna nazwa dnia tygodnia
N	numeryczna reprezentacja dnia tygodnia wg ISO-8601
S	zwyczajowy angielski dopisek dnia miesiąca (st, nd, rd or th); działa też jako parametr j
w	numeryczna reprezentacja dnia tygodnia
z	numer dnia roku (licząc od zera)

b) tygodnie

W	numer tygodnia w roku wg ISO-8601, tydzień rozpoczyna się od poniedziałku

c) miesiące

F	pełna nazwa miesiąca
m	numeryczna reprezentacja miesiąca, wraz z przewodnim zerem
M	trzyliterowa reprezentacja nazwy miesiąca
n	numeryczna reprezentacja miesiąca, bez przewodniego zera
t	liczba dni w danym miesiącu

d)lata

L	wyświetla 1 gdy rok jest przestępny, zero gdy nie jest
o	rok wg ISO-8601
Y	numeryczna reprezentacja roku 4 cyframi
y	numeryczna reprezentacja roku 2 cyframi

e)godziny

a określenie przed/po południem małymi literami (am/pm)
A określenie przed/po południem dużymi literami (AM/PM)
B internetowy czas Swatch (000-999)
g 12-godzinny format godziny bez przewodnich zer
G 24-godzinny format godziny bez przewodnich zer
h 12-godzinny format godziny z przewodnimi zerami
H 24-godzinny format godziny z przewodnimi zerami
i minuty z przewodnimi zerami
s sekundy z przewodnimi zerami
u milisekundy

f)strefy czasowe

e identyfikator strefy czasowej
I zwraca 1 gdy strefa posiada czas letni i zimowy, w przeciwnym wypadku zwraca 0
O różnica pomiędzy wybraną strefą czasową a Greenwich time (GMT), w godzinach
P jak 'O' , lecz z kropką pomiędzy godziną a minutami
T skrót nazwy strefy czasowej
Z przesunięcie strefy czasowej względem UTC w milisekundach (na zachód wartość negatywna, na wschód pozytywna)

g)opcje pełnej daty/godziny

c data wg ISO 8601
r data sformatowana wedle RFC 2822
U sekundy od początku ery Unix (January 1 1970 00:00:00 GMT)

Istnieje jeszcze jeden sposób wyświetlenia daty i czasu – można użyć poniższych funkcji:

string date(string $format, int $znacznikCzasu = time()) - funkcja zwraca sformatowany, lokalny czas serwera (strefa czasowa określana jest w pliku php.ini pod parametrem date.timezone; domyślnie ustawiona jest UTC). Pod zmienną $format podstawia się format daty i czasu (według podanych wyżej parametrów) w jaki reprezentowana ma być data i czas. Opcjonalnie można zmienić $znacznikCzasu – musi on być jednak podany wedle standardu systemu Unix.

array getdate ([int $timestamp = time()]) - funkcja zwraca aktualny czas i datę w postaci tablicy. Opcjonalnie można zmienić $timestamp – musi on być jednak podany wedle standardu systemu Unix. Opis kluczy tablicy dostępny jest pod adresem http://www.php.net/manual/en/function.getdate.php .

array localtime ([int $timestamp = time() [, bool $is_associative = false]]) - zwraca w postaci tablicy lokalny czas (jeżeli nie podamy żadnego parametru). Opcjonalnie można zmienić $timestamp – musi on być jednak podany wedle standardu systemu Unix; jeżeli parametr $is_associative zostanie zmieniony na true klucze tablicy zostaną zamienione z numerycznych na nazwy (opis http://www.php.net/manual/en/function.localtime.php).

Zadanie.
Należy sprawdzić możliwości formatowania daty ze wszystkimi możliwymi parametrami. Następnie trzeba odnaleźć sposób na porównanie dwóch dat (np. przeszłej i aktualnej) i zwrócenie różnicy pomiędzy nimi. Na koniec proszę dodać do poprzedniego przykładu funkcji, która przy dodawaniu nowej osoby automatycznie pobierze i zapisze datę dodania w odpowiednim polu obiektu typu Osoba; proszę dodać możliwość dodania daty urodzenia danej osoby - wtedy przy wyświetlaniu danych o osobie trzeba dodać dodatkową informację ile użytkownik ma lat (różnica pomiędzy dniem dzisiejszym a datą urodzenia).

4. Przesyłanie plików z przeglądarki.

Niekiedy zachodzi potrzeba wysłania pliku/plików na serwer ze stroną; najbardziej charakterystycznym przykładem jest wysłanie obrazka reprezentującego użytkownika na forum, przesłanie zdjęcia do foto albumu lub pliku mp3 do audio blogu. Oczywiście można przesyłać pliki dowolnego typu o teoretycznie dowolnej wielkości. Należy jednak pamiętać o pewnych ograniczeniach. Po pierwsze pliki nie mogą przekroczyć określonego rozmiaru (opcja upload_max_filesize domyślnie pozwala na przesłanie 2Megabajtów); po drugie czas przesyłania pliku nie może przekroczyć czasu wykonania skryptu (max_execution_time); ostatnim bardzo ważnym czynnikiem przesłania pliku jest rozmiar danych przesyłanych metodą post (post_max_size) – domyślnie maksymalny rozmiar wynosi 8Megabajów. Pewne znacznie mogą mieć również wartości parametrów max_input_time oraz memory_limit. Wszystkie parametry dostępne są w pliku konfiguracyjnym php.ini.
Formularz przesyłania plików powinien wyglądać następująco:

```
<html>
<body>
<form enctype="multipart/form-data" action="index.php"
                    method="post" >
<input type="hidden" name="MAX_FILE_SIZE" value="512000" />
<input type="file" name="uploadFile" /><br />
<input type="submit" value="wyślij" />
</form>
<?php
        var_dump($_FILES);
?>
</body>
</html>
```

Powyższy skrypt wstawi pole umożliwiające podanie pliku, który następnie zostanie przesłany na serwer. Pole typu „file” zawiera dodatkowo przycisk „Przeglądaj...”, którego naciśnięcie otwiera standardowe okno przeglądarki plików (z którego można wybrać żądany plik do przesłania). Proszę zauważyć, że formularz zawiera także jeszcze jedno pole, które jest typu „hidden”; pola tego typu nie są widoczne dla użytkownika. Programista stron może dzięki tym polom przekazywać dodatkowe niezależne od wyboru klienta dane, w zależności od swoich potrzeb. W tym przypadku przekazywana jest zmienna MAX_FILE_SIZE (w tym wypadku bardzo ważna jest jej nazwa – musi się tak nazywać), która zawiera maksymalny rozmiar przesyłanego przez nas pliku określony w bajtach (w tym wypadku 0,5 Megabajta). Dokumentacja PHP usilnie zaleca stosowanie tego pola – jego wartość usprawnia działanie skryptu oraz unika sytuacji, gdy użytkownik będzie czekał na załadowanie pliku tylko po to, by ostatecznie serwer odrzucił przesyłany plik ze względu na zbyt duży rozmiar. Trzeba jednak pamiętać, że wartość ta NIE ZMIENIA konfiguracji serwera PHP – jeżeli ustawimy np. wartość 20000000 (20 Megabajtów) przy domyślnej konfiguracji PHP (2 Megabajty maksymalnego rozmiaru pliku) to plik i tak nie zostanie wrzucony na serwer!
Jeżeli dodamy jakikolwiek plik do pola „file” oraz klikniemy przycisk „wyślij” to skrypt poprzez linię
var_dump($_FILES);
wyświetli nam dane jakie zapisuje PHP o przesłanym pliku. Przykładowe wynik działania skryptu:

```
array (size=1)
  'uploadFile' =>
    array (size=5)
      'name' => string 'DSC05373.jpg' (length=12)
      'type' => string 'image/jpeg' (length=10)
```

```
'tmp_name' => string 'C:\wamp\tmp\php8B1D.tmp' (length=23)
'error' => int 0
'size' => int 64020
```

Tablica $_FILES zawiera jedną pozycję pod kluczem 'uploadFile' (taką wartość zawiera parametr name pola „file”). Oczywiście gdybyśmy mieli kilka pól typu „file” to tablica $_FILES zawierałaby kilka pozycji.
Pod wspomnianym wcześniej kluczem PHP alokuje następną tablicę (jej klucze zostają z góry ustalone – programista nie ma na nie wpływu).
Klucz 'name' zawiera właściwą nazwę przesyłanego pliku (tak jak się nazywał przed przesłaniem).
Pod pozycją 'type' PHP przechowuje rodzaj przesłanego pliku – określany jest na podstawie rozszerzenia pliku. Pod 'tmp_name' zapisywane jest aktualne, tymczasowe położenie pliku na serwerze; jeżeli plik nie zostanie skopiowany z tej lokacji to wraz z zamknięciem połączenia przez użytkownika zostanie usunięty.
Osobnego komentarza wymaga zmienna 'error'. Pod nią zapisywany jest potencjalny błąd przesyłania pliku na serwer. Możliwe wartości:
UPLOAD_ERR_OK
Wartość: 0; Brak błędu, plik został przesłany na serwer

UPLOAD_ERR_INI_SIZE
Wartość: 1; Przesyłany plik przekroczył rozmiar zdefiniowany przez parametr upload_max_filesize w php.ini.

UPLOAD_ERR_FORM_SIZE
Wartość: 2; Przesyłany plik przekroczył rozmiar zadeklarowany w zmiennej MAX_FILE_SIZE poprzez formularz HTML.

UPLOAD_ERR_PARTIAL
Wartość: 3; Przesyłany plik został przesłany tylko w części (np. połączenie zostało na chwilę przerwane co zaowocowało zagubieniem niektórych pakietów danych).

UPLOAD_ERR_NO_FILE
Wartość: 4; Żaden plik nie został przesłany.

UPLOAD_ERR_NO_TMP_DIR
Wartość: 6; Brak folderu tymczasowego.

UPLOAD_ERR_CANT_WRITE
Wartość: 7; Błąd zapisu pliku na dysk.

UPLOAD_ERR_EXTENSION
Wartość: 8; Rozszerzenie PHP zatrzymało procedurę przesyłania pliku. PHP nie może ustalić które rozszerzenie spowodowało przerwanie procedury przesyłania; zalecane jest sprawdzenie listy załadowanych rozszerzeń poprzez phpinfo() w celu ustalenia, które z rozszerzeń mogło spowodować powyższy błąd.

Ostatnie pole zawarte pod kluczem 'size' zawiera rozmiar przesłanego pliku podany w bajtach (w naszym wypadku plik posiada ok. 63 kilobajtów).

Jeżeli plik ma zostać zapisany trwale na serwerze trzeba go przenieść odpowiednią funkcją. Funkcja przenoszenia pliku ma następującą postać:

bool move_uploaded_file(string $nazwa_pliku , string $lokalizacja_docelowa) - funkcja przenosi przesłany plik $nazwa_pliku (za zmienną podaje się wartość obecną w polu 'tmp_name') w miejsce określone zmienną $lokalizacja_docelowa (zmienna może mieć np. postać „./folder/ $_FILES['nazwa']['name']). Jeżeli plik zostanie przeniesiony poprawnie funkcja zwróci wartość true; w przeciwnym wypadku wartość false.

Dobrze jest też przez przeniesieniem sprawdzić czy plik rzeczywiście został załadowany poprzez formularz HTTP metodą POST. Służy do tego następująca funkcja:

bool is_uploaded_file(string $nazwa_pliku) - pod zmienną $nazwa_pliku należy podstawić $_FILES['nazwa']['tmp_name']. Jeżeli funkcja zwróci wartość false znaczyć to może, że dostępny plik został załadowany inna metodą (i jest potencjalnym zagrożeniem, np. furtką do ataku na nasz serwis WWW).

Procedura zapisu pliku może wyglądać następująco:

```
$location = './files/' . $_FILES['uploadFile']['name'];
        if(is_uploaded_file($_FILES['uploadFile']['tmp_name']))
        {
                if(!move_uploaded_file($_FILES['uploadFile']['tmp_name'], $location))
                  echo 'Nie udało się zapisać pliku na serwerze (sprawdź uprawnienia do pliku)';
        }
        else
                echo 'Plik nie został przesłany przez formularz';
```

W HTML5 formularz wysyłania plików można utworzyć w taki sposób:

```
<html>
<body>
<form enctype="multipart/form-data" action="index.php"
                method="post" >
<fieldset>
   <legend>Prześlij pliki</legend>
   <label>Dodaj pliki:
     <input type="file" multiple name="files[]">
   </label><br>
        <input type="submit" value="wyślij" />
 </fieldset>
</form>
</body>
</html>
```

To co da się zauważyć to możliwość wyboru kilku plików jednocześnie (poprzez zaznaczenie kilku plików w oknie wyboru; można także używać klawiszy shit, crtl i ctrl+A). Nie trzeba też podawać maksymalnej wielkości wysyłanego pliku (chociaż nadal jest zalecane jej podanie). HTML5 znacznie zmieni procedurę wysyłania plików na serwer; szczegółowe informacje na ten temat znajdują się pod adresem http://www.w3.org/TR/file-upload/ .

Na potrzeby naszego ćwiczenia należy wykorzystać dowolny plik/pliki graficzne (jpg/gif/png) o maksymalnej wielkości 200-300 kilobajtów.

Zadanie:
1) Sprawdzić jak zachowa się formularz z możliwością wysłania kilku plików jednocześnie; jak będzie wyglądać tablica $_FILES?
2) Do przykładu z klasami Osoba – Funkcja dopisać możliwość posiadania przez daną osobę obrazka graficznego; proszę dorobić (jeżeli jeszcze nie istnieje) formularz dodawania nowych osób i dodać do niego pole z możliwością wysłania obrazka (bądź kilku obrazków). Przy funkcji wywołującej dane na temat wskazanej osoby proszę dodać wyświetlenie obrazka, który ją reprezentuje. Gdy obrazków jest kilka proszę losowo wyświetlać jeden obrazek z dostępnej puli. Gdy dana osoba będzie kasowana z bazy osób proszę również kasować obrazki do niej należące.

Aby wylosować obrazek dobrze będzie posłużyć się poniższą funkcją:

int rand (int $min , int $max) - funkcja losuje i zwraca liczbę całkowitą; liczba losowana jest z przedziału określonego poprzez zmienne $min (dolny przedział) oraz $max (górny przedział).

5. Elementy grafiki komputerowej w PHP.

Język PHP umożliwia tworzenie/obróbkę obrazów graficznych. O ile tworzenie grafiki stanowiłoby prawdziwe wyzwanie (można wydawać tylko polecenia rysowania linii, uzupełniania tła i/lub ustawiać kolory poszczególnych pikseli) to obróbka jest nad wyraz prosta i niemal zawsze da się uzyskać upragniony efekt.
Istnieje 5 rozszerzeń językowych umożliwiających operacje graficzne:
- Cairo – biblioteka do obsługi grafiki wektorowej (port z języka C). Wykorzystywana głównie przez system Mac OS. Umożliwia pomniejszanie/powiększanie wskazanego obrazu, skalowanie, przenoszenie i obracanie bez jakiejkolwiek straty dla źródła. Ponadto wspiera renderowanie (wraz z antyaliasingiem) tekstu z czcionek TrueType. W PHP istnieje jako rozszerzenie z biblioteki PECL; aby móc wykorzystywać tę bibliotekę trzeba najpierw zainstalować odpowiednie rozszerzenie.
- Exif – wtyczka umożliwia działanie na meta danych plików graficznych. Dzięki tym danym można dowiedzieć się między innymi o aparacie graficznym robiącym zdjęcie, naświetleniu, użytych opcjach czy geolokalizacji. Wtyczka jest dostarczana wraz z wersją PHP; trzeba ją jednak przeważnie włączać (domyślnie wyłączona).
- GD – podstawowe rozszerzenie operacji na obrazach PHP. Rozpoznaje większość popularnych plików graficznych (jpg, png, bmp, gif, tiff, swf). Pozwala na ich wycinanie, scalanie, zmianę kolorów, dorysowanie pewnych elementów czy też dodanie tekstu z dowolnej czcionki. Domyślnie dostarczana wraz z PHP (od wersji 4.1.0).
- Gmagick – biblioteka pozwala na pobieranie szczegółowych informacji o załadowanym poprzez nią obrazów; poza meta danymi umożliwia odczytanie np. nasycenia kolorem poszczególnych pikseli. Dodatkowo posiada moduł operacji na obrazie – rysowanie, pomniejszanie, powiększanie, obracanie itd. Wymaga osobnej instalacji z repozytorium PECL.
- ImageMagick – rozbudowana biblioteka (klasa) pozwalająca w łatwy sposób edytować i przetwarzać załadowane obrazy. Wymaga instalacji z repozytorium PECL.

Ponieważ każdy dostępny serwer z PHP na pewno posiada rozszerzenie GD to właśnie ono zostanie szerzej omówione.

Najpierw warto sprawdzić z którą wersją GD mamy do czynienia. Służy do tego funkcja :

array gd_info(void) – funkcja zwraca w postaci tablicy informacje o zainstalowanej wersji rozszerzenia, obsługiwanych typach obrazów oraz czcionek.

Najprostszy przykład wykorzystania funkcji operujących na grafice:

```
<?php
$im = @imagecreate(110, 20)          //1
    or die("Cannot Initialize new GD image stream");    //2
$background_color = imagecolorallocate($im, 0, 0, 0);  //3
$text_color = imagecolorallocate($im, 233, 14, 91);    //4
imagestring($im, 1, 5, 5,  "A Simple Text String", $text_color);    //5
imagepng($im, 'test.png');     //6
imagedestroy($im); //7
?>
```

W powyższym przykładzie pierwsza linia tworzy pusty obrazek o wymiarach 110 x 20 pikseli; jeżeli funkcja nie będzie mogła utworzyć obrazu zwróci błąd – "Cannot Initialize new GD image stream ". Kolejna funkcja ustawia tło obrazka na czarne; jej opis jest następujący:

int imagecolorallocate (resource $image , int $red , int $green , int $blue) - funkcja zwraca id koloru, pod którym będzie dostępny w podanym obrazku $image; kolor ustawiany jest poprzez składowe RGB. Jeżeli funkcja użyta zostaje po raz pierwszy po wywołaniu funkcji imagecreate() to kolor ustawiony przez nią staje się automatycznie kolorem tła tworzonego obrazu.

Oznacza to, że w tej linii ustawiamy kolor tła dla tworzonego obrazu. W zmiennej $background_image zapisujemy jego id – dzięki temu można później kolor ten wykorzystać przy innym elemencie obrazu; w naszym przypadku, ponieważ nigdzie dalej kolor ten nie jest wykorzystywany linia mogłaby wyglądać następująco:

```
imagecolorallocate($im, 0, 0, 0);
```

Czwarta linia wygląda podobnie do poprzedniej; w niej jednak pobierany id tworzonego koloru dla obrazu – będzie nam potrzebne do jednego z elementów (tekstu).

Piąta linia odpowiada za utworzenie elementu obrazu – tekstu. Składnia wywołanej funkcji:

bool imagestring (resource $image , int $font , int $x , int $y , string $string , int $color) - funkcja rysuje tekst w obrazie $image, rozpoczynając przy podanych współrzędnych $x oraz $y obrazu (w pikselach); kolor czcionki określony jest poprzez zmienną $color (id koloru dostępnego w zmiennej $image; tworzony poprzez funkcję imagecolorallocate()). Zmienna $font może przyjąć wartość 1-5 dla wbudowanej czcionki systemowej (im wyższa liczba tym większy rozmiar czcionki) lub id dowolnej czcionki utworzonej poprzez funkcję imageloadfont().

W opisywanym przypadku wykorzystujemy wbudowaną czcionkę o rozmiarze 1, rozpoczynając pisanie od 5 piksela szerokości i 5 piksela wysokości; wybieramy kolor zaalokowany pod zmienną $text_color.

Linią numer 6 tworzymy nasz obraz w formacie png; drugi parametr funkcji wskazuje plik pod jakim zostanie zapisany tworzony obraz (w przykładzie zapisze się on w katalogu wywoływanego skryptu).

Na koniec każdy obraz (odnośnik do niego) powinien zostać zniszczony; służy do tego wywołana funkcja pod linią numer siedem.

W przykładzie została przedstawiona funkcja tworząca nowy, czysty obraz. PHP posiada szereg funkcji do tworzenia nowego bądź otwarcia istniejącego już obrazu:

resource imagecreatefromgd (string $filename)

resource imagecreatefromgd2 (string $filename)

resource imagecreatefromjpeg (string $filename)

resource imagecreatefromgif (string $filename)

resource imagecreatefrompng (string $filename)

resource imagecreatefromwbmp (string $filename)

resource imagecreatefromwebp (string $filename)

resource imagecreatefromxbm (string $filename)

resource imagecreatefromxpm (string $filename)

Powyższe funkcje tworzą obraz z podanego pliku lokalnego/zdalnego; w zależności od formatu pliku należy wybrać odpowiednią funkcję; jeżeli format pliku nie będzie zgodny/plik nie zostanie znaleziony zamiast zmiennej resource otrzymamy zmienną bool z wartością false.

Istnieje także kilka innych funkcji tworzących zmienną obrazu:

resource imagecreatefromgd2part (string $filename , int $srcX , int $srcY , int $width , int $height)

funkcja tworzy nowy obraz z fragmentu już istniejącego obrazu zapisanego w formacie gd2. Zmienne $srcX oraz $srcY oznaczają współrzędne punktu, od którego obraz ma być pobrany; zmienne $width oraz $height wyznaczają odpowiednio szerokość oraz wysokość tworzonego obrazu. Jeżeli plik $filename nie zostanie odnaleziony funkcja zwróci wartość false

resource imagecreatefromstring (string $image) - funkcja tworzy obraz z podanego ciągu znakowego; ciąg znakowy powinien być obrazem zapisanym pod tą postacią. PHP automatycznie rozpoznaje typ przekazywanego w ten sposób obrazu (JPG, PNG, WBMP, GD2, GIF). Rozwiązanie tego typu może przydać się podczas przekazywania obrazów strumieniowo z różnych lokacji (rozproszony typ serwerowy).

resource imagecreatetruecolor (int $width , int $height) - funkcja tworzy nowy, pusty obraz; w przeciwieństwie do imagecreate() w standardzie true color (24-bit).

Mając już otwarty obraz (przechwycony w utworzonej zmiennej) można go wyświetlić bezpośrednio w przeglądarce bądź zapisać do pliku. Służą do tego odpowiednie funkcje:

bool imagejpeg (resource $image , string $filename , int $quality) - funkcja wyświetla/zapisuje obraz $image w formacie jpg/jpeg. Parametr $filename jest opcjonalny – jeżeli nie zostanie ustawiony bądź zostanie mu ustawiona wartość NULL funkcja wyświetli obraz; jeżeli zostanie ustawiona nazwa pliku wtedy funkcja zapisze obraz właśnie do niego. Ostatni parametr, również opcjonalny, określa procentowo jakość zapisywanego/wyświetlanego obrazu (wartość pomiędzy 0-100); domyślnie jakość określona jest na 75%. Jeżeli obraz nie może zostać stworzony/zapisany funkcja zwraca wartość false; przy sukcesie wartość true.

bool imagegif (resource $image , string $filename) - wywołanie tej funkcji powoduje wyświetlenie/zapis obrazu $image do formatu gif; funkcja działa identycznie do poprzedniej

bool imagepng (resource $image, string $filename, int $quality, int $filters) - funkcja wyświetla/zapisuje obraz w formacie png. Parametr opcjonalny $quality działa inaczej niż w funkcji imagejpeg() - przyjmuje wartość od 0 (brak kompresji) do 9 (pełna kompresja). Drugi opcjonalny parametr $filters umożliwia ustawienie odpowiednich bitów filtrujących, dzięki którym znacząco da się zmniejszyć rozmiar pliku. Odpowiednie bity reprezentowane są poprzez stałe PNG_FILTER_XXX (możliwe kombinacje PNG_FILTER_NONE, PNG_FILTER_SUB, PNG_FILTER_UP, PNG_FILTER_AVG, PNG_FILTER_PAETH). Można też ustawić bit PNG_NO_FILTER (wyłączenie wszystkich filtrów) bądź PNG_ALL_FILTERS (ustawienie wszystkich filtrów). Dokładny opis filtrowania obrazów PNG dostępne jest pod adresem http://www.w3.org/TR/PNG-Filters.html .

bool imagewbmp (resource $image, string $filename, int $foreground) - funkcja wyświetla/zapisuje obraz w formacie wbmp. Opcjonalny parametr $foregraund określa kolor (jego id utworzone poprzez funkcje imagecolorallocate() tła obrazu; domyślnie tło jest koloru czarnego.

bool imagexbm (resource $image , string $filename, int $foreground) - funkcja wyświetla/zapisuje obraz w formacie xbm. Działa identycznie do poprzedniej funkcji.

bool imagewebp (resource $image , string $filename) - funkcja wyświetla/zapisuje obraz w formacie webP

Przed użyciem funkcji imagepng() można użyć wywołać jeszcze dwie funkcje:

bool imagesavealpha (resource $image , bool $saveflag) - funkcja ustawia flagę $saveflag w obrazie $image odpowiedzialną za zapis kanału alfa (przezroczystość). Domyślne NIE JEST zapisywany kanał alfa ($saveflag = false). Wymaga wersji GD 2.0.1

bool imagealphablending (resource $image , bool $blendmode) - funkcja ustawia flagę $blendmode w obrazie $image, która to odpowiada za włączenie/wyłączenie łączenia kolorów poszczególnych pikseli w trybie przezroczystości. Prześledźmy działanie funkcji na przykładzie. Mamy obraz z czerwonym prostokątem. Jeżeli flaga zostanie ustawiona (true) to gdy nakładamy na tło inny kwadrat (półprzezroczysty) to piksele należące zarówno do pierwszego jak i rysowanego kwadratu są ze sobą porównywane, a kolor nadany każdemu z takich pikseli jest połączeniem koloru z kwadratu „litego” z procentową różnicą koloru „półprzezroczystego”. Przy wartości false flagi kolor rysowanego piksela (drugiego kwadratu) jest kopiowany dosłownie wedle informacji kanału alfa, by ostatecznie zastąpić docelowy kolor piksela. Domyślnie łączenie NIE JEST włączone. Wymagana wersja GD 2.0.1

Poniżej zostaną przedstawione niektóre przydatne funkcje umożliwiające dokonanie operacji na załadowanym obrazie:

int imagecolorallocatealpha (resource $image , int $red , int $green , int $blue , int $alpha) - funkcja działa niemal identycznie do funkcji imagecolorallocate() z tą różnicą, że dzięki niej można określić kanał alfa (współczynnik przezroczystości) danego koloru; działa na obrazach PNG, GIF lub utworzonych funkcją imagecreatetruecolor().

int imagecolorat (resource $image , int $x , int $y) - funkcja zwraca id koloru obrazu $image wskazanego piksela (poprzez koordynaty $x i $y).

array imagecolorsforindex (resource $image , int $index) - zwraca w postaci tablicy (z indeksami „red”, „green”, „blue”) kolor RGB mieszczący się pod wskazanym indeksem koloru w obrazie (indeks można pobrać np. funkcją imagecolorat())

bool imagesetpixel (resource $image , int $x , int $y , int $color) - funkcja ustawia kolor wskazanego piksela obrazu; kolor ustawiany na podstawie id obecnego w obrazie koloru

bool imagefill (resource $image , int $x , int $y , int $color) - funkcja wypełnia (ustawia każdy piksel) wskazanego obrazu wskazanym kolorem (jego id); wypełnianie rozpoczyna się od wskazanego punktu XY

bool imagecopy (resource $dst_im , resource $src_im , int $dst_x , int $dst_y , int $src_x , int $src_y , int $src_w , int $src_h) - funkcja kopiuje fragment wskazanego obrazu $src_img do docelowego obrazu $dst_im. Kopiowanie następuje od wskazanego punktu $src_x/$src_y i do wskazanej szerokości ($src_w) i wysokości ($src_h).

bool imagecopymerge (resource $dst_im , resource $src_im , int $dst_x , int $dst_y , int $src_x , int $src_y , int $src_w , int $src_h , int $pct) - funkcja kopiuje fragment wskazanego obrazu $src_im po czym scala ten fragment do docelowego obrazu $dst_im. W przeciwieństwie do poprzedniej funkcji prócz określenia punktów kopiowania ($src_x, $src_y, $src_w, $src_h) można określić także punkt wklejenia w docelowym obrazie ($dst_x, $dst_y). Zmienna $ptc określa procentowo przezroczystość kopiowanego fragmentu obrazu – dla 0 kopiowany jest i nakładany na docelowy obraz bez żadnych zmian, natomiast przy wartości 100 kopiowany fragment NIE POJAWI się na docelowym obrazie – zostaną skopiowane jedynie jego kolory.

int imageloadfont (string $file) - ładuje bitmapę czcionki i w przypadku sukcesu zwraca id załadowanej czcionki (numer id zawsze większy od 5 w celu zapobieżenia konfliktu z wbudowanymi rodzajami).

int imagefontwidth (int $font)
int imagefontheight (int $font) - funkcje zwracają w pikselach odpowiednio – szerokość oraz wysokość znaku we wskazanej czcionce ($font przyjmuje id pożądanej czcionki).

int imagesx (resource $image)
int imagesy (resource $image) - funkcje zwracają w pikselach odpowiednio – szerokość oraz wysokość wskazanego obrazu

bool imagestringup (resource $image , int $font , int $x , int $y , string $string , int $color) - funkcja wypisuje tekst na obrazku $image pionowo od góry do dołu; zmienne $x oraz $y określają start tekstu od LEWEGO DOLNEGO rogu obrazu

resource imagecrop (resource $image , array $rect) - funkcja wycina z obrazu $image wskazany fragment i zwraca go; fragment wskazywany jest poprzez tablicę punktów koordynujących, która musi zawierać następujące klucze: „x”, „y”, „with”, „height”.

array getimagesize (string $filename, array &$imageinfo) - funkcja zwraca wielkość wskazanego pliku obrazu (szerokość oraz wysokość) jako zerowy i pierwszy element zwracanej tablicy. Opcjonalny parametr $imageinfo zwraca w postaci tablicy dodatkowe informacje o wskazanym obrazie, które są dodawane do obrazu.

resource imagerotate (resource $image , float $angle , int $bgd_color, int $ignore_transparent = 0) funkcja obraca wskazany obraz o zadeklarowany kąt $angle (skala stopniowa, ruch przeciwny do wskazówek zegara). $bgd_color określa jako kolor ma mieć strefa, która zostanie odsłonięta przez obrót (narożniki obrazu). Opcjonalny parametr, gdy różny od zera, pomija przezroczystość obrazu (domyślnie zatrzymuje).

bool imageellipse (resource $image , int $cx , int $cy , int $width , int $height , int $color) - funkcja rysuje elipsę, której środek wyznaczony jest przez punkty $cx, $cy, a $width i $height oznaczają długości średnic w poziomie i pionie; $color przyjmuje numer id koloru, którym ma nastąpić rysowanie.

bool imagerectangle (resource $image , int $x1 , int $y1 , int $x2 , int $y2 , int $color) - funkcja rysuje prostokąt; $x1 oraz $y1 wyznaczają punkt prostokąta mieszczący się góry z lewej strony; $x2 i $y2 wyznaczają punkt prostokąta mieszczący się w prawym dolnym rogu.

Opis pozostałych funkcji na stronie: http://www.php.net/manual/en/ref.image.php

Zadania

1. Proszę znaleźć przykłady wykorzystania przedstawionych funkcji na stronie podręcznika PHP; czy wszystkie działają? Jeżeli nie – jak można obejść niedogodność?
2. W funkcjach zostały przedstawione dwie do rysowania – jedna z nich do elips, druga do prostokątów. Obie figury rysowane są bez środka (tylko ich obrys). Proszę znaleźć funkcje rysujące te figury wraz z wypełnieniem. Jakie jeszcze figury można narysować za pomocą GD?
3. Proszę stworzyć klasę do obsługi przesyłanej grafiki dla potrzeb rozwijanego skryptu. Klasa powinna rozpoznawać typ przesyłanego pliku; jeżeli plik będzie nieprawidłowego typu ma zostać zwrócony błąd. Ponadto proszę stworzyć metodę dodającą do przesłanej grafiki znaku wodnego; użytkownik powinien mieć możliwość opracować własny, chociaż może dostać propozycję takiego znaku (np. imię z klasy Osoba). Wedle gustu użytkownika znak powinien mieć płynną kalibrację przezroczystości – od jej braku do pełnej (brak widoczności znaku).
4. Proszę dodać użytkownikowi możliwość przesyłania 10 osobistych obrazów-zdjęć. Do klasy obrabiającej zdjęcia należy dodać metodę działań na zdjęciu – dodawania np. elipsy o wskazanym kolorze i we wskazanym miejscu; możliwość wycinania kawałka obrazu lub połączenia jednego z drugim. Ponadto można dodać własne, dodatkowe funkcje działań na obrazie (w tym proszę zaprojektować przynajmniej dwie statyczne metody obróbki zdjęć).

Zadanie dodatkowe:

Proszę przetestować tworzenie obrazów przy pomocy strumienia znakowego. W tym celu należy odczytać plik graficzny poleceniem fopen. Następnie zawartość pliku powinno się zakodować do postaci base64 (kod używany np. do kodowania przesyłek e-mail). Służy do tego funkcja:

string base64_encode (string $data) - funkcja zwraca kod base64 z podanego ciągu znakowego

Tak zakodowane dane należy przesłać do drugiego skryptu PHP, który będzie posiadał funkcję dekodującą:

string base64_decode (string $data, bool $strict = false) - funkcja dekoduje podane dane na ciąg znakowy; jeżeli ustawimy opcjonalny parametr $strict na true to funkcja zwróci błąd gdy zadany ciąg będzie zawierał znaki spoza alfabetu kodu base64.

Na koniec należy stworzyć obraz z tego ciągu znakowego poprzez odpowiednią funkcję. Proszę zapisać tak utworzony obraz na dysku w odpowiednim pliku graficznym.