Mistrz Kształcenia Zawodowego

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Człowiek - najlepsza inwestycja

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

LIDER PROJEKTU

PARTNER PROJEKTU

Towarzystwo Oświatowe
Ziemi Chrzanowskiej
w Chrzanowie

eurokreator S.C.

Mistrz Kształcenia Zawodowego

realizatorem projektu jest

Towarzystwo Oświatowe Ziemi Chrzanowskiej w Chrzanowie

w partnerstwie z firmą Eurokreator s.c.

Spis treści

Spis treści

ROZDZIAŁ I

SIECI

Słowniczek

AP – punkt dostępu do sieci.

Brama sieciowa (gateway) – urządzenie dzięki któremu komputery w sieci lokalnej komunikują się z urządzeniami w innych sieciach.

DHCP (Dynamic Host Configuration Protocol) – protokół komunikacyjny umożliwiający uzyskanie od serwera danych takich jak adresu IP, adresu IP bramy sieciowej, adresu serwera DNS, maski podsieci.

DMZ - strefa zdemilitaryzowana jest to wydzielony na zaporze sieciowej obszar sieci komputerowej nie znajdujący się zarówno w sieci wewnętrznej jak i zewnętrznej. Jest to obszar wystawiony „bezpośrednio" na Internet poza ochroną firewalla.

DNS (Domain Name System) – system serwerów, protokół i usługa zamieniający adres IP składający się z 4 grup liczb (IPv4) na bardziej zrozumiały i łatwy do zapamiętania np. www.onet.pl zamiast 213.180.130.200.

Dnsmasq – serwer DHCP i DNS.

Firmaware - oprogramowanie wbudowane w urządzenie, zapewniające procedury obsługi tego urządzenia.

FTP (ang. File Transfer Protocol) - to protokół służący do transmisji plików. Przeważnie usługę ftp stosuje do przesyłania danych z odległej maszyny do lokalnej lub na odwrót.

Host – komputer podłączony do sieci komputerowej.

Hotspot – punkt dostępu, umożliwiający połączenie z Internetem najczęściej poprzez sieć bezprzewodową.

IP (Internet Protocol) – protokół komunikacyjny.

JFFS2 – system plików dziennika używany w urządzenia z pamięcią Flash.

Kabel koncentryczny (coaxial cable) – Przewód telekomunikacyjny, wykorzystywany do transmisji sygnałów zmiennych małej mocy.

LAN (Local Area Network) – lokalna sieć komputerowa.

MAC (Media Access Control) – sprzętowy adres karty sieciowej .

Maska podsieci – liczba służąca do wyodrębnienia w adresie IP części sieciowej od części hosta.

Multicast - sposób dystrybucji informacji, dla którego liczba odbiorców może być dowolna.

NAT – technika przesyłania ruchu sieciowego poprzez router, która wiąże się ze zmianą źródłowych lub docelowych adresów IP. Większość systemów korzystających z NAT ma na celu umożliwienie dostępu wielu hostom w sieci komputerowej do Internetu przy wykorzystaniu publicznego adresu IP.

NVRAM - pamięć komputerowa, która nie traci zawartych informacji po zaniku zasilania.

Port trigering - automatyczne przekierowanie danych portów jeśli router wykryje na nich rozpoczęcie ruchu sieciowego.

PPTP (Point to Point Tunneling Protocol) - protokół komunikacyjny umożliwiający tworzenie wirtualnych sieci prywatnych wykorzystujących technologię tunelowania.

Proxy - oprogramowanie lub serwer z odpowiednim oprogramowaniem pośredniczący między siecią,

QoS – (Quality of Service)- usługa np. kształtowania i ograniczenia przepustowości łącza, aplikacji czy innych usług sieciowych.

RJ 11 - sześciokrotny (szeroki na sześć styków) wtyk telefoniczny z dwoma stykami przeznaczony dla zarejestrowanego gniazda 11, używany zazwyczaj do zakończenia przewodów łączących

RJ 45 - typ złącza stosowany do podłączania modemów, sieci, etc. Wykorzystywane są piny 4. i 5. – podłączenie linii oraz 7. i 8. – rezystor umieszczony w gnieździe, sterujący mocą modemu.
Repeater - urządzenie wzmacniające sygnał.
Router - urządzenie sieciowe łączące różne sieci komputerowe.

SLIP (Serial Line Interface Protocol) - to protokół transmisji przez łącze szeregowe. Uzupełnia on działanie protokołów TCP/IP tak, by możliwe było przesyłanie danych przez łącza szeregowe.

SNMP - protokoły sieciowe wykorzystywane do zarządzania urządzeniami sieciowymi.

SSID – nazwa sieci bezprzewodowej.

TCP/IP (Transmission Control Protocol / Internet Protocol) - to zespół protokołów sieciowych używany w sieci Internet.

Telnet – standard protokołu komunikacyjnego używanego w sieciach do obsługi odległych terminali na zasadzie klient-serwer.

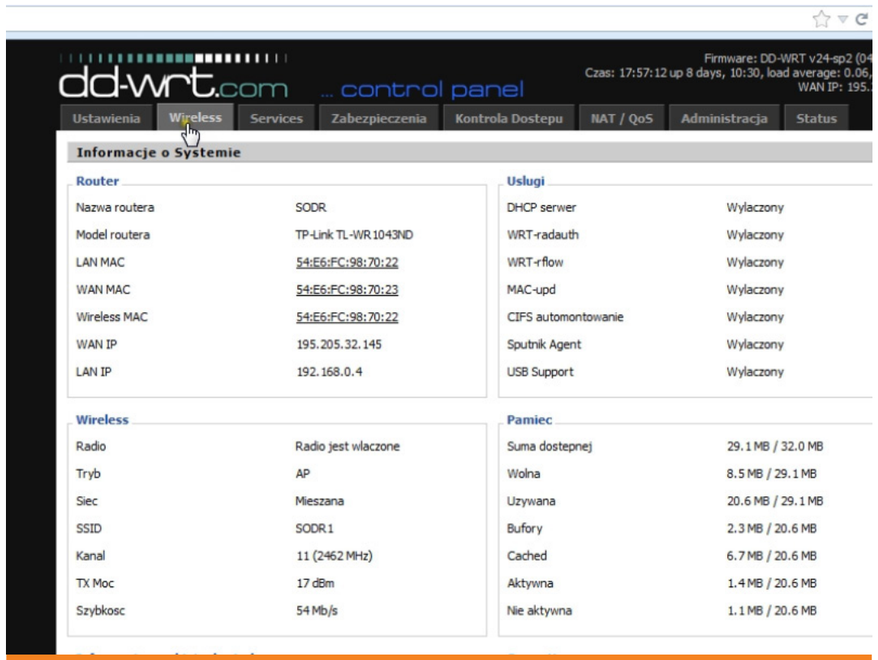
Topologia sieci komputerowej – model układu połączeń różnych elementów (linki, węzły itd.) sieci komputerowej.

VPN - tunel, przez który płynie ruch w ramach sieci prywatnej pomiędzy klientami końcowymi za pośrednictwem publicznej sieci. Połączenie takie można skonfigurować w celu podniesienia bezpieczeństwa i szyfrowania danych.

WAN (Wide Area Network) – rozległa sieć komputerowa.

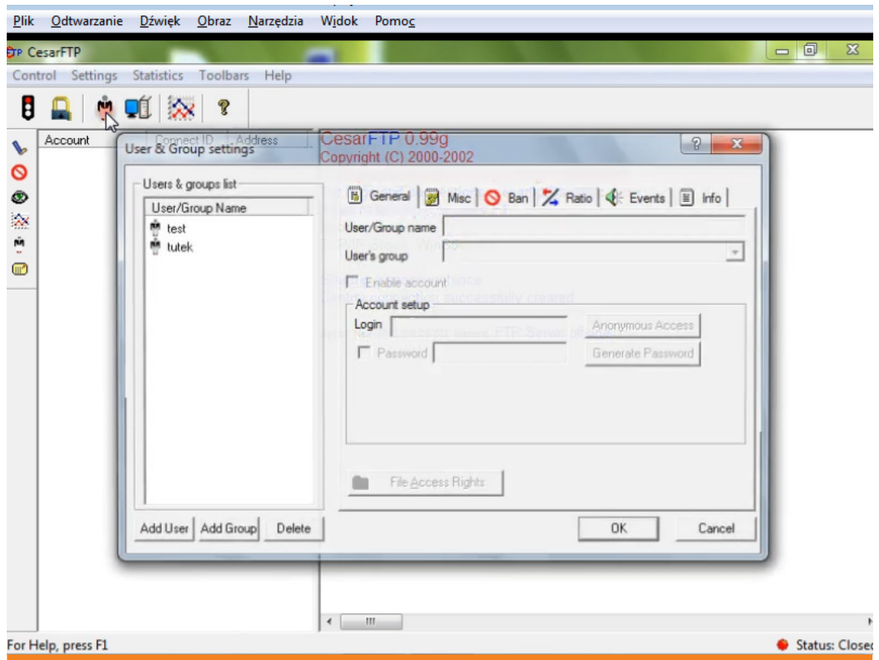
Wireless – sieć bezprzewodowa.

DD-WTR - podstawy

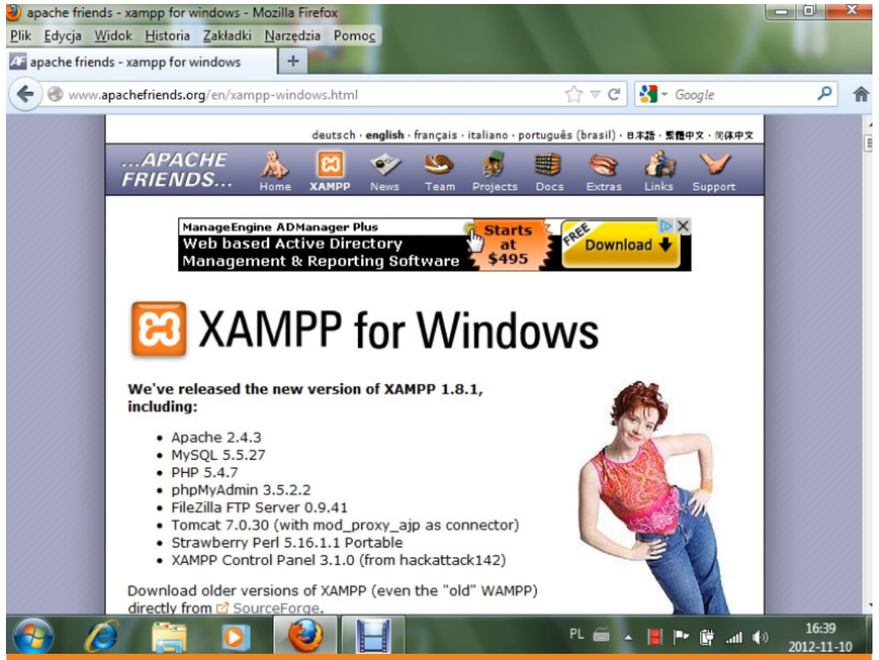


DD-WTR - tworzenie sieci prywatnych

TeamViewer



CesarFTP - FTP w kilka minut



Własny serwer WWW - XAMPP

Tworzenie Wirtualnej maszyny - Virtual Box

Wyszukanie i zamiana adresu MAC - CC Get MAC

Wyszukanie i zamiana adresu MAC - CC Get MAC

PROGRAMOWANIE W PHP

1. Środowisko pracy

1.1 Serwer PHP

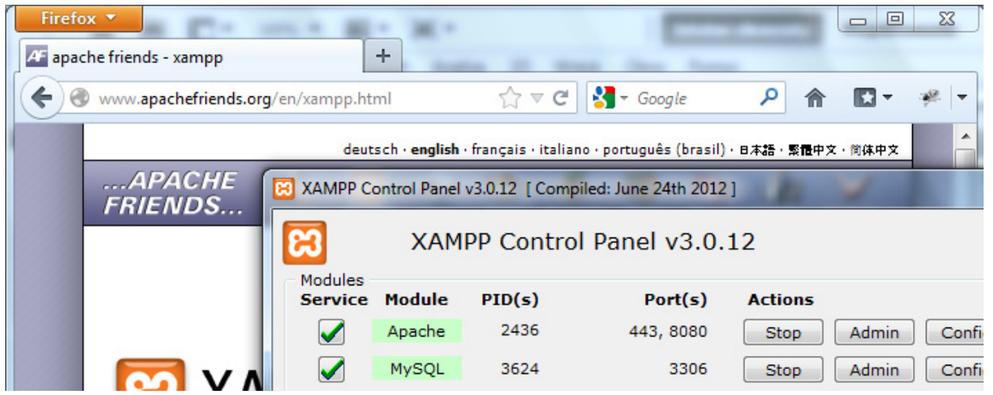
Skrypty PHP (plik.php) przetwarzane są przez serwer. Wynik interpretacji tego kodu przekazywany jest do przeglądarki użytkownika. Aby serwer „rozumiał" pliki PHP niezbędny jest interpreter tego języka programowania.

Najszybszym rozwiązaniem jest instalacja programu, który automatycznie zainstaluje wszystko co niezbędne do działania stron napisanych w PHP.

Jednym z takich programów dostępnych bezpłatnie jest XAMPP.
(www.apachefriends.org/en/xampp.html)

Program zawiera serwer Apache wraz z interpreterem PHP, bazą danych MySQL i innymi przydatnymi dodatkami.

Po instalacji należy uruchomić program i włączyć serwer Apache za pomocą opcji Start.



Obrazek 1- Włączony serwer Apache i MySQL.

Uruchomiony serwer dostępny jest pod adresem localhost.

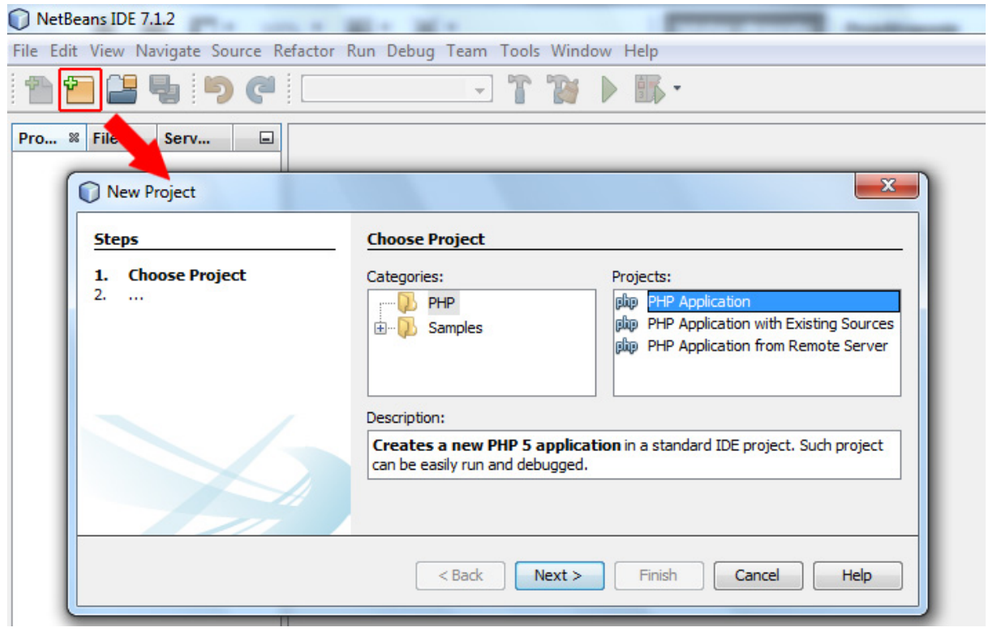
Obrazek 2- Strona główna serwera

1.2 Środowisko programistyczne

Pliki PHP można tworzyć w dowolnym edytorze tekstu. Począwszy od Notatnika po zawansowane programy.
Program NetBeans (www.netbeans.org) jest jednym z najpopularniejszych środowisk programistycznych w którym można pisać w wielu językach programowania, w tym w PHP. Dostępny jest za darmo.

1.3 Pierwszy projekt

Serwer Apache włączony. Środowisko programistyczne NetBeans również.
Należy utworzyć nowy projekt. Menu File ->New project…
Otworzone zostanie okno tworzenia nowego projektu.
Na liście Projects w tym przypadku musi być zaznaczona pozycja
PHP Application

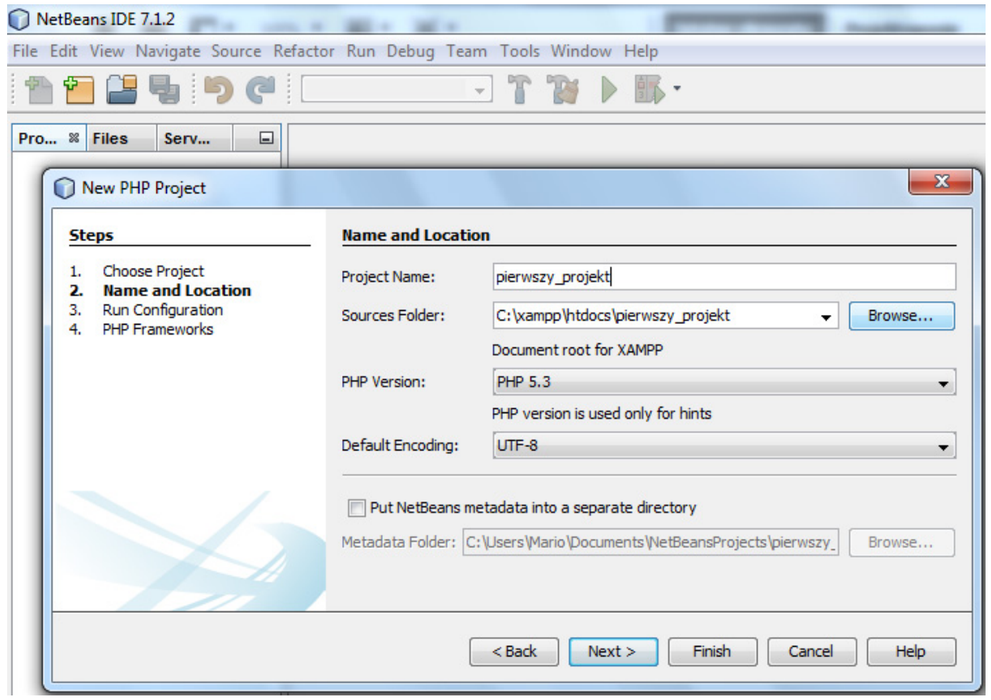


Obrazek 3- Włączony serwer Apache i MySQL.

Po przejściu do drugiego kroku (Next) tworzenia nowego projektu należy podać nazwę projektu (Project Name) oraz ścieżkę do folderu (Sources Folder) w którym będą znajdowały się pliki PHP.
Folder plików PHP musi znajdować się w folderze htdocs serwera XAMPP.
Domyślnie znajduje się on na dysku C:

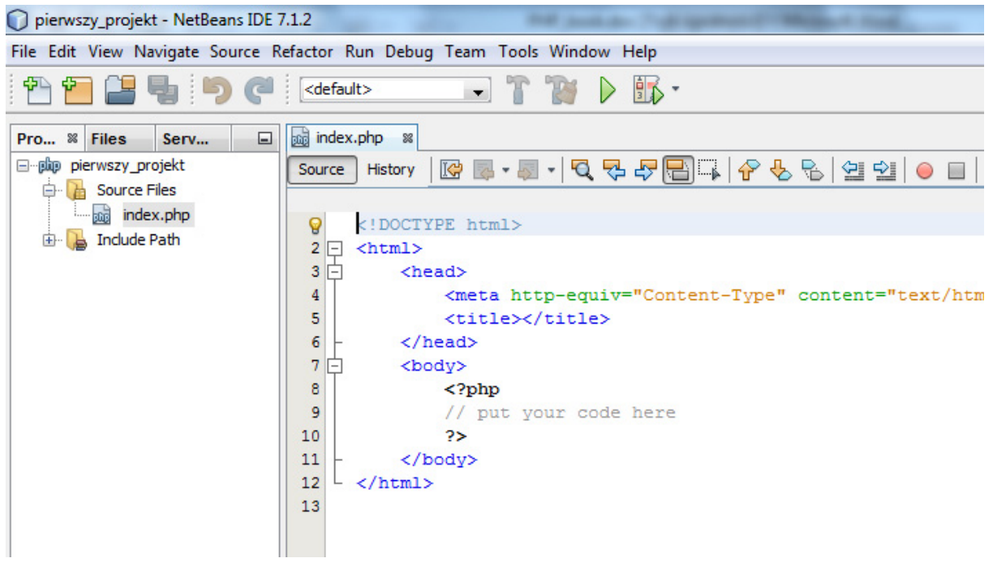
Przykład 1.

C:\xampp\htdocs\pierwszy_projekt



Obrazek 4- Nowy projekt nosi nazwę „pierwszy_projekt”

Kliknięcie przycisku Finish spowoduje utworzenie nowego projektu i wygenerowanie pliku index.php z przykładowym kodem HTML i PHP.



Obrazek 5- Nowy projekt i przykładowa treść pliku index.php

1.4 Plik index.php

Pliki PHP mogą posiadać dowolne nazwy (bez spacji i polskich znaków) jednak plik index.php posiada szczególną rolę. Jest to plik domyślny. Jeśli adres strony nie zawiera nazwy konkretnego pliku PHP do wyświetlenia to zawsze zostanie wyświetlony plik index.php.

Przykład 2.

www.przyklad.pl/artykul1.php
W tym przypadku zostanie wyświetlona treść wygenerowana przez plik artykul1.php

www.przyklad.pl
W tym przypadku zostanie wyświetlona treść wygenerowana przez plik index.php

2. Programowanie

2.1 Gdzie jest PHP?

Plik PHP może zawierać kod HTML lub zwykły tekst, który nie wymaga pracy interpretatora PHP. Aby interpreter wiedział, który fragment dokumentu to kod PHP, należy umieścić go między znacznikami „<?php” i „?>”

Przykład 3.

Zwykly tekst...

<?php

Kod PHP

?>

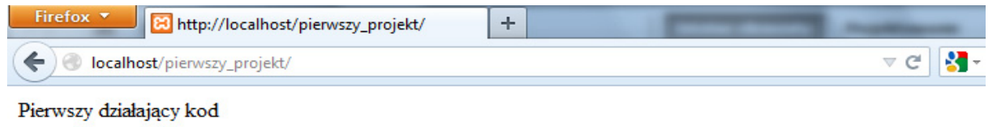
2.2 Pierwszy skrypt

W poprzednim rozdziale program NetBeans wygenerował plik index.php wraz z przykładową treścią. Utworzone zostały znaczniki „<?php” , „?>” między którymi zostanie wpisany kod PHP.

Przykład 4.

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv=”Content-Type” content=”text/html; charset=UTF-8”>
    <title></title>
  </head>
  <body>
    <?php

            echo ‘Pierwszy działający kod’;

    ?>
  </body>
</html>
```

Efekt działania kodu można zobaczyć pod adresem: localhost/pierwszy_projekt/



Obrazek 6- Treść generowana przez plik index.php

Jedynie efekt działania kodu PHP jest widoczny w oknie przeglądarki natomiast sam kod jest niewidoczny dla użytkownika.

W celu wyświetlenia przykładowego tekstu, została użyta instrukcja echo zaś sam tekst został umieszczony między apostrofami. W tym przypadku tekst można umieścić również w cudzysłowu i efekt działania kodu będzie taki sam.

Przykład 5.

```
<?php

        echo ”Pierwszy działający
kod”;

?>
```

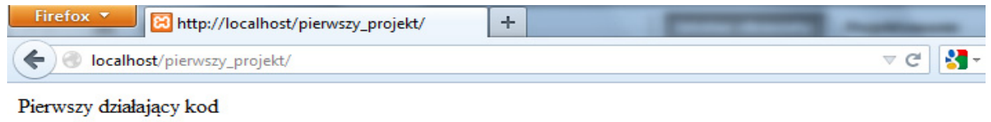
Każda instrukcja PHP musi być zakończona średnikiem „;” w przeciwnym wypadku zostanie wygenerowany błąd.

2.3 Komentarze
Kod PHP może zawierać komentarze, które nie są wyświetlane w przeglądarce ani nie są interpretowane.
Komentarz mieszczący się w jednej linii rozpoczyna znacznik „//”

Przykład 6.

```
<?php

        echo ”Pierwszy działający kod”;  // Zapamiętać

?>
```


W oknie przeglądarki nic się nie zmieni.



Obrazek 7- Skrypt wyświetla taką samą treść

Jeśli komentarz ma mieścić się w więcej niż jednej linii, należy umieścić go między znacznikami „/*” i „*/”.

Przykład 7.

```
<?php

        echo ”Pierwszy działający kod”;  /* Muszę zapamiętać  ten kod */

?>
```

Znaczniki komentarzy mogą służyć też do wyłączenia dowolnego fragmentu kodu PHP. Kod taki zostanie potraktowany jak każdy inny komentarz.

Przykład 8.

```
<?php

/*
        echo ”Pierwszy działający kod”;
*/

?>
```

Kod PHP został zakomentowany dlatego w oknie przeglądarki nic się nie pojawi.

Obrazek 8- Pusta strona. Kod nie jest interpretowany

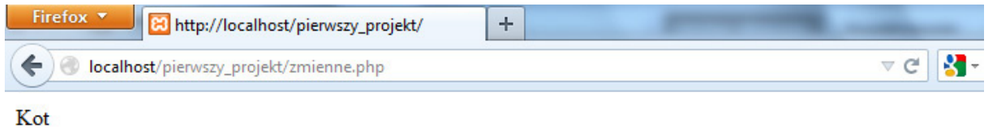
2.4 Zmienne (właściwości)

Zmienna (właściwość) jest identyfikatorem znakowym reprezentującym jakąś wartość. Inaczej mówiąc zmienna przechowuje jakąś wartość.
Zmienną tworzy się znakiem dolara „$" po którym występuje nazwa identyfikatora. Identyfikator może posiadać dowolną długość i może składać się z liczb, cyfr oraz dolnych kresek jednak nazwa nie może rozpoczynać się od cyfry.

Przykład 9.

```
<?php

        $zmienna1 = "Kot";
        $Zmienna1 = "Pies";

        echo $zmienna1;  //Wyświetlony zostanie napis „Kot"

?>
```

Wielkość liter w nazwach zmiennych ma znaczenie dlatego kod wygeneruje napis „Kot".



Obrazek 9

Rodzaj danych przechowywanych przez zmienną określa jej typ.
Występuje kilka typów zmiennych:

- integer – zmienna przechowuje liczby całkowite
- float (double) - liczby rzeczywiste
- string – ciągi znaków
- boolean – wartość true lub false
- array - przechowuje tablice
- object - przechowuje obiekty

2.5 Operatory

Operatory to symbole służące do operacji na zmiennych.
Operatory dzielą się na:

arytmetyczne - służą do operacji na liczbach

- " + " zwraca sumę dwóch liczb lub ciągów
- " - " zwraca różnicę
- " * " zwraca iloczyn
- " / " zwraca iloraz
- " % " zwraca resztę z dzielenia

Przykład 10.

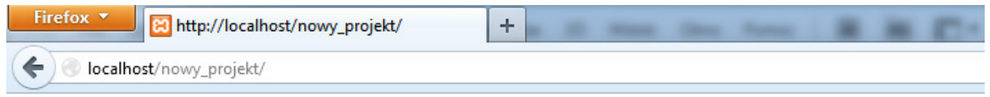
```
<?php

        $a = 3 + 5; //zmiennej a przypisz sumę 3 i 5

?>
```

operatory przypisania - służą do przypisywania zmiennym wartości

- " = " znak równości w języku PHP jest znakiem przypisania
- " $i+=3 " zwiększenie wartości o 3
- " $i-=3 " zmniejszenie wartości o 3
- " $i*=3 " przypisanie wartości 3 razy większej
- " $i/=3 " przypisanie wartości 3 razy mniejszej
- " $i%=3 " przypisanie wartości reszty z dzielenia zmiennej przez 3

```
<?php

        $a = 10;  //zmiennej a przypisz 10
        $b = 8;  //zmiennej b przypisz 8

        $c = $a - $b; //odejmij wartość zmiennej b od wartości zmiennej a i wynik
przypisz                //zmiennej c

        $c+=20; //do wartości zmiennej c dodaj 20

        echo $c; //wyświetl zawartość zmiennej c

?>
```

Poniżej widoczny jest efekt działania kodu.



Obrazek 10

operatory operacji bitowych

- " $a & $b" - AND. Ustawiane są bity, które są ustawione w obu zmiennych.
- " $a | $b" – OR. Ustawiane są bity, które są ustawione w jednej lub drugiej zmiennej.
- " $a ^ $b" – XOR. Ustawiane są bity, które są ustawione w jednej lub drugiej zmiennej, ale nie w obu.
- " ~ $a" – NOT. Inwerter – ustawiane są bity które nie są ustawione w zmiennej $a i odwrotnie.
- " $a << $b" - Przesunięcie w lewo. Przesuń bity z $a $b-razy w lewo (każdy krok oznacza pomnożenie przez 2)
- " $a >> $b" - Przesunięcie w prawo. Przesuń bity z $a $b-razy w prawo (każdy krok oznacza podzielenie przez 2)

operatory porównania - używane w instrukcjach warunkowych

- " == " sprawdza, czy dwie zmienne są równe, co do wartości
- " != " sprawdza, czy zmienne są różne co do wartości
- " === " sprawdza, czy zmienne są identyczne
- " !== " sprawdza, czy zmienne są nieidentyczne
- " > " sprawdza, czy zmienna z lewej strony jest większa od zmiennej z prawej strony
- " < " sprawdza, czy zmienna z prawej strony jest większa od zmiennej z lewej strony
- " >= " sprawdza, czy zmienna z lewej strony jest większa bądź równa od zmiennej z prawej strony
- " <= " sprawdza, czy zmienna z prawej strony jest większa bądź równa od zmiennej z lewej strony

operator kontroli błędów

- " @ " - wyrażenie przed którym znajduje się znak małpy, nie spowoduje wyświetlenia się jakiegokolwiek błędu wygenerowanego przez to wyrażenie.

operator wykonania (wywołania) - służy do uruchamiania zewnętrznych programów lub poleceń powłoki.

- " ‘ ’ " polecenie pomiędzy znakami odwróconego apostrofu zostanie wykonane

operatory inkrementacji i dekrementacji

- " $i++ " postinkrementacja, zwiększa wartość zmiennej o 1
- " ++$i " preinkrementacja, zwiększa wartość zmiennej o 1
- " $i-- " postdekrementacja, zmniejsza wartość zmiennej o 1
- " --$i " predekrementacja, zmniejsza wartość zmiennej o 1

operatory logiczne

- " ! " operator zaprzeczenia (logiczne NOT)
- " && " operator koniunkcji (logiczne AND)
- " || " operator alternatywy (logiczne OR)

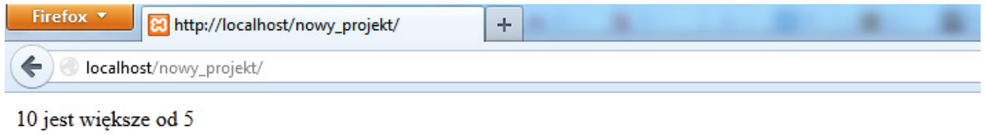
2.6 Instrukcja warunkowa IF

Instrukcja warunkowa umożliwia wykonanie dowolnego kodu w jakimś szczególnym przypadku (spełnienia lub nie, warunku instrukcji).
Instrukcje warunkową rozpoczyna słowo kluczowe if po którym w nawiasie okrągłym () należy podać warunek instrukcji. Kod który ma zostać wykonany w przypadku spełnienia warunku instrukcji zamknięty jest w nawiasie wąsiastym {}

Przykład 11.

```
<?php

        if (10 > 5) {

                echo  "10 jest większe od 5"; /*napis zostanie wyświetlony jeśli
                                                warunek będzie spełniony */

        }

?>
```

Oczywiście 10 jest większe od 5 więc kod między nawiasami {} zostanie wykonany.



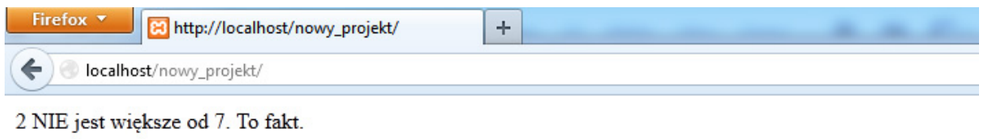
Obrazek 11

W instrukcji warunkowej if można użyć dowolnego operatora porównania.

Instrukcja if może być rozbudowana o sekcję else if oraz else.
Jeśli kod ma wyświetlać informację również w przypadku niespełnienia warunku, wystarczy dodać polecenie else.

Przykład 12.

```
<?php

    if (2 > 7) {

        echo  “2 jest większe od 7. Akurat:)”; /*napis zostanie wyświetlony jeśli warunek będzie spełniony */

    }  else {

        “2 NIE jest większe od 7. To fakt.”; /*napis zostanie wyświetlony jeśli warunek nie będzie spełniony */
}

?>
```

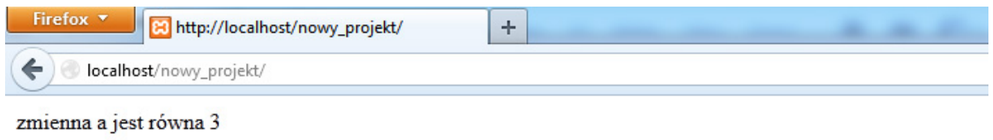


Obrazek 12

Jest jeszcze opcja, że liczby będą sobie równe. Można więc skorzystać z polecenia else if, który zawiera własny warunek.

Przykład 13.

```
<?php

   $a=3;

   if ($a>3) {

      echo “zmienna a jest większa od 3”;

   } else if ($a==3) {

      echo “zmienna a jest równa 3”;

   } else {

      echo “zmienna a jest mniejsza od 3”;
   }

?>
```



Obrazek 13

2.7 Instrukcja warunkowa SWITCH

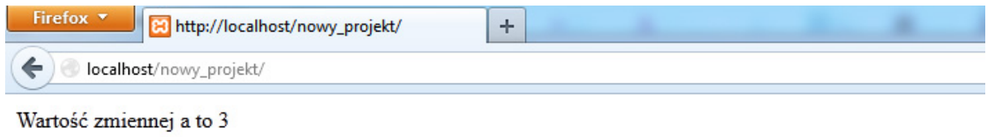
SWITCH jest instrukcją warunkową w której jedną zmienną można porównać z wieloma wartościami.

Przykład 14.

```
<?php

$a = 3;

switch ($a)  {  //nazwa zmiennej, której wartość będzie sprawdzana
case 1: //czy wartość zmiennej a wynosi 1?
```

```
 echo “Wartość zmiennej a to 1”;
 break;

case 2: //czy wartość zmiennej a wynosi 2?
 echo “Wartość zmiennej a to 2”;
 break;

case 3: //czy wartość zmiennej a wynosi 3?
 echo “Wartość zmiennej a to 3”;
 break;

default:
 echo “Wartość zmiennej a nie wynosi 1, 2 ani 3”;
 break;
}
?>
```

Ponieważ zmienna „$a” posiada wartość 3, zostanie wyświetlony odpowiedni komunikat.



Obrazek 14

Kod w sekcji default wyświetlany jest w przypadku gdy w żadna wartość case nie wynosi 3. Inaczej mówiąc sekcja default jest domyślną.

2.8 Pętla FOR

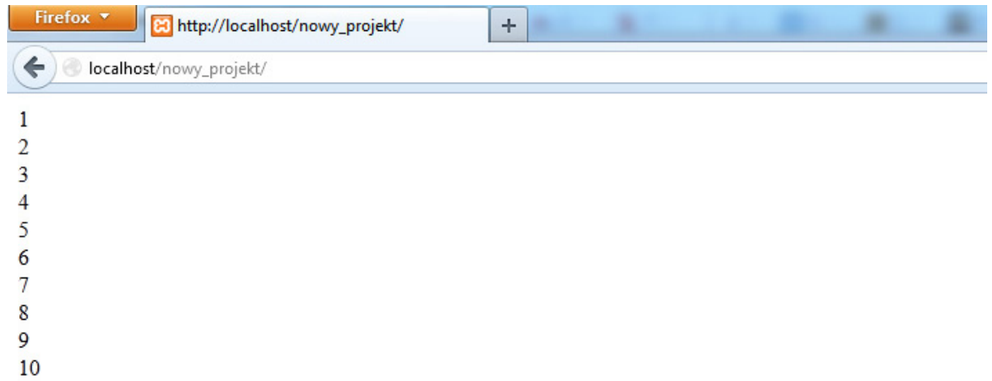
Pętla służy do wielokrotnego wykonywania zawartego w niej kodu. Aby to zrobić określoną ilość razy, można użyć pętli FOR.
Pętla FOR posiada następująca składnię:

```
for( inicjalizacja zmiennej ; warunek ; modyfikacja zmiennej) {
        //wyrażenie do wykonania
}
```


Przykład 15.

```
<?php

        for( $x = 1; $x <= 10; $x++ ) {
                echo $x.”<br>”;
        }

?>
```

Przykładowa pętla wyświetla liczby od 1 do 10.
Zmiennej „$x” zostaje przypisana wartość 1 następnie sprawdzany jest warunek czy „$x” jest mniejsze lub równe 10. Warunek jest prawdziwy. Kod „echo...” zostanie wykonany.
Zmienna „$x” zostaje zwiększona o 1 (postinkrementacja ++) czyli jej wartość w tym momencie wynosi już 2. Warunek nadal jest spełniony. Pętla jest powtarzana do momentu gdy warunek nie zostanie spełniony czyli gdy zmienna „$x” osiągnie wartość 11.



Obrazek 15

2.9 Pętla WHILE

Pętla WHILE jest używana w przypadku gdy dana operacja ma zostać wykonywana aż do momentu spełnienia określonego warunku.
Efekt z przykładu numer 14 można uzyskać również z użyciem pętli WHILE.

Przykład 16.

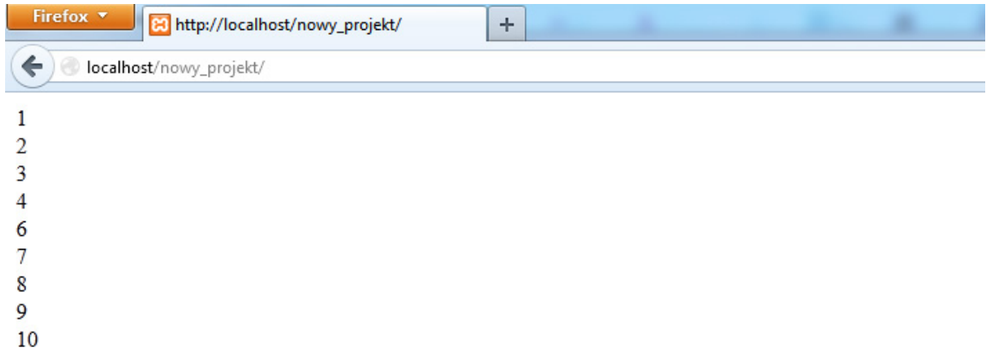
```
<?php

$x=1;

while ($x<=10) {

   echo $x."<br>";

   $x++;

}

?>
```

Jeśli zachodzi potrzeba natychmiastowego ponownego wykonania pętli bez wykonywania dalszej części kodu, należy użyć polecenia continue.

Przykład 17.

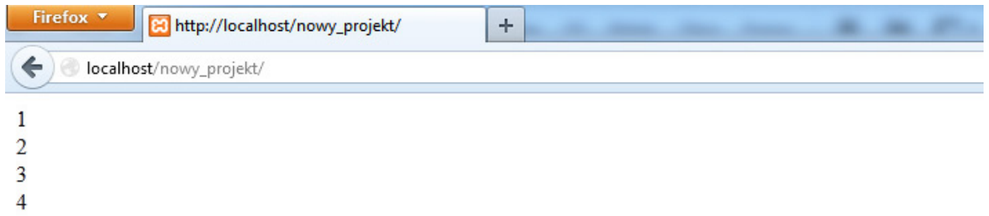
```
<?php

$x=0;

while ($x<10) {

   $x++;

   if ($x==5) {
      continue;
   }
   echo $x."<br>";

}

?>
```

Cyfra 5 nie zostanie wyświetlona.



Obrazek 16

Jeśli pętla ma zostać przerwana, polecenie continue wystarczy zastąpić poleceniem break.



Obrazek 17

2.10 Pętla DO... WHILE

Pętla DO... WHILE działa na tej samej zasadzie jak pętla WHILE. Różnica polega na tym, że warunek pętli sprawdzany jest na jej końcu, a nie początku. Dlatego niezależnie od spełnienia warunku czy też nie, pętla zostanie wykonana przynajmniej jeden raz.

Przykład 18.

```
<?php

$x=1;

do {

   echo $x."<br>";
```

```
    $x++;

} while ($x<=10)

?>
```

Efekt działania kodu jest taki sam jak w przykładnie numer 14.

2.11 Instrukcja SWITCH

3. Programowanie Zorientowane Obiektowo (OOP)

3.1 Klasy
Klasa to “szablon” do tworzenia konkretnych egzemplarzy obiektów. Posiada swoje właściwości (zmienne) oraz metody (funkcje). Inaczej mówiąc, klasy to zbiory (grupy) funkcji.
Klasę tworzy się za pomocą słowa kluczowego class.
Nazwy klas zaczynają się od dużej litery.

Przykład 19.

```
class Nazwaklasy {

        //metody, właściwości...

}
```

3.2 Metody
Funkcje w klasach to metody.

Przykład 20.

```
class Nazwaklasy {

        function nazwa_metody() {

                //jakiś kod...

        }

}
```

3.3 Dziedziczenie
Dziedziczenie klasy pozwala na przejęcie metod oraz właściwość przez inną klasę i rozszerzenie jej o nowe funkcje. Dzięki temu można rozszerzyć istniejącą już klasę i utworzyć zupełnie nowy obiekt.
Aby rozszerzyć klasę, należy użyć polecenia extends.

Przykład 21.

```
<?php

   class Samochod {

      public $marka=’Fiat’;

      public function pokazMarke() {

         echo $this->marka.’<br/>’;

      }

      public function odpal() {

         echo ‘bruuumm<br/>’;

      }

   }

$obiekt = new Samochod();
$obiekt->marka=’Opel’;
$obiekt->pokazMarke();  //zostanie wyświetlony napis Opel


class InnySamochod extends Samochod {

   public $kolor;

   public function pokazKolor() {

      echo $this->kolor.’<br/>’;
      parent::odpal();
   }

}

$nowyobiekt = new InnySamochod();
$nowyobiekt->marka=’Ford’;
```



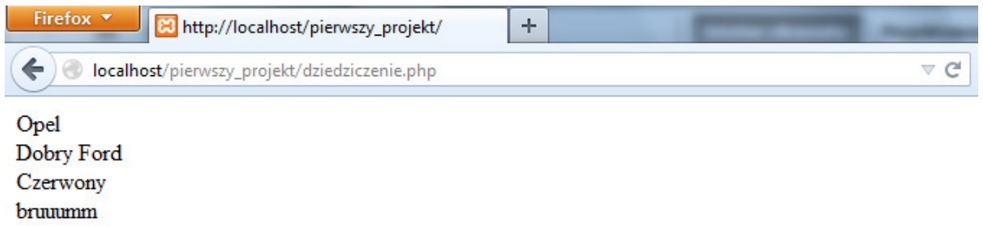
```
$nowyobiekt->kolor=’Czerwony’;
$nowyobiekt->pokazMarke();  //zostanie wyświetlony napis Ford
$nowyobiekt->pokazKolor();  //zostanie wyświetlony napis Czerwony
                            //zostanie wyświetlony napis bruuumm

?>
```

Jak widać w przykładzie numer 1, klasa InnySamochod() nie posiada metody pokazMarke(), jednak klasa ta jest rozszerzeniem klasy Samochod() w której ta metoda się znajduje, dlatego metoda pokazMarke() występuje również w niej.
Tym sposobem klasa InnySamochod() posiada metody klasy po której dziedziczy.
Dodatkowo klasa została rozszerzona o metodę pokazKolor().
W dowolnym momencie można dodać kolejne metody w klasie Samochod(), które automatycznie będą dostępne również w klasie InnySamochod();
Jeśli podklasa nie posiada własnego konstruktora to zostanie użyty konstruktor klasy nadrzędnej.
Dostęp do dowolnej metody klasy nadrzędnej (tzw. superklasy) można uzyskać poprzez polecenie parent::
Metoda pokazKolor() zawiera odwołanie do metody odpal() superklasy. Dzięki temu po wywołaniu metody pokazKolor(), samochód zostanie „odpalony”
i wyświetlony zostanie napis „bruuumm”.
Należy pamiętać że wielodziedzieczenie nie jest obsługiwane. Oznacza to, że dziedziczenie z więcej niż jednej klasy nie jest możliwe.
W klasie podrzędnej można nadpisać metody superklasy. Wystarczy utworzyć metodę o takiej samej nazwie, jak metoda która ma zostać nadpisana.

Przykład 22.

```
<?php
…
   class InnySamochod extends Samochod {

      public $kolor;

      public function pokazKolor() {

         echo $this->kolor.’<br/>’;

         parent::odpal();
```

```
    }

    public function pokazMarke() {

        echo 'Dobry '.$this->marka.'<br/>';

    }

}

$nowyobiekt = new InnySamochod();
$nowyobiekt->marka='Ford';
$nowyobiekt->kolor='Czerwony';
$nowyobiekt->pokazMarke();  //zostanie wyświetlony napis Dobry Ford
$nowyobiekt->pokazKolor();  //zostanie wyświetlony napis Czerwony
                            //zostanie wyświetlony napis bruuumm
?>
```



Obrazek 18

Do klasy InnySamochod() została dodana metoda pokazMarke() która występuje również w superklasie Samochod(). Dzięki temu metoda została nadpisana i użycie pokazMarke() spowoduje wyświetlenie napisu „Dobry Ford", a nie samo „Ford".
Można uniemożliwić nadpisanie metody za pomocą słowa kluczowego final.

Przykład 23.

```
<?php
…
public final function pokazMarke() { … }
…
?>
```


W tym momencie próba nadpisania metody pokazMarke() skończy się wygenerowaniem błędu.
Słowo kluczowe final ma zastosowanie również w przypadku klas. Jego użycie uniemożliwi rozszerzenie klasy.

Przykład 24.

```
<?php

   final class Samochod {
…
   }
class InnySamochod extends Samochod {
…
}
?>
```

Próba utworzenie klasy InnySamochod() dziedziczącej po klasie Samochod() w typ przypadku spowoduje wyświetlenie błędu.

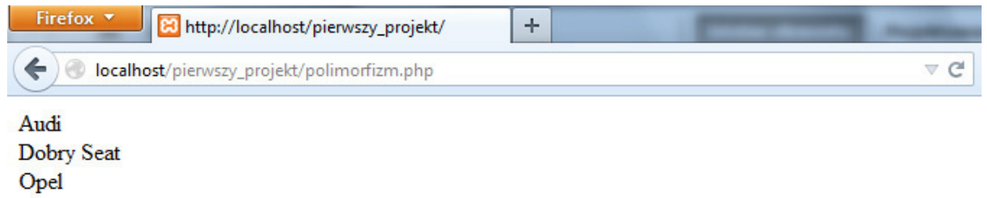
3.4 Polimorfizm

Dzięki polimorfizmowi (tzw. wielopostaciowość) możliwe jest wywołanie metod o takiej samej nazwie, zawartych w różnych obiektach (egzemplarzach klasy).

Przykład 25.

```
<?php

   class Samochod {

      public $marka;

      public function pokazMarke() {

         echo $this->marka.'<br/>';

      }

   }

   class Kabrio extends Samochod {
```

```
}

class Sedan extends Samochod {

   public function pokazMarke() {

      echo 'Dobry '.$this->marka.'<br/>';

   }

}

$obiekt1 = new Kabrio();
$obiekt1->marka='Audi';
$obiekt1->pokazMarke(); //zostanie wyświetlony napis Audi

$obiekt2 = new Sedan();
$obiekt2->marka='Seat';
$obiekt2->pokazMarke(); //zostanie wyświetlony napis Dobry Seat

?>
```



Obrazek 19

Zostały stworzone dwa obiekty klas, które zawierają metody o takich samych nazwach. Obie klasy dziedziczą po superklasie Samochod().
Każdy obiekt wykorzystuje metodę pokazMarke() na własny sposób i „nie wchodzą sobie w drogę".
Tym właśnie jest polimorfizm.

3.5 Klonowanie

Klonowanie umożliwia stworzenie kopii obiektu.

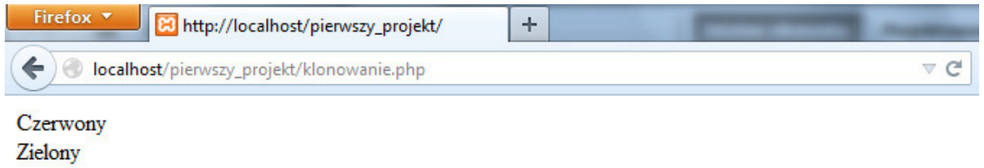
Przykład 26.

```
<?php
…
   $obiekt1 = new Samochod();
   $obiekt1->kolor='Czerwony';

   $obiekt2 = $obiekt1;
   $obiekt2->kolor='Zielony';

   echo $obiekt1->kolor.'<br/>'; //wyświetlony zostanie napis Zielony
   echo $obiekt2->kolor; //wyświetlony zostanie napis Zielony
?>
```

Utworzony został nowy obiekt klasy Samochod(). Właściwości kolor przypisano „Czerwony". Następnie obiekt został skopiowany:
$obiekt2 = $obiekt1;
Ponieważ $obiekt2 zawiera teraz metody i właściwości takie same jak $obiekt1, można właściwości kolor w tym obiekcie również nadać jakąś wartość np. Zielony.
Efekt? Obydwa samochody zostały „przemalowane" na zielony.
Dzieję się tak dlatego że obiekt $obiekt2 jest jedynie odniesieniem do $obiekt1. Nie jest natomiast samodzielnym klonem. Aby $obiekt2 był samodzielną kopią należy użyć słowa kluczowego clone.

Przykład 27.

```
<?php
…
   $obiekt1 = new Samochod();
   $obiekt1->kolor='Czerwony';

   $obiekt2 = clone $obiekt1;
   $obiekt2->kolor='Zielony';

   echo $obiekt1->kolor.'<br/>'; //wyświetlony zostanie napis Czerwony
   echo $obiekt2->kolor; //wyświetlony zostanie napis Zielony

?>
```



Obrazek 20

Tym razem każdy obiekt klasy Samochod() ma własny, niezależny kolor. $obiekt1 i $obiekt2 to teraz dwa odrębne obiekty.

3.6 Konstruktory i destruktory

Konstruktor to metoda wewnątrz klasy która jest wykonywana automatycznie w trakcie tworzenia obiektu (egzemplarza klasy). Metoda konstruktora może występować w dwóch wariantach:

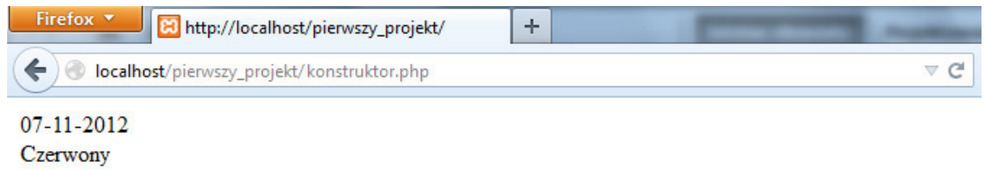
- metoda o nazwie __construct()
- metoda o nazwie identycznej jak nazwa klasy w której się znajduje

Przykład 28.

```
<?php

   class Samochod {

      public $kolor;
      public $data;


      public function __construct() {

        echo $this->data=date("d-m-Y", time()).'<br/>';

      }


      public function pokazKolor() {

         echo $this->kolor.'<br/>';

      }
```



```
   }

   $obiekt = new Samochod();  //wyświetlona zostanie aktualna data
   $obiekt->kolor='Czerwony';
   $obiekt->pokazKolor();       //wyświetlony zostanie napis Czerwony

?>
```



Obrazek 21

Powyższy przykład zawiera metodę konstruktora __construct(). Jest to pierwszy wariant. Zmiana nazwy metody na samochod() czyli na taką samą jak nazwa klasy (drugi wariant) nie spowoduje zmian.

```
public function samochod() { … }
```

Działanie konstruktora widać w trakcie tworzenia nowego obiektu klasy Samochod():

```
$obiekt = new Samochod();
```

Już w tym momencie wyświetlony zostanie napis z aktualną datą gdyż metoda __construct() jest uruchamiana automatycznie w przeciwieństwie do metody pokazKolor() którą trzeba wywołać „ręcznie".

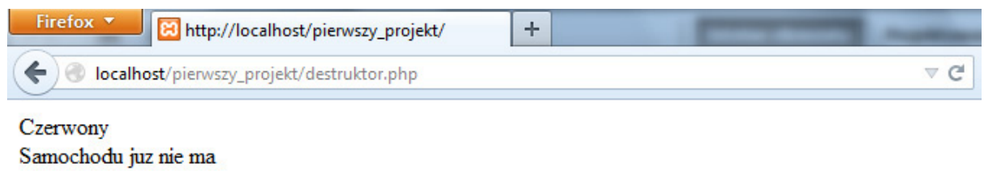
Jeśli w klasie znajdą się dwa konstruktory jednocześnie: __construct() oraz samochod() (w przypadku jak w przykładzie 8). to wykonana zostanie metoda __construct() natomiast metoda o nazwie identycznej jak nazwa klasy czyli samochod() zostanie zignorowana.

Destruktor jest to metoda, która wykonuje się automatycznie podczas niszczenia obiektu.

Metoda destruktora nosi nazwę __destruct()

Przykład 29.

```
<?php

   class Samochod {

      public $kolor;

      public function pokazKolor() {

         echo $this->kolor.'<br/>';

      }


      public function __destruct() {

         echo "Samochodu juz nie ma";

      }

   }

   $obiekt = new Samochod();
   $obiekt->kolor='Czerwony';
   $obiekt->pokazKolor(); // wyświetlony zostanie napis Czerwony
                          // wyświetlony zostanie napis Samochodu juz nie ma

?>
```



Obrazek 22

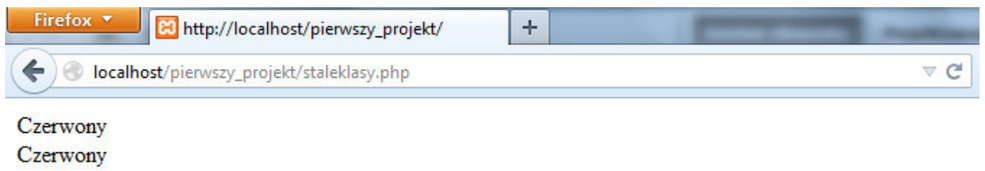
Natychmiast po wykonaniu metody pokazKolor() automatycznie została uruchomiona metoda destruktora, która wyświetliła napis „Samochodu juz nie ma".

3.7 Stałe klasy

Stałe działają podobnie do zmiennych statycznych jednak ich zawartość można jedynie odczytać.
Definiowanie stałej w klasie odbywa się za pomocą słowa kluczowego const, a nie za pomocą znaku dolara ($) jak w przypadku zmiennych.

Przykład 30.

```
<?php

   class Samochod {

      const KOLOR = ,Czerwony';

      public function pokazKolor() {

         echo self::KOLOR.'<br/>';

      }

   }

   echo Samochod::KOLOR.'<br/>';   //zostanie wyswietlony napis Czerwony
   $obiekt = new Samochod();
   $obiekt->pokazKolor(); //zostanie wyswietlony napis Czerwony

?>
```



Obrazek 23

Dostęp do stałej klasowej wewnątrz klasy uzyskuję się za pomocą słowa kluczowego self i operatora ::

Dostęp do stałych klasowych jest publiczny, a więc w każdym momencie można wywołać kod:

echo Samochod::KOLOR; //zostanie wyswietlony napis Czerwony

Dostęp do stałej klasowej poza klasą uzyskuję się również za pomocą operatora ::
Należy podać nazwę klasy przed operatorem :: i nazwę stałej po operatorze.

3.8 Klasy i metody abstrakcyjne, Interfejsy.
Klasa abstrakcyjna jest to klasa której obiekty nie mogą być tworzone, może być natomiast dziedziczona (rozszerzana).
Klasę abstrakcyjną tworzy się za pomocą słowa kluczowego abstract.

Przykład 31.

```
<?php
   abstract class Samochod {
      public $kolor;
      public function pokazKolor() {
         echo 'ma kolor '.$this->kolor;
      }
      abstract public function pokazMarke(); // metoda pokazujaca marke

   }
   class Osobowy extends Samochod {
      public function pokazMarke() {
         echo 'Polonez ';
      }
   }

   $obiekt = new Osobowy();
   $obiekt->kolor='czerwony';
   $obiekt->pokazMarke(); //wyswietlony zostanie napis
   $obiekt->pokazKolor();  // Polonez ma kolor czerwony

?>
```


Napisana została klasa abstrakcyjna Samochod() następnie klasa Osobowy() po niej dziedzicząca.
Próba stworzenia obiektu klasy abstrakcyjnej Samochod() skończyłaby się wygenerowaniem błędu. Utworzony zostaje więc obiekt klasy Osobowy(). Obiekt ten dziedziczy metody klasy abstrakcyjnej Samochod() dzięki czemu możliwe jest wywołanie metod pokazMarke() (wyświetli 'Polonez ') i pokazKolor() (wyświetli 'ma kolor czerwony').

W przykładzie 11 klasa abstrakcyjna posiada metodę abstrakcyjną pokazMarke(). Każda klasa zawierająca choćby jedną abstrakcyjną metodę musi być abstrakcyjna. Jeżeli dziedziczymy po klasie abstrakcyjnej, wszystkie metody abstrakcyjne muszą być przedeklarowane w klasie dziedziczącej. Dlatego też klasa Osobowy() również posiada metodę pokazMarke() która nadpisuje dziedziczoną metodę abstrakcyjną klasy Samochod(). Jeśli metoda abstrakcyjna nie zostanie nadpisana w klasie dziedziczącej, dojdzie do wygenerowania błędu.

W przypadku gdyby klasa Osobowy() również byłaby klasa abstrakcyjną, nie ma konieczności przedeklarowania dziedziczonych metod abstrakcyjnych.

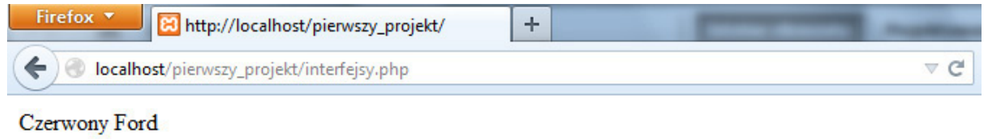
Modyfikator dostępu metody abstrakcyjnej nie może być bardziej restrykcyjny od zadeklarowanego w klasie abstrakcyjnej po której dziedziczy, czyli nie można zamienić modyfikatora public na private, ale private na public jak najbardziej.

Interfejs jest kasą zawierającą wyłącznie deklaracje metod. Można powiedzieć że interfejs jest mocno abstrakcyjną klasą.
Metody w interfejsie są domyślnie abstrakcyjne, a więc, wszystkie klasy

implementujące interfejs muszą zawierać metody je nadpisujące.
Interfejs wymusza na klasach go implementujących aby posiadały niezbędne metody, a więc interfejs jest zbiorem reguł.
Interfejs tworzony jest za pomocą słowa kluczowego interface.

Przykład 32.

```
<?php

interface Samochod {

   public function pokazKolor();
   public function pokazMarke();
}

class Osobowy implements Samochod {

   public $kolor;
   public $marka;

   public function pokazKolor() {

      echo $this->kolor.’ ‘;
      $this->pokazMarke();

   }

   public function pokazMarke() {
      echo $this->marka;
   }

}
$obiekt = new Osobowy();
$obiekt->kolor=’Czerwony’;
$obiekt->marka=’Ford’;
$obiekt->pokazKolor(); // wyswietlony zostanie napis Czerwony Ford
?>
```

Obrazek 24

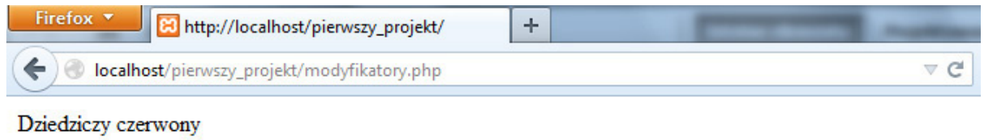
Klasa Osobowy() implementuje za pomocą słowa kluczowego implements interfejs Samochod() i dziedziczy jego metody.
Trzeba pamiętać, że interfejs zawiera wyłącznie deklaracje metod i nie można w nich umieszczać definicji. Możliwe jest to dopiero w klasie implementującej interfejs gdzie metody należy nadpisać.

3.9 Dziedziczenie a modyfikatory dostępu.
Modyfikatory dostępu to słowa kluczowe określające poziom dostępu do metod i właściwości danej klasy.
Występują trzy modyfikatory dostępu: public, private i protected.

Przykład 33.

```
<?php

   class Samochod {

      public $kolor;  //dziedziczy

      public function pokazKolor() {

         echo ,Dziedziczy  ,.$this->kolor;

      }

      private function pokazMarke() {

         echo ,Nie dziedziczy’;

      }

      protected function pokazRejestracje() {
```

```
        echo ‘Dziedziczy’;

    }


}

$obiekt = new Samochod();
$obiekt->kolor=’czerwony’;
$obiekt->pokazKolor(); //wyswietla napis Dziedziczy czerwony

$obiekt-> pokazMarke ();  //wywietla błąd (brak dostępu)
$obiekt-> pokazRejestracje ();//wywietla błąd (brak dostępu)

?>
```



Obrazek 25

Public – modyfikator ten oznacza, że każda właściwość lub metoda zadeklarowana w ten sposób, jest dostępna z wewnątrz i zewnątrz klasy oraz w klasach pochodnych (dziedziczących).
Public jest modyfikatorem domyślnym. Oznacz to, że każda właściwość lub metoda nie posiadająca modyfikatora dostępu, domyślnie jest publiczną.

Private – właściwość lub metoda z użyciem tego modyfikatora, nie jest dostępna z zewnątrz klasy, natomiast metody tej samej klasy mają do nich dostęp. Modyfikator ten sprawia, że właściwości i metody nim poprzedzone nie są dostępne w klasach pochodnych.

Protected – właściwości lub metody z tym modyfikatorem są dostępne wyłącznie wewnątrz swojej klasy tak jak w przypadku modyfikatora private, jednocześnie są dostępne w klasach pochodnych jak w przypadku modyfikatora public. Dostęp z zewnątrz klasy nie jest jednak możliwy.

3.10 Agregacja i kompozycja
Agregacja oznacza tworzenie klasy przy użyciu innych istniejących klas (obiektów składowych). Tworzona klasa może składać się z dowolnej liczby obiektów. Agregację często nazywa się relacją typu „zawiera” np. samochód zawiera silnik.

Przykład 34.

```
<?php

class Naped {

  public function silnik() {
     return ‘bruuumm’;
  }
}

class Samochod {

  public $pod_maska;

  public function odpal() {

     $this->pod_maska = new Naped();
     echo $this->pod_maska->silnik();

  }

}

$obiekt = new Samochod();
$obiekt->odpal();

?>
```

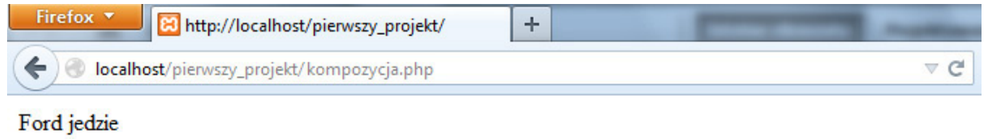
Klasa Samochod w metodzie odpal() zawiera obiekt obiekt klasy Naped().

Kompozycja jest szczególnym przypadkiem agregacji. Różni się od agregacji tym, że klasa składa się z obiektów, które bez tej klasy istnieć by nie mogły.
Kompozycja nazywana jest relacją typu „posiada” np. samochód posiada silnik.

Przykład 35.

```
<?php

class Silnik {

  public function odpal() {

    echo “Ford jedzie”;

    }
  }


class Samochod {

  private $auto;

  public function __construct() {

    $this->auto = new Silnik();

  }

  public function odpal() {

    return $this->auto->odpal();

  }
}

$obiekt = new Samochod();
$obiekt->odpal();


?>
```

Klasa Samochod() posiada obiekt Silnik().



Obrazek 26

GRAFIKA 2D

Addytywny model barw: metoda uzyskiwania dowolnej barwy przez zmieszanie w różnych proporcjach światła o trzech barwach prostych. Ze względu na dominującą rolę w widmie barwnym promieniowania w zakresie czerwieni, zieleni i niebieskiego, model addytywny najlepiej sprawdza się w przypadku użycia RGB jako jego barw podstawowych.

Barwa niekolorowa inaczej: achromatyczna lub neutralna; barwa nie zawierająca wyróżniającej składowej koloru, czyli szarość mieszająca się między czernią a bielą; barwa o zerowym nasyceniu.

Barwa prosta: inaczej: monochromatyczna; zjawisko barwne wywołane światłem o jednej długości fali (np. 500 nm).

Barwa: wrażenie wywołane w mózgu przez działające na oko promieniowanie elektromagnetyczne w zakresie widma widzialnego (380-780 nm). Istnieją rozmaite sposoby opisu barwy, np. przy użyciu przestrzeni barwnych lub za pomocą rozkładu widmowego.

CMYK: skrót z ang. nazw barw farb triadowych (Cyan, Magenta, Yellow, blacK lub Key). Jest to tzw. substraktywny model barw oparty o mieszanie farb CMY. Mieszaniu teoretycznie mogą podlegać trzy barwy CMY, aby uzyskać wszystkie odcienie barw, w tym także szarości, aż do pełnej czerni, lecz w wyniku niedoskonałości farb stosuje się dodatkowo farbę czarną dla nadania czystości jej odcienia. Model ten stosowany jest powszechnie do uzyskiwania wielobarwnych odbitek drukowych.

Color Management System (CMS): system zarządzania barwą, zapewniający komunikację między urządzeniami (wejścia, wyświetlania i wyjścia) w celu zapewnienia dokładności reprodukcji barw.

Densytometr: przyrząd pomiarowy służący do wyznaczania gęstości optycznej dla światła przechodzącego (densytometr transmisyjny) lub odbitego (densytometr refleksyjny).

EPS: z ang. Encapsulated PostScipt, czyli PostScript w pigułce. Plik postscriptowy, który można umieszczać w innym pliku postscriptowym. Stosowany do zapisu grafiki bitmapowej, wektorowej oraz łączącej oba te rodzaje.

Gama barw: inaczej: gamut; zakres barw możliwych do zreprodukowania przez dane urządzenia

Gęstość optyczna: wielkość charakteryzująca ilość promieniowania przechodzącego przez materiał transparentny (diapozytyw lub negatyw) lub odbitego od materiału nieprzezroczystego (fotografia, odbitka drukarska). Jej wartość oblicza się jako ujemny logarytm współczynnika przepuszczania lub odbijania światła. Teoretycznie gęstość optyczna może przybierać wartość od zera (ciało doskonale białe lub idealnie przezroczyste) do nieskończoności (ciało doskonale czarne), lecz w praktyce poligraficznej jej zakres jest ograniczony przedziałem 0,03-5 dla materiałów transparentnych, zaś dla materiałów odbijających światło mieści się w granicach 0,1-3.

Głębia bitowa: liczba bitów opisujących możliwe do uzyskania wartości tonalne jednego piksela obrazu. 1-bitowa głębia odpowiada dwóm wartością, najczęściej barwie czarnej i białej; 8-bitowa oznacza 256 wartości (najczęściej odcieni szarości), 24- bitowa 16 777 216 wartości. Współcześnie produkowane skanery często używają głębi 36, 42 czy nawet 48 bitów (3x12, 14, 16); zebrany w ten sposób nadmiar informacji umożliwia dokonanie bardzo głębokich korekcji obrazu cyfrowego bez utraty jakości.

Grafika wektorowa: Obraz wygenerowany na podstawie matematycznych opisów określających pozycję, długość oraz kierunek kreślenia linii.

Hexachrome opatentowana: przez firmę Pantone przestrzeń barwna oparta na sześciu podstawowych barwach (farbach drukarskich). Oprócz nieco zmodyfikowanej triady CMYK stosowane są dodatkowo farba zielona i pomarańczowa. Umożliwia to reprodukcję o wiele szerszej gamy barw niż CMYK.

Impozycja z ang. imposition (montaż): rozmieszczenie drukowanych elementów (stron, użytków) na formie kopiowej lub drukowej, z uwzględnieniem sposobu drukowania i oprawy; np. tworzenie składek introligatorskich. Terminem tym określa się tzw. montaż elektroniczny, wykonywany przy użyciu komputera.

JPEG: jeden z popularnych formatów zapisu plików graficznych. Dzięki specjalnej metodzie kompresji, zwanej kompresją stratną (można regulować poziom szczegółowości ilustracji: im mniej szczegółów, tym mniejszy plik), pliki JPEG mogą osiągać bardzo małe rozmiary, lecz kosztem jakości obrazu. W żadnym razie nie wolno wielokrotnie zapisywać pliku z kompresją JPEG kompresja stratna każdorazowo pogłębi degradację obrazu.

Kalibracja: regulacja urządzenia na drodze pomiaru jego odchyłek od wartości standardowych i użycia zebranych danych do kompensacji owych odchyłek podczas normalnej pracy. Kalibrację w procesach reprodukcyjnych stosuję się w celu zwiększania dokładności odtwarzania urządzeń wejściowych i wyjściowych, jak skanery, monitory naświetlarki.

Mapa bitowa: obraz utworzony z siatki pikseli lub kropek narzędzia do przesuwania pozwalają przesuwać zaznaczenia, warstwy i linie pomocnicze.

Narzędzia do zaznaczania: pozwalają zaznaczać obszary prostokątne, obszary w kształcie elips oraz pojedyncze wiersze i kolumny.

Narzędzia typu Lasso: pozwalają zaznaczać obszary dowolne, ograniczone wielokątami oraz przyciągnięte do jakichś elementów.

Narzędzie Różdżka: pozwala zaznaczyć obszary o podobnych kolorach.

Obiekt: element rysunku, taki jak obrazek, kształt, linia, tekst, krzywa, symbol lub warstwa.

Okno dokowane: okno zawierające dostępne polecenia i ustawienia związane z danym narzędziem lub zadaniem.

Paleta wysuwana: przycisk otwierający grupę powiązanych narzędzi lub elementów menu.

PostScript: język programowania wysokiego poziomu, służący do opisu strony do druku, wykorzystywany do przekazywania informacji z komputera do urządzeń wyjściowych (np. naświetlarki). Zawiera komendy umożliwiające umieszczenie grafiki rastrowej i wektorowej, nadawanie parametrów barwnych obiektom, decydowanie o rastrowaniu itp. Stworzony w połowie lat 80. przez firmę Adobe, zrewolucjonizował proces prepress, przenosząc do na płaszczyznę systemów otwartych.

Przestrzeń barwna: zestaw parametrów opisujących barwę jako punkt w przestrzeni zbudowanej względem trzech osi (np. RGB, CMY, XYZ, Lab)

Raster: struktury reprodukowanego obrazu, złożone z regularnie rozmieszczonych punktów o zmiennej wielkości (raster tradycyjny, krzyżowy, AM) albo nieregularnie rozmieszczonych punktów stałej wielkości (raster stochstyczny, FM)

RGB: skrót z ang. nazw barw filtrów (Red, Green, Blue). Stanowi tak zwany addytywny model barwy. Stosowany powszechnie do uzyskiwania wielobarwnych obrazów , np. w telewizji czy na monitorze komputera oraz , w przypadku skanerów, do analizy obrazu wielobarwnego. Jest ściśle związany z określonym typem urządzenia.

Rozbarwienie: separacja, wyciąg barwny; zmiana danych graficznych kompozytowych, czyli takich, gdzie elementowi obrazu przypisane są łącznie wartości w jakimś modelu barwnym (np. RGB, Lab, CMYK) na osobne obrazy w odcieniach szarości, odpowiadające poszczególnym farbom drukarskim (najczęściej CMYK). Niekiedy rozbarwieniem nazywa się sam proces konwersji danych barwnych do modelu CMYK. Rozbarwienie jest również stosowane na określenie gotowych form kopiowych (filmów) służących do drukowania.

Rozdzielczość: liczba pikseli obrazu przypadająca na jednostkę długości. Używane jednostki do dpi dots per inch, czyli punkty na cal (rzadziej ppi piksele na cal) oraz dpc punty na centymetr.

Rysunek: praca tworzona w programie CorelDRAW, na przykład różne grafiki, logo, plakaty i biuletyny.

Spektrofotometr: przyrząd pomiarowy badający światło emitowane przez ciało (np. przez monitor komputera) lub światło odbite od ciała (np. od powierzchni papieru, farby), w wielu zakresach spektralnych (widmowych); wynik pomiaru może przybrać postać wykresu widmowego, a także wartości RGB, HSB, CMY, CMYK, czy nawet gęstości optycznej.

Substraktywny model barw: metoda uzyskiwania dowolnej barwy przez filtrowanie światła przy użyciu trzech filtrów (farba drukarska, toner, tusz itd.) umieszczonych na białym podłożu. Barwy filtrów: niebieskozielony (cyan), purpurowy (magenta) i żółty (yellow) są barwami dopełniającymi triady addytywnego modelu RGB.

Tekst akapitowy : rodzaj tekstu, do którego można zastosować opcje formatowania i który można formatować w dużych blokach.

Tekst ozdobny: rodzaj tekstu, do którego można zastosować efekty specjalne, np. cień.

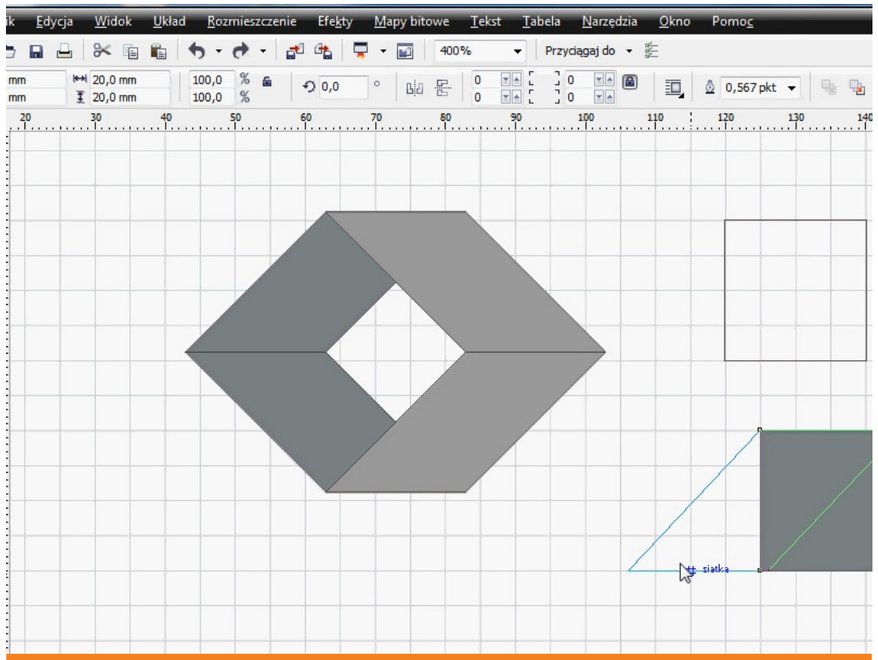
UV ang. ultraviolet: niewidzialne promieniowanie o częstotliwości większej (a długości fal mniejszej) niż światło fioletowe (poniżej 380 nm); wykorzystywane w poligrafii np. przy kopiowaniu obrazu z filmu na formy offsetowe oraz do utrwalania farb i lakierów.

Wtyczki: to niewielkie programy, które zwiększają możliwości lub dodają nowe właściwości do twojego oprogramowania.

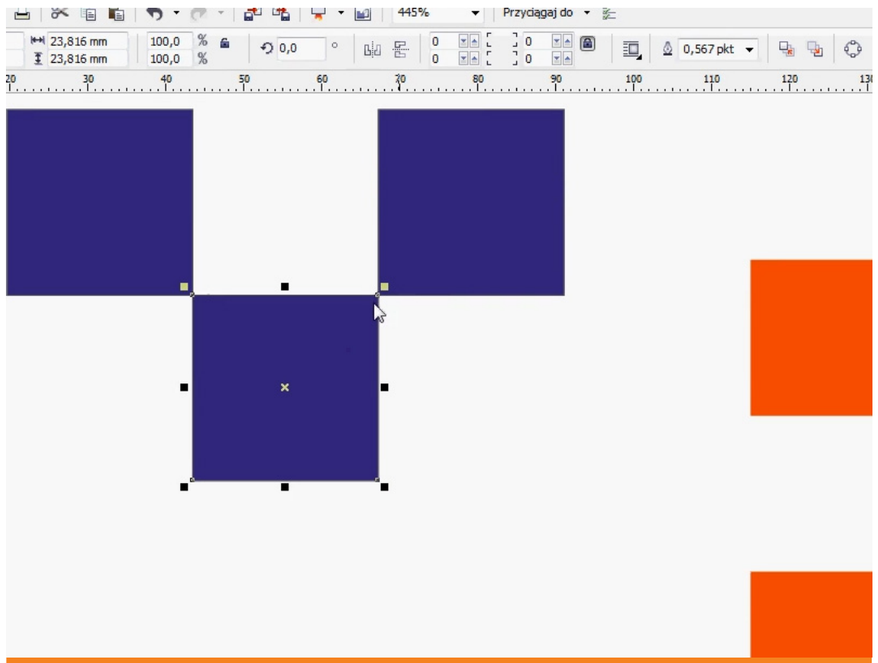
Zalewki: elementy tworzone na granicy dwóch farb w drukowaniu wielobarwnym, zapobiegające widocznym błędom pasowania.

Zalewkowanie: ang. trapping; tworzenie zalewek.

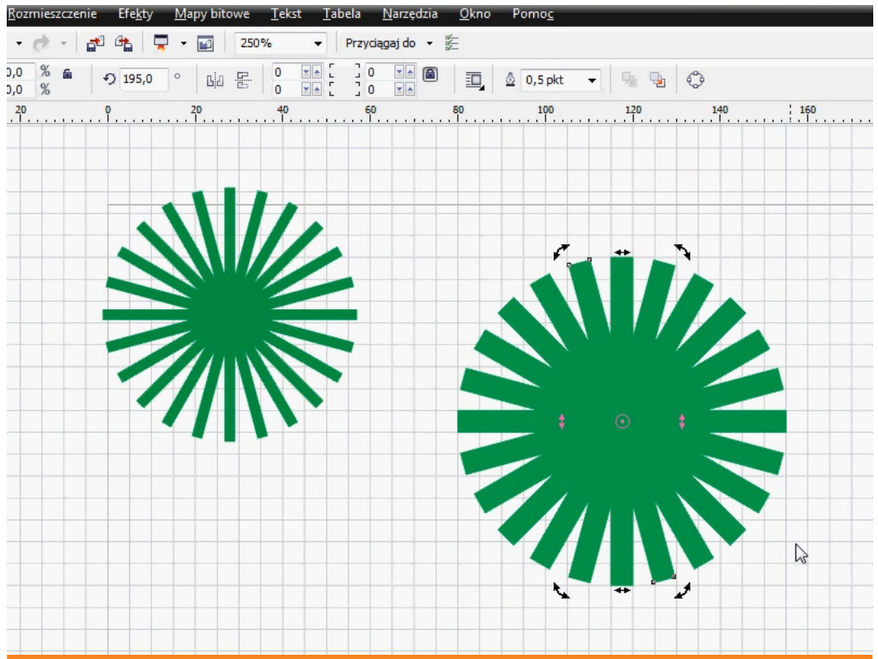
Siatka, przyciąganie do obiektu



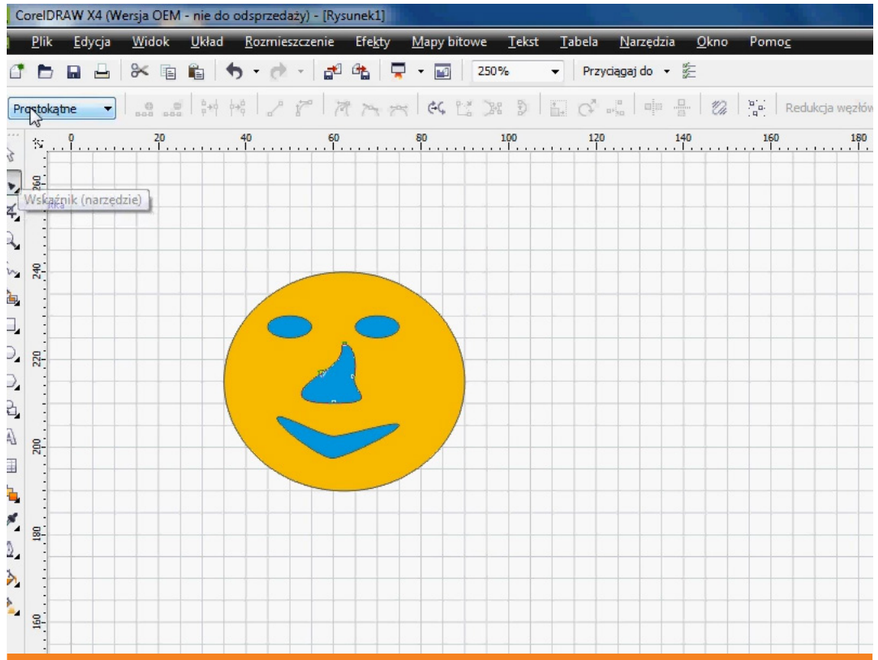
CorelDraw - siatka



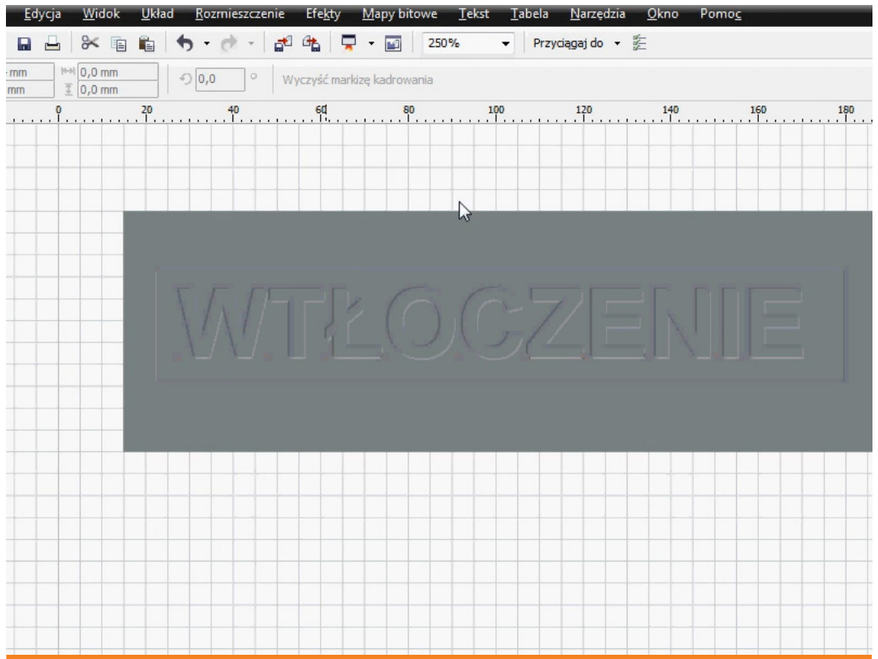
CorelDraw - przyciąganie do obiektu



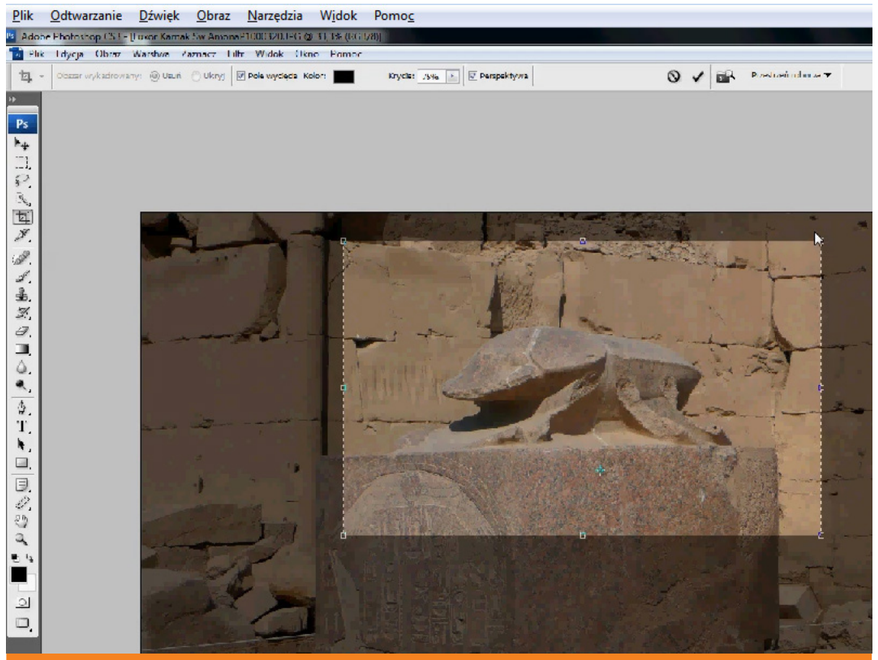
CorelDraw - powtarzanie



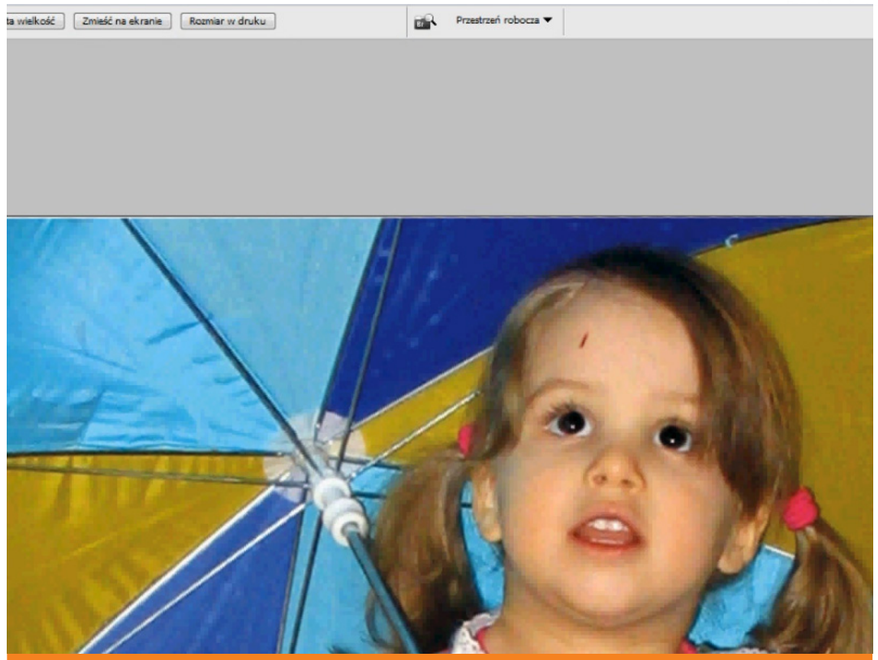
CorelDraw - grupowanie



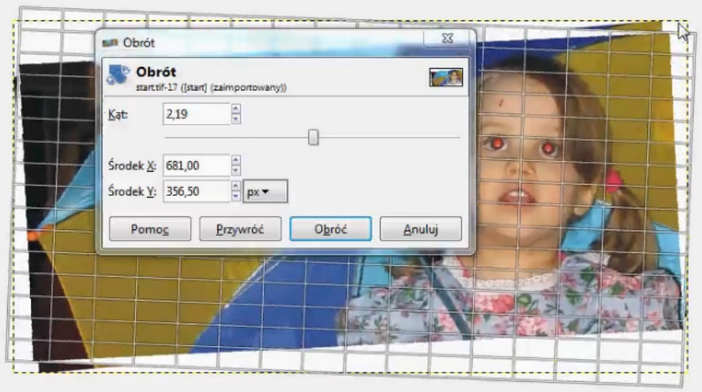
CorelDraw - napisy



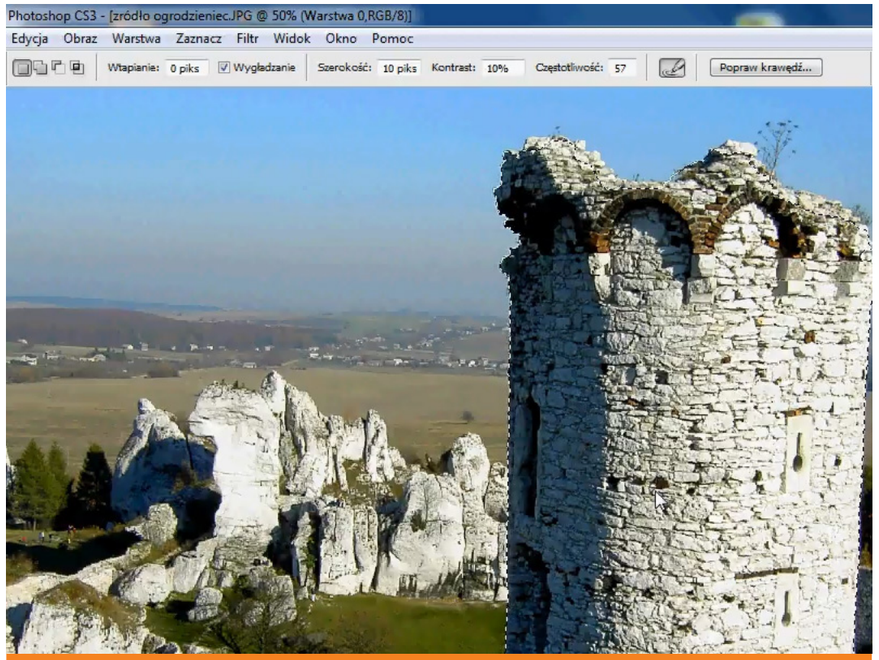
Photoshop - kadrowanie



Photoshop - korekcja zdjęć



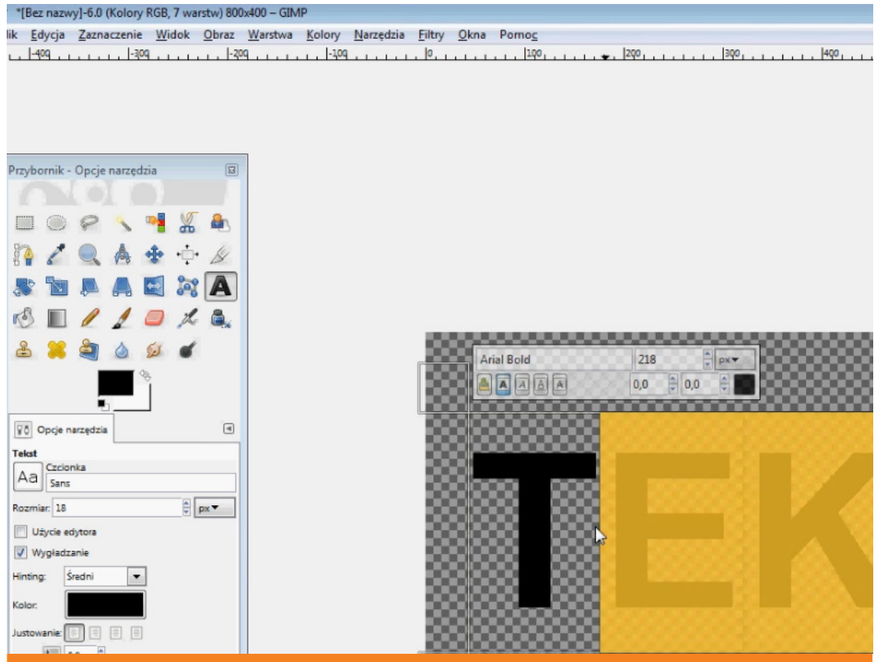
Gimp- korekcja zdjęć



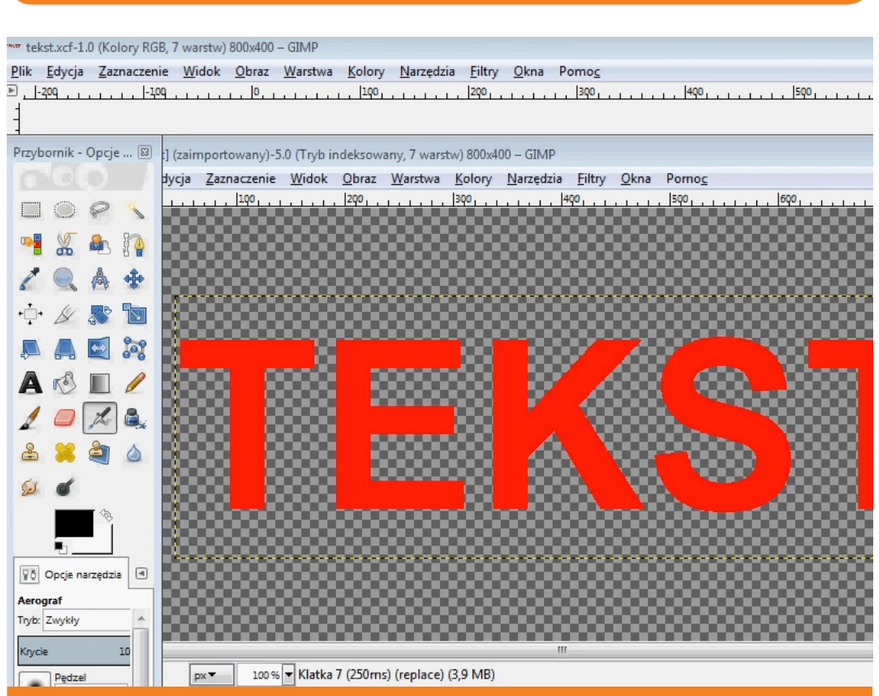
Photoshop - punkt zbiegu: podstawy

Photoshop - punkt zbiegu: zaawansowane

Gimp - animacja



Gimp animacja - cz I



Gimp animacja - cz II

GRAFIKA 3D

BLENDER

Najczęściej stosowane skróty klawiaturowe z objaśnieniem po polsku

OM = Object Mode only (tylko w trybie obiektowym)
SM = Sculpt Mode only (tylko w trybie rzeźbienia)
EM = Edit Mode only (tylko w trybie edycji)
PM = Pose Mode only (tylko w wypadku pracy z “kościami” – “armature”)

Kolejność jest alfabetyczna. Skróty z dodatkowym klawiszem np Alt + A należy rozumieć jako “A” – podstawowy, “Alt” - dodatkowy

Skrót klawiaturowy	Operacja
A	Select all / Deselect all – zaznacz / odznacz wszystko
Shift + A	Add menu - menu Dodaj
Alt + A	Play animation toggle – Uruchomienie animacji na listwie czasu
CTRL + A	Apply menu (such as freeze transforms) OM – menu Zastosuj
B	Border select (marquee select) – pojawia się kursor do zaznaczania krawędzi
Shift + B	Marquee zoom - pojawia się kursor zoomu
Shift + B	Set render border (Active Camera) – przy zaznaczonej kamerze określa obszar renderingu
C	Circle select – zaznaczanie w kształcie koła
C + kółko myszy	pozwala zwiększyć lub zmniejszyć ten obszar zaznaczenia

Skróty klawiaturowe z komentarzem

C	Clay brush SM - w trybie rzeźbienia włącza pędzel typu Clay - glina
Shift + C	3D Cursor to origin – kursor dopasowuje się do środka współrzędnych
Alt + C	Convert menu OM – pokazuje menu Konwertuj
Alt + C	Close / Open a curve EM – zamknij/otwórz krzywą
D	Draw brush SM – w trybie rzeźbienia pokazuje pędzel Draw - Rysuj
Shift + D	Duplicate - kopiuj
Alt + D	Linked Duplicate (Instance) – to też kopiowanie, ale można skopiować dowolną ilość razy trzymając Alt + D i potwierdzając kopiowanie prawym klikiem myszy
E	Extrude region EM – “Wyciągnij” obszar, dobudowuje geometrię
Alt + E	Extrude menu EM
Shift + E	Crease EM – ta funkcja pozwala wpływać na ostrość krawędzi
E	End frame assign (Timeline window) – ustawia końcową klatkę
CTRL + E	Edges menu EM – menu Krawędzie
F	Create Face (3+ vertices selected) EM – tworzy powierzchnię (przynajmniej 3 werteksy muszą być zaznaczone). Weteksy czyli, mówiąc najprościej, punkty
F	Create Edge (2 vertices selected) EM – tworzy krawędź, trzeba zaznaczyć wcześniej 2 werteksy
i	krawędź powstaje pomiędzy nimi

Skróty klawiaturowe z komentarzem

F	Brush size adjust SM – rozmiar pędzla
Shift + F	Brush strength adjust SM – siła nacisku pędzla
Shift + F	Camera Fly mode – kamera płynna
CTRL + F	Faces menu EM – menu Powierzchni (obszarów pomiędzy werteksami)
Alt + F	Fill create faces EM – wypełnij powierzchnie
G	Move (Grab) – przesuń obiekt
Alt + G	Reset location OM – resetuj położenie, obiekt wraca na miejsce, w którym był utworzony
CTRL + G	Create new group – twórz nową grupę
CTRL + Shift + G	Add selected to active group OM – dodaj zaznaczony obiekt do aktywnej grupy
Alt + Shift + G	Remove selected from active group OM – usuń zaznaczony obiekt z grupy obiektów
Alt + G	Ungroup (Node Editor) – w Edytorze Nodów pozwala rozgrupować
CTRL + G	Vertex Groups menu EM – menu Grupa Werteksów
Shift + G	Select Similar menu EM – menu Zaznacz Podobne
CTRL + G	Add selected objects to group – dodaj zaznaczone obiekty do grupy
H	Hide selected – ukryj zaznaczone, czyli ukryj to co ma być niewidoczne w tym momencie – służy to np w celu pozbycia się na chwilę nadmiaru obiektów na scenie

Skróty klawiaturowe z komentarzem

Shift + H	Hide unselected – ukryj to co jest niezaznaczone
Alt + H	Unhide all - pokaż ponownie wszystko
CTRL + H	Restrict selected from render OM – elementy zaznaczone nie będą sie renderować
CTRL + Alt + H	Allow selected to render OM – będą się renderować elementy zaznaczone
I	Insert Keyframe menu – menu wstawiania Klatki Kluczowej
Alt + I	Delete keyframe – usuń klatkę animacji
CTRL + I	Select Inverse – zaznacz odwrotność, czyli jak w programach graficznych 2D
CTRL + J	Join selected objects OM – połącz zaznaczone obiekty
Alt + J	Covert selected triangles to Quads EM – zamień zaznaczone powierzchnie trójkątne na czworokątne. Siatka geometryczna może składać się z trójkątów, czworokątów lub wielokątów. Czworokaty (Quads) są uważane za najlepsze do tworzenia topologii obiektu (schemat budowy geometrycznej), czasem jednak importując obiekt z innego formatu niż .blend dostajemy siatkę trójkątów i trzeba ją przerobić na czworokąty.
LMB + K	Knife cut EM – “Nóż”
CTRL + L	Make Links menu OM – twórz powiązania
M	Move object to a different layer OM – przenieś obiekt do innej warstwy

Skróty klawiaturowe z komentarzem

Alt + M	Merge menu EM – menu Scalania
N	Properties panel toggle – włącz panel Właściwości
CTRL + N	Recalculate normals to outside EM – odwóć powierzchnie na zewnątrz. Chodzi o to, że powierzchnia Face jest dwuwymiarowa i jest skierowana albo na zewnątrz, albo do wnętrza, zależnie od tego jak nią manipulowaliśmy. Czasami, gdy pojawiają się artefakty na powierzni obiektu wystarczy ustawić jeden kierunek dla wszystkich powierzchi
CTRL + Shift + N	Recalculate normals to inside EM - odwóć powierzchnie do wnętrza.
O	Proportional Editing on/off toggle EM – edycja proporcjonalna
CTRL + O	Open file – otwórz plik
P	Start Game Engine OM – włącz silnik do tworzenia gier
P CTRL + P	Separate EM - oddziel Make Parent – twórz “Rodzica”. To jest związane z zarządzanie hierarchią obiektów, możemy kilka obiektów podporządkować pod jeden nadrzędny i tym nadrzędnym manipulować
CTRL + Q	Quit Blender – wyjdź z programu
CTRL + Alt + Q	Quad View toggle – podgląd czworokątów geometrii
R	Rotate - obracaj
Alt + R	Clear rotation OM– resetuj obrót

Skróty klawiaturowe z komentarzem

CTRL + R	Loop Cut EM – linia cięcia, czyli taka linia, którą możemy przeciąć/podzielić powierzchnię zwiększając złożoność jej geometrii
S	Scale – narzędzie do skalowania
Alt + S	Reset Scale OM – resetuje skalowanie
S	Smooth brush SM – pędzel wygładzający
S	Start frame assign (Timeline window) – nadaje początek animacji, ustala klatkę, od której animacja się zacznie
Shift + S	Snap menu – menu Przyciągania (snap – mechaniz “magnesu”)
CTRL + S	Save File – zapisz plik
T	Object Tools panel toggle – pojawi się Panel Obiektu
Shift + T	Flatten/Contrast brush SM – pędzel zwiększający kontrast powierzchni, lub rozmywający ten kontrast
U	UV Mapping menu EM – menu UVM czyli odpowiadające za teksturowanie
CTRL + U	Save User Settings – zapisz ustawienia użytkownika (np zmiany w interfejsie)
CTRL + Alt + U	User Preferences window – okno Ustawienia Użytkownika
V	Object Mode / Vertex Paint Mode toggle – przełącza pomiędzy trybem obiektowym, a trybem malowania
CTRL + V	Vertices menu EM – menu Werteksów

Grafika 3D - Blender

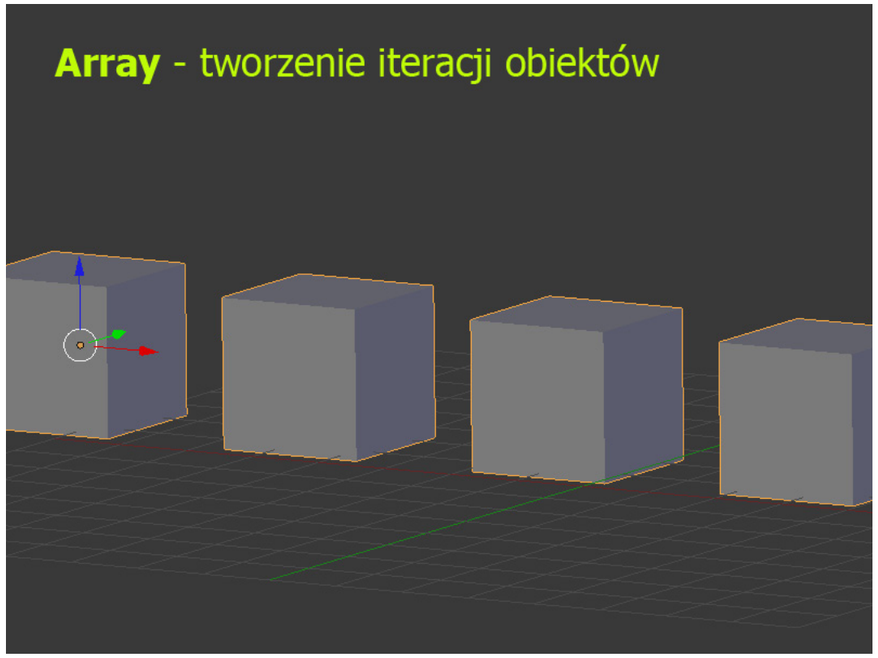
Skróty klawiaturowe z komentarzem

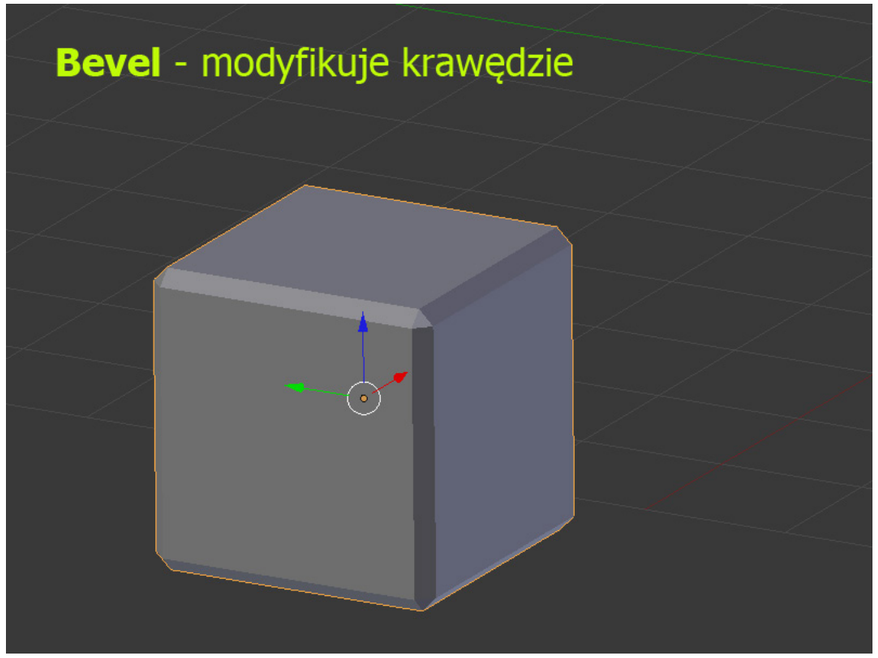
W	Specials menu (varies per object) EM – po prostu przydatne funkcie
Shift + W	Warp EM – zniekształcanie geometrii
X	Delete menu - usuń
Z	Solid / Wireframe toggle – przełączanie pomiędzy widokiem wypelnienia i siatki
Alt + Z	Solid / Textured toggle - przełączanie pomiędzy widokiem wypelnienia i tekstury
CTRL + Z	Undo – cofanie czynności
CTRL + Shift + Z	Redo – ponawianie czynności
NUM 0	Active camera view – widok z kamery, która jest aktywna
CTRL + Alt + NUM 0	Move camera to current view – dostosowuje widok kamery do tego co widać na ekranie
NUM 1	Front view – widok z przodu
NUM 3	Side view - widok z boku
NUM 7	Top view - widok z góry
NUM 5	Perspective/Orthographic view toggle – widok perspektywiczny lub ortogonalny. Widok ortogonalny to taki, gdzie nie ma perspektywy i nawet odległe krańcze obiektu mają takie same proporcje i rozmiary jak te bliskie
CTRL + NUM 1	Back view - widok z tyłu
CTRL + NUM 3	Other side view - widok z boku, ale drugiego

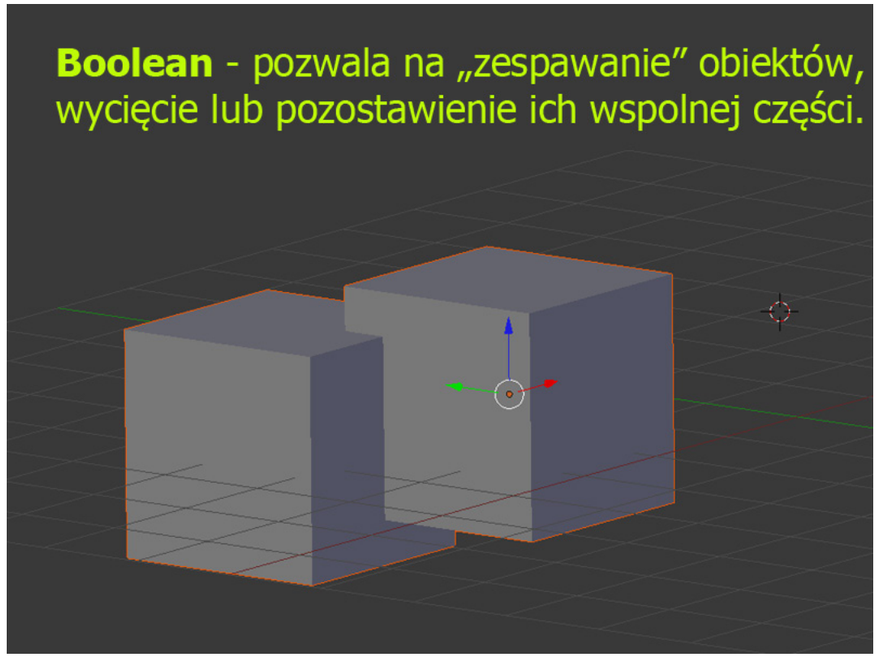
Skróty klawiaturowe z komentarzem

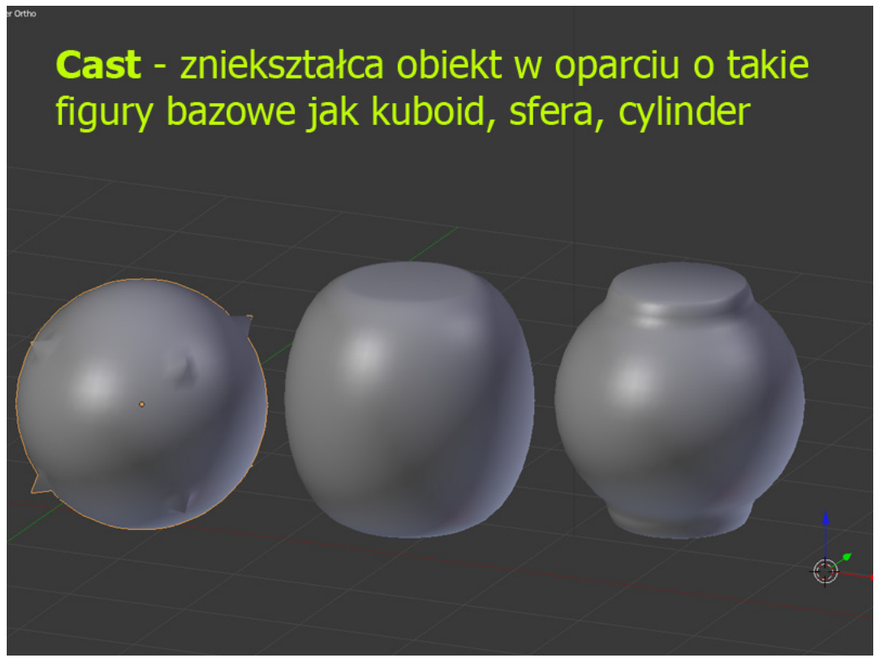
CTRL + NUM 7	Bottom view - widok od spodu
NUM 4/NUM 6	Rotate view left/right in iterations – obraca widok w bok skokowo
NUM 2/NUM 8	Rotate view up/down in iterations – obraca widok góra/dół skokowo
Keyboard #	View layer 1 – 10 – pokazuje penel Warstwy
~	View all layers – pokaż wszystkie warstwy
HOME	Frame all in view – całość widać w oknie, np w wypadku, gdy się pogubimy na scenie, ta funkcja pokaże ją w całości
Tab	Object Mode / Edit Mode toggle – przełącza pomiędzy trybem obiektowym, a trybem edycji
Spacebar	Search (3D view) – szukaj (trzeba mieć kursor w głównym oknie 3D)
Alt + Spacebar	Orientation menu – menu Orientacj
CTRL + LMB drag	Lasso select – zaznaczanie typu lasso
CTRL + LMB click	Extrude / Create new component or bone EM – tworzy element
Shift + Left/Right Arrow	Go to end start/end frame – idź do klatki pierwszej / ostatniej (dotyczy animacji)
Shift + Spacebar	Maximize current view toggle – maksymalizuje widok
F1	Open file – otwórz plik
F2	Save As – zapisz jako

Przykłady deformatorów

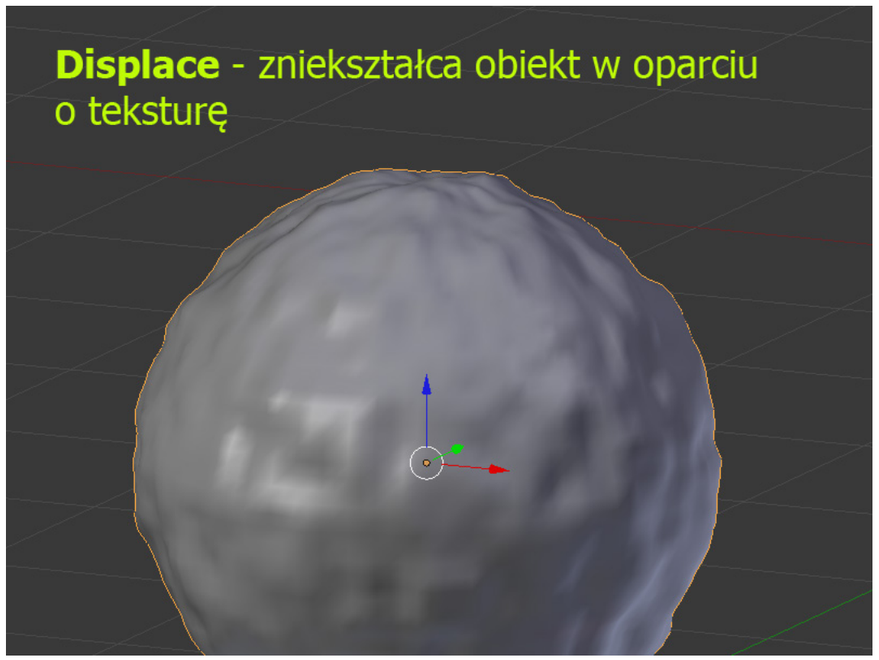


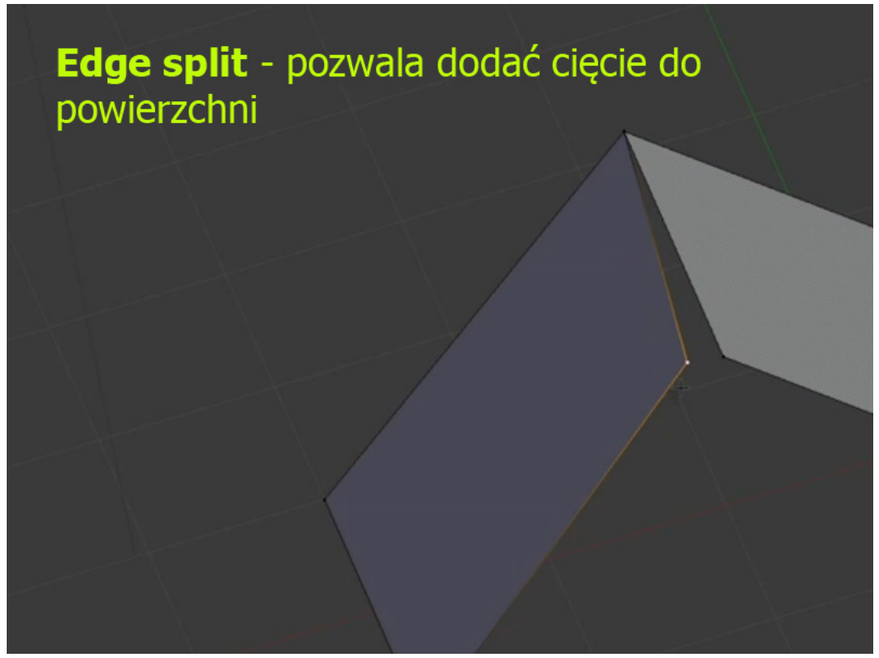


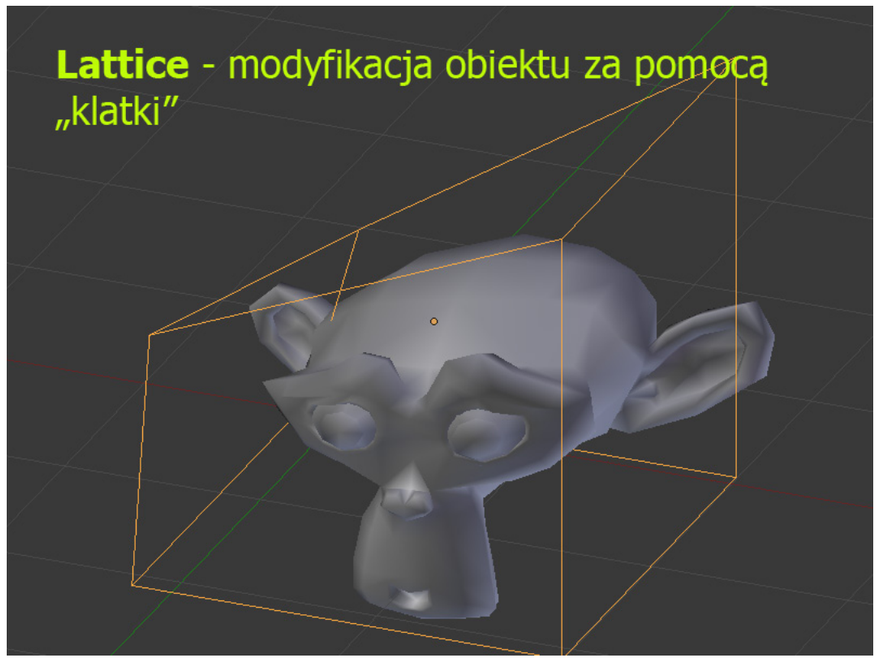




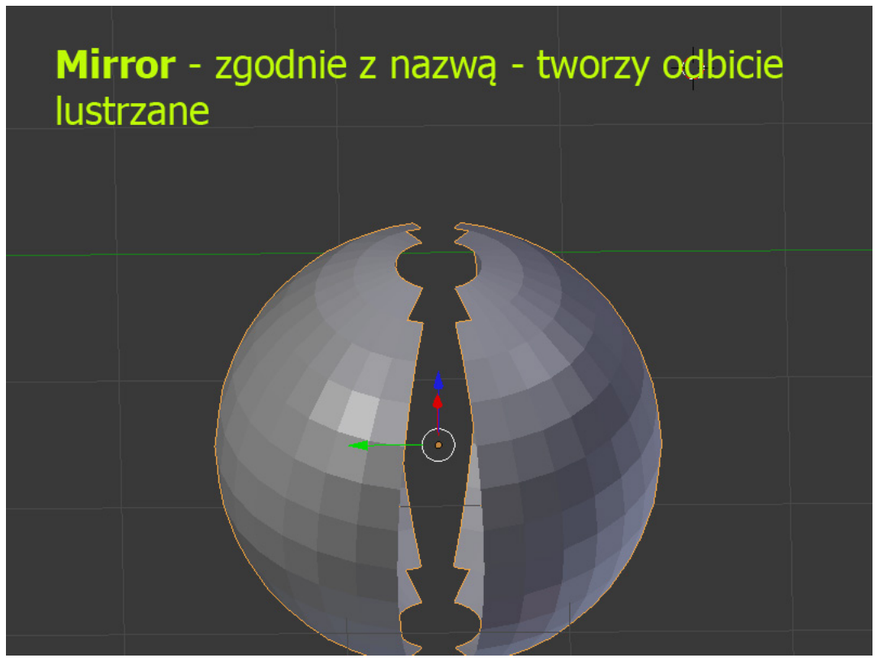
Grafika 3D - Blender
Przykłady deformatorów

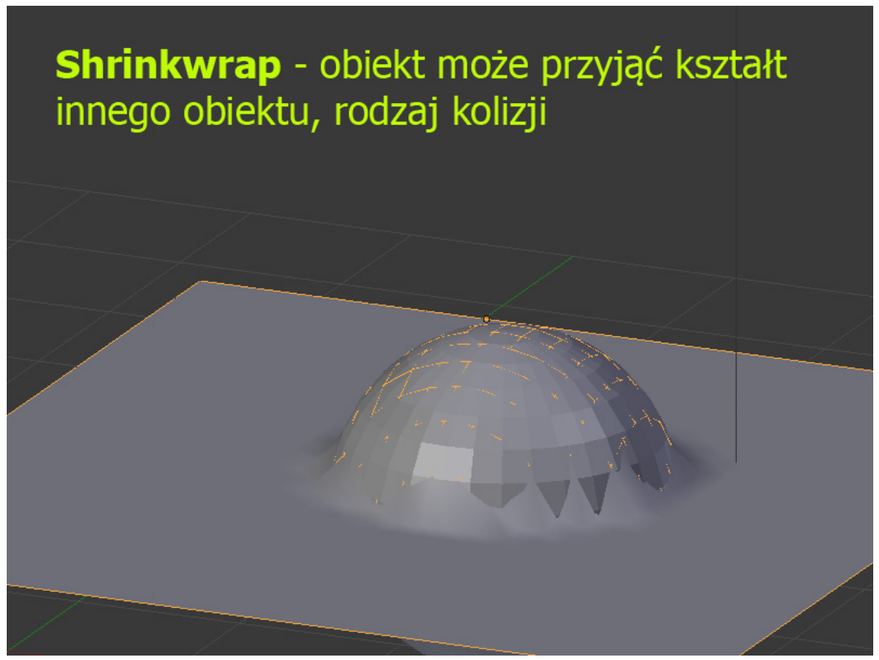




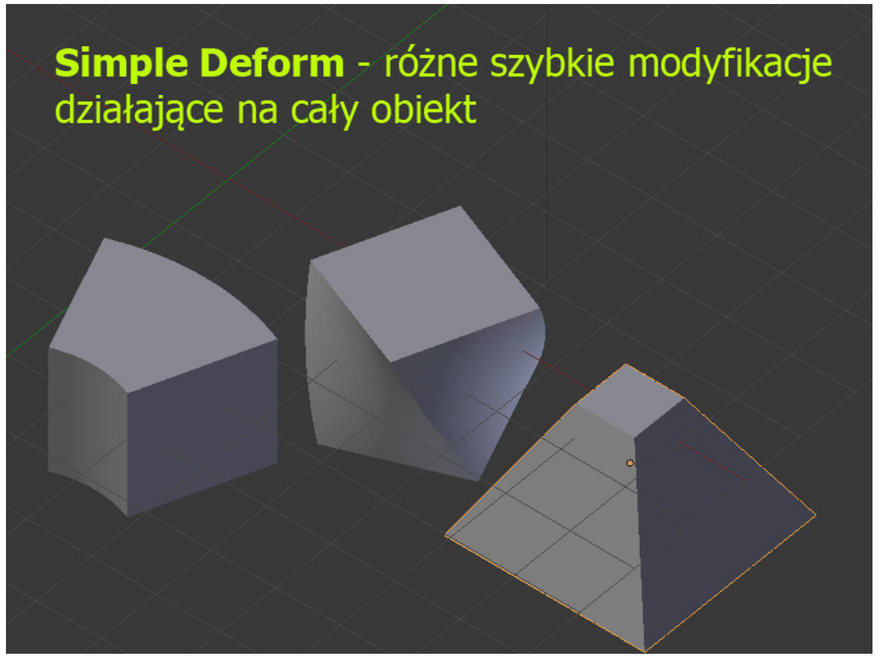


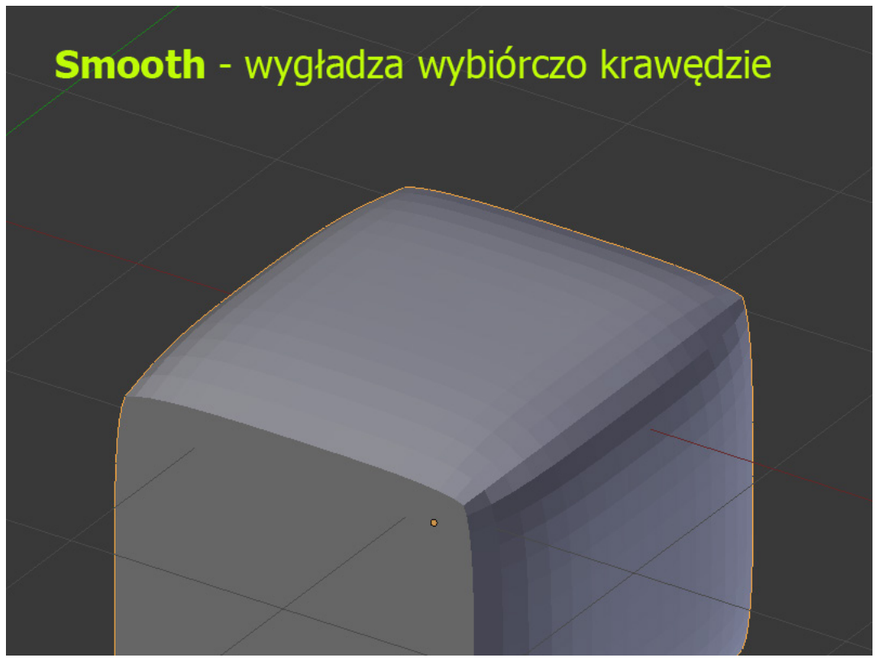
Grafika 3D - Blender
Przykłady deformatorów

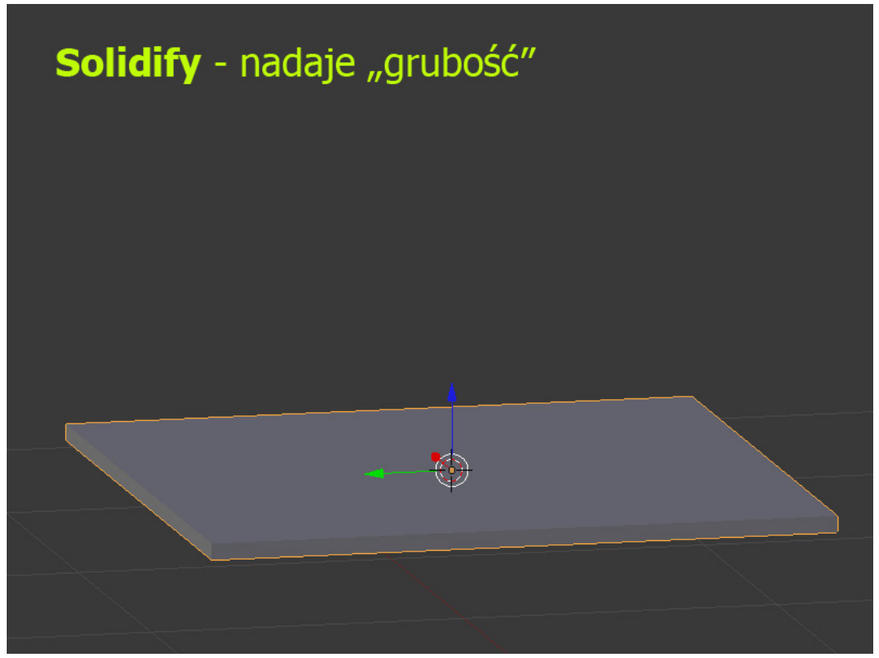


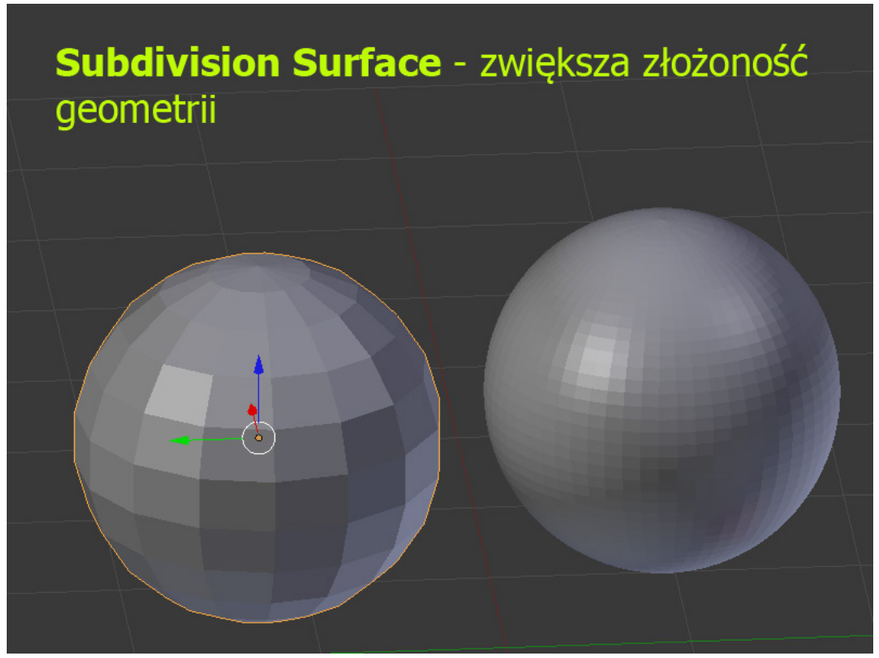


Grafika 3D - Blender
Przykłady deformatorów

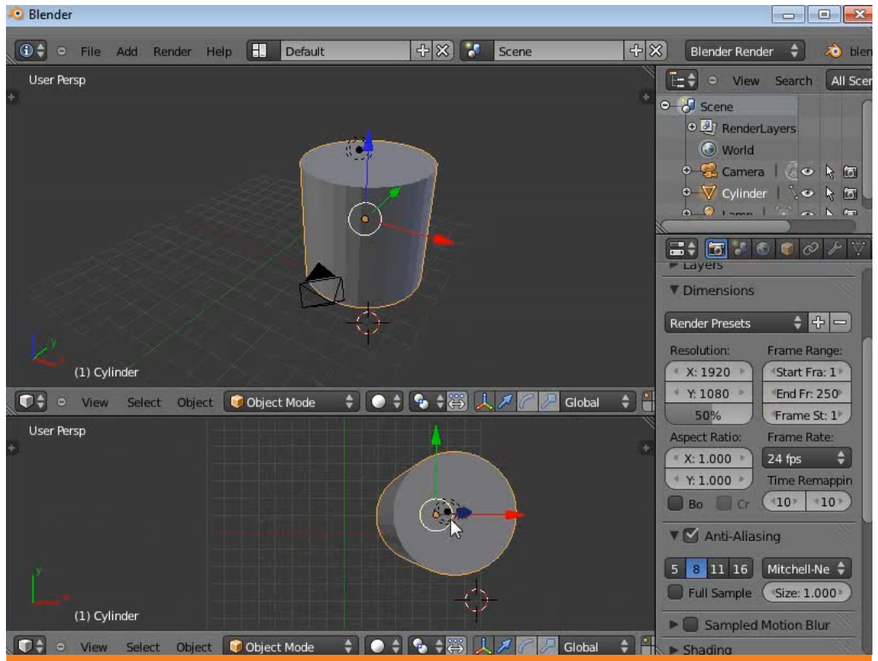




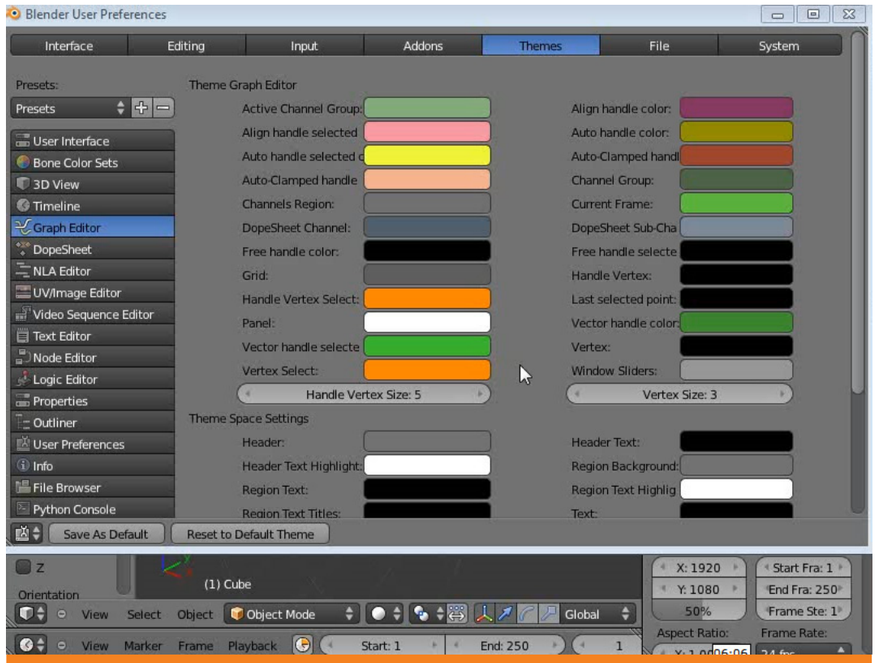




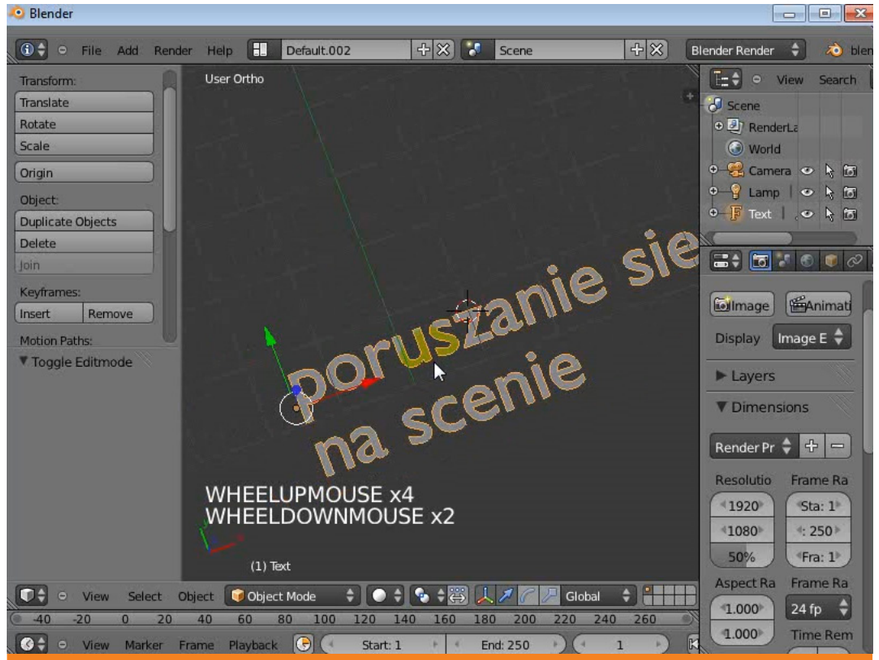
Grafika 3D - Blender
Przykłady deformatorów



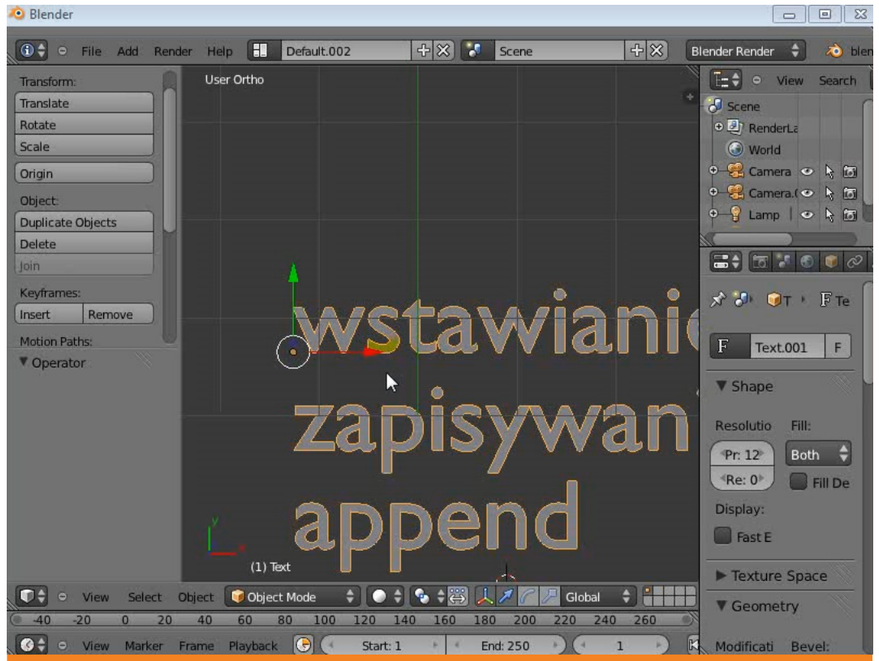
Blender - interfejs



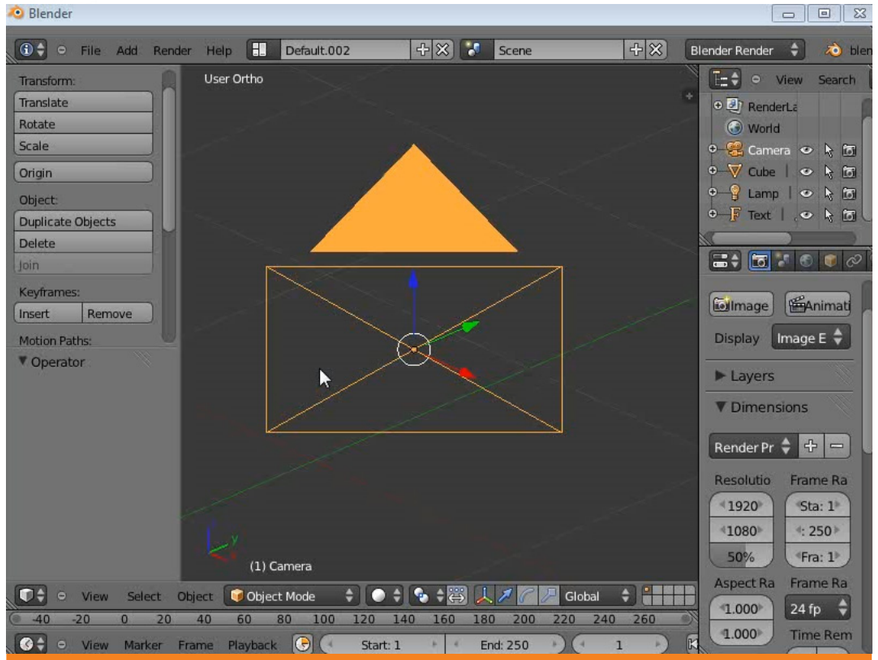
Blender - ustawienia



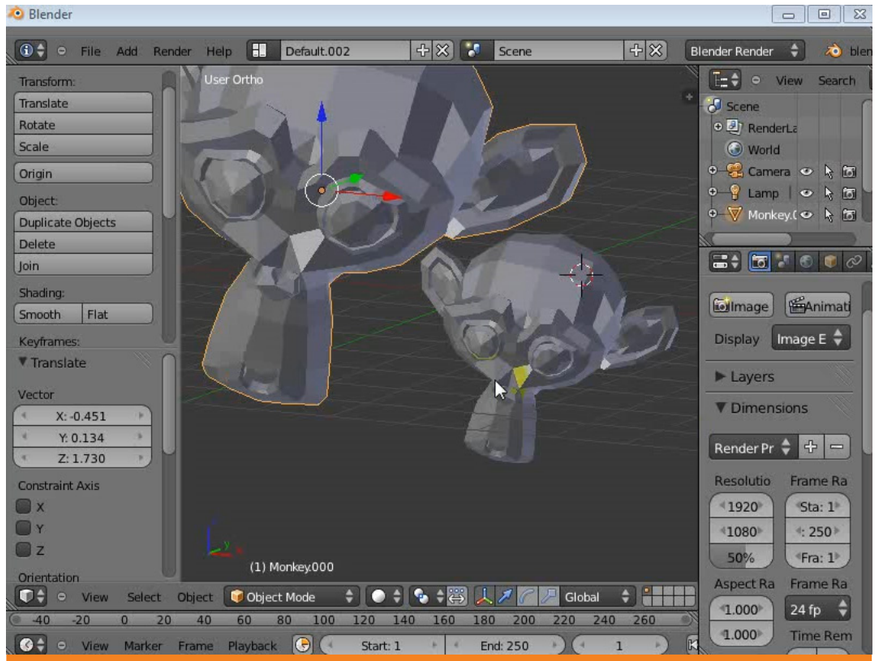
Blender - poruszanie się po scenie



Blender - dodawanie, zapisywanie, append

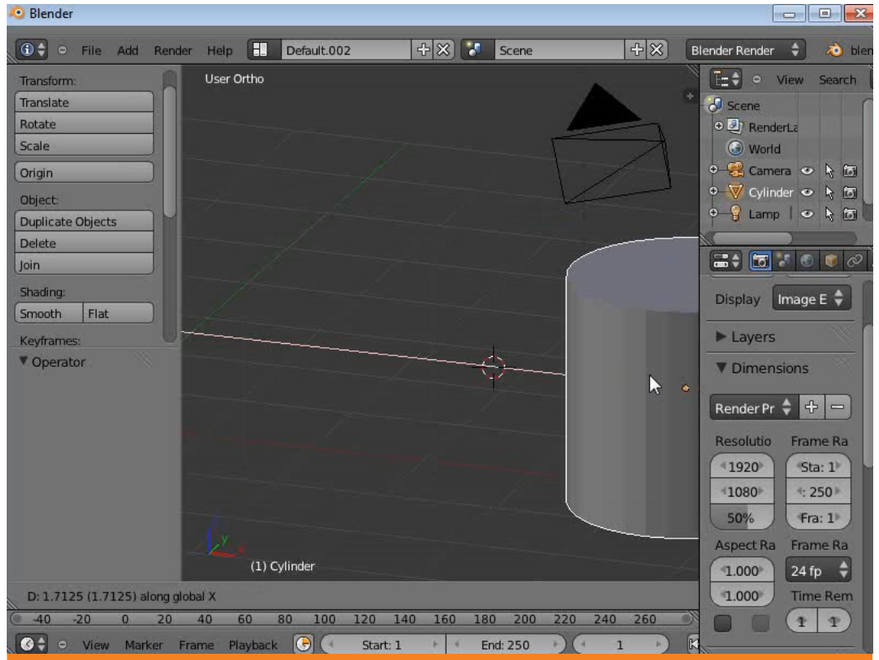


Blender - kamera

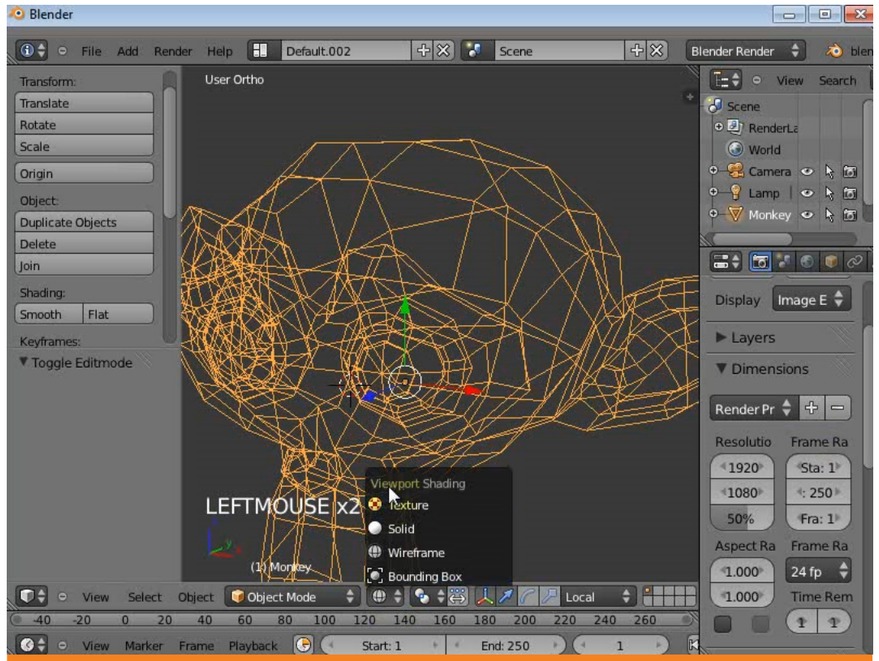


Blender - skalowanie, rotacja

Oś lokalna i globalna, tryby edycji

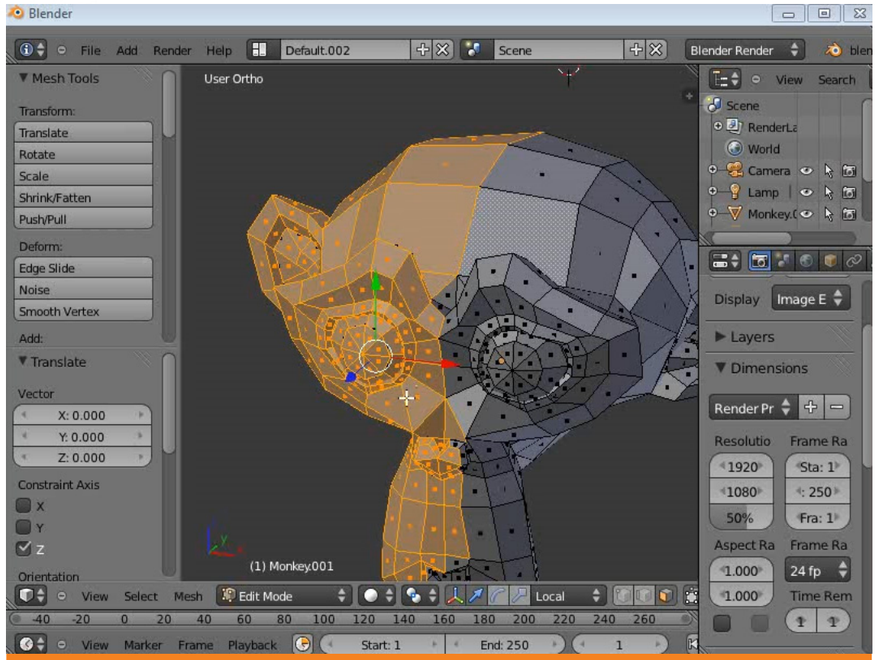


Blender - oś lokalna i globalna

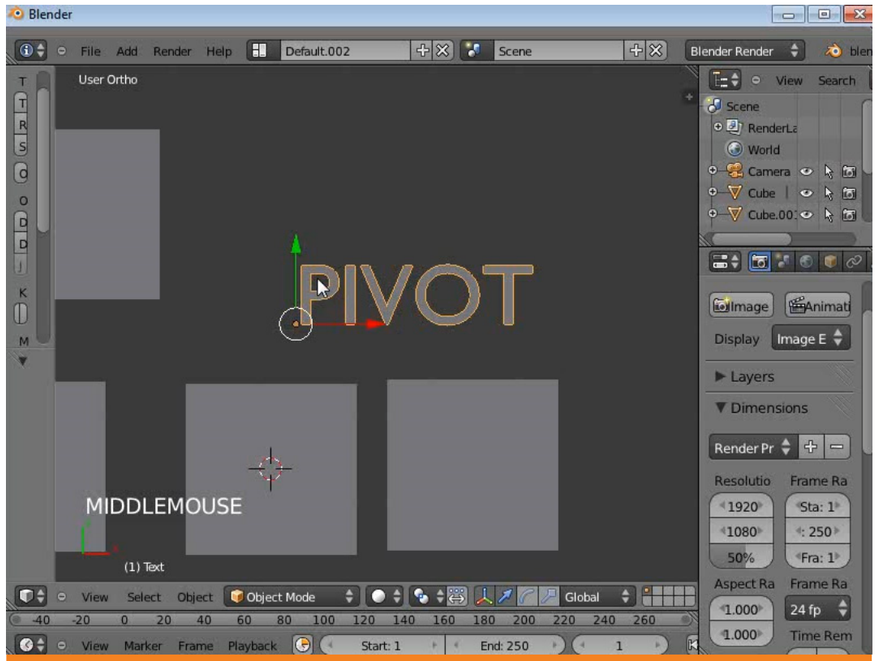


Blender - tryby edycji

Zaznaczanie, piwot

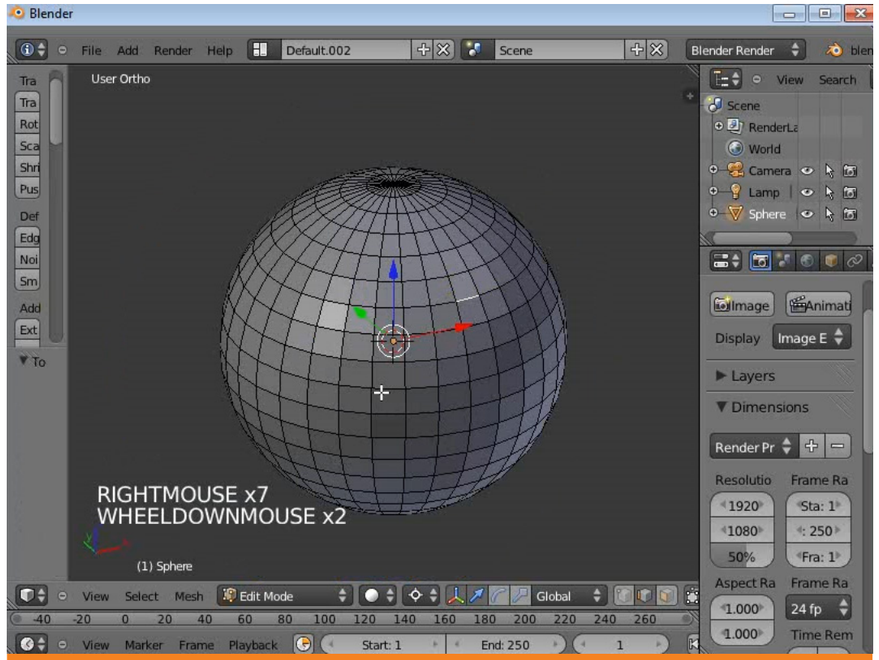


Blender - zaznaczanie

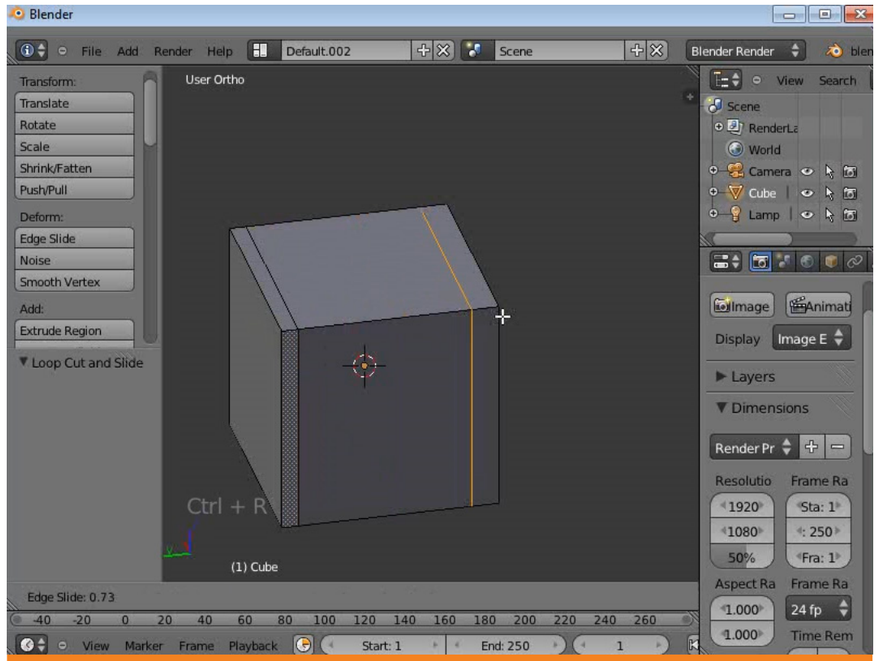


Blender - piwot

Podstawy edycji, extrude, loop

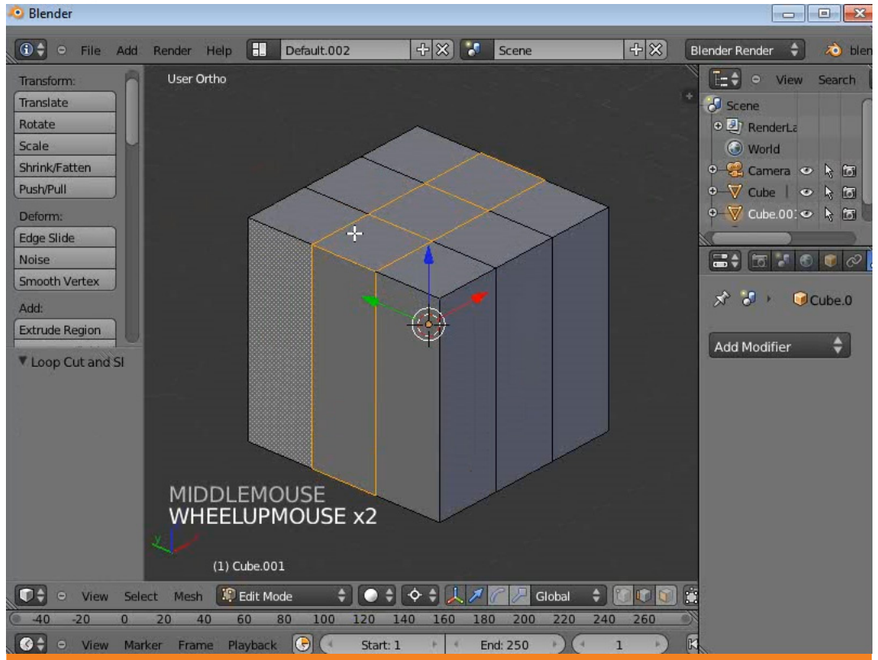


Blender - podstawy edycji

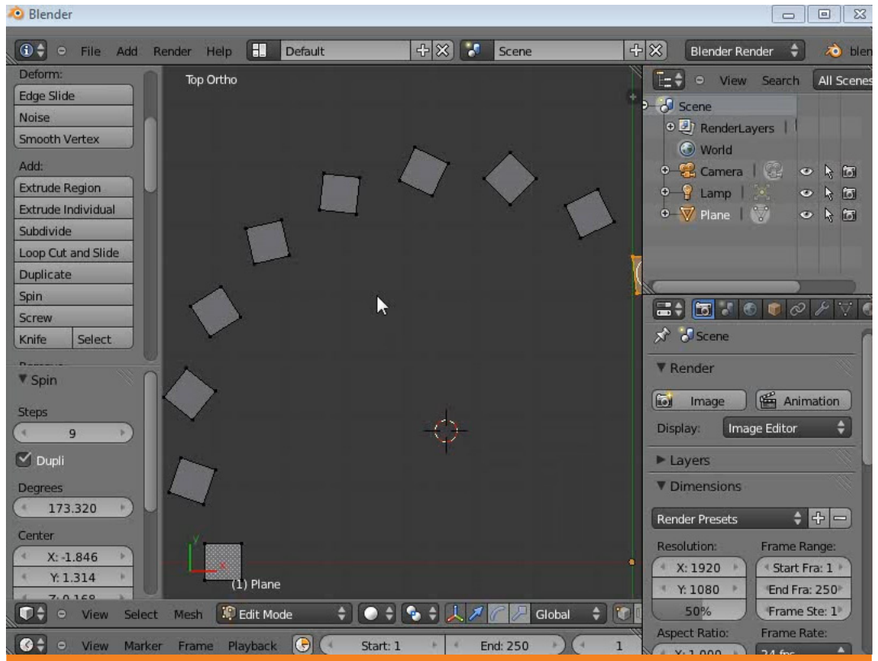


Blender - extrude, loop

Subsurf, spin

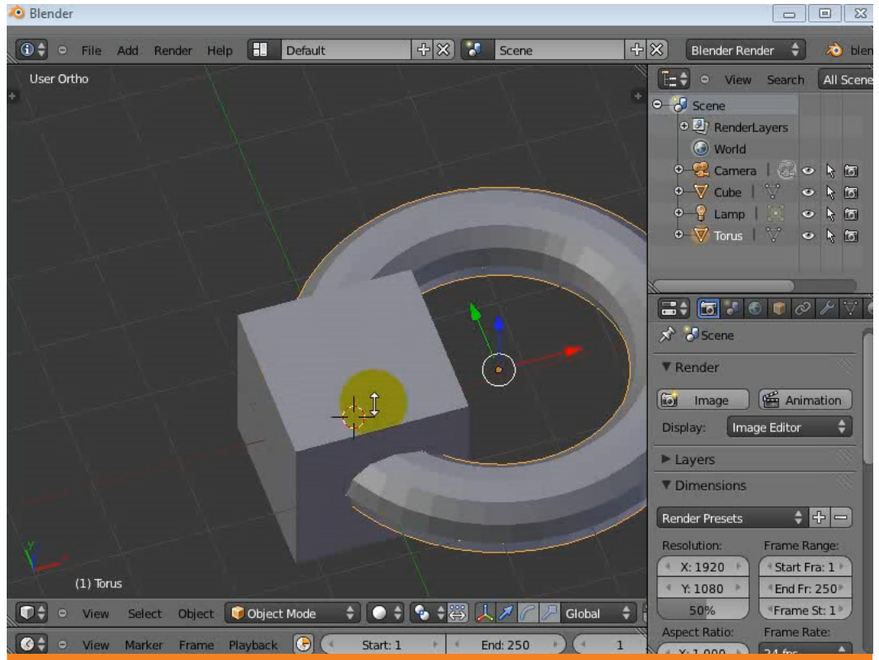


Blender - subsurf

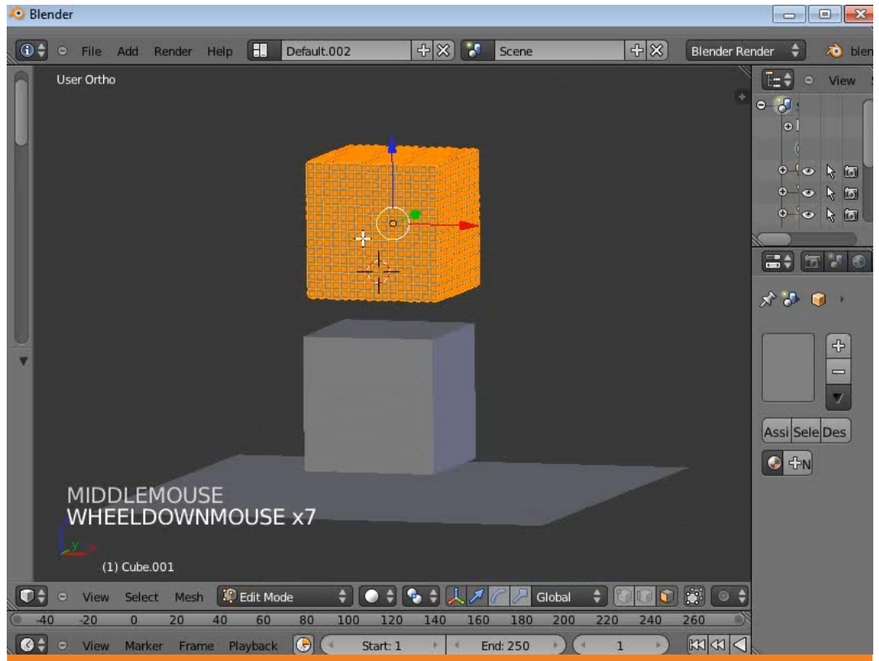


Blender - spin

Boolean, podziały geometrii

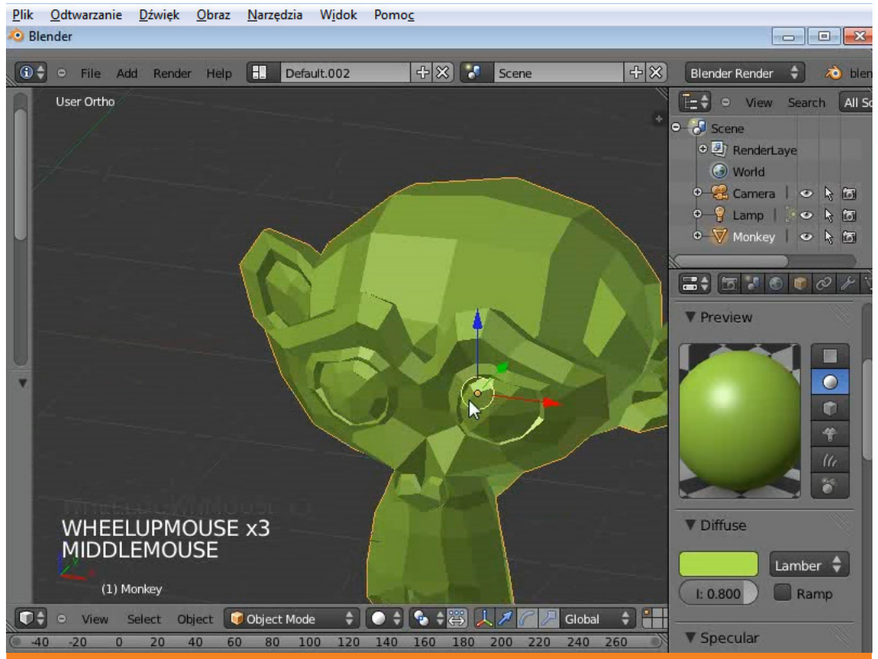


Blender - boolean

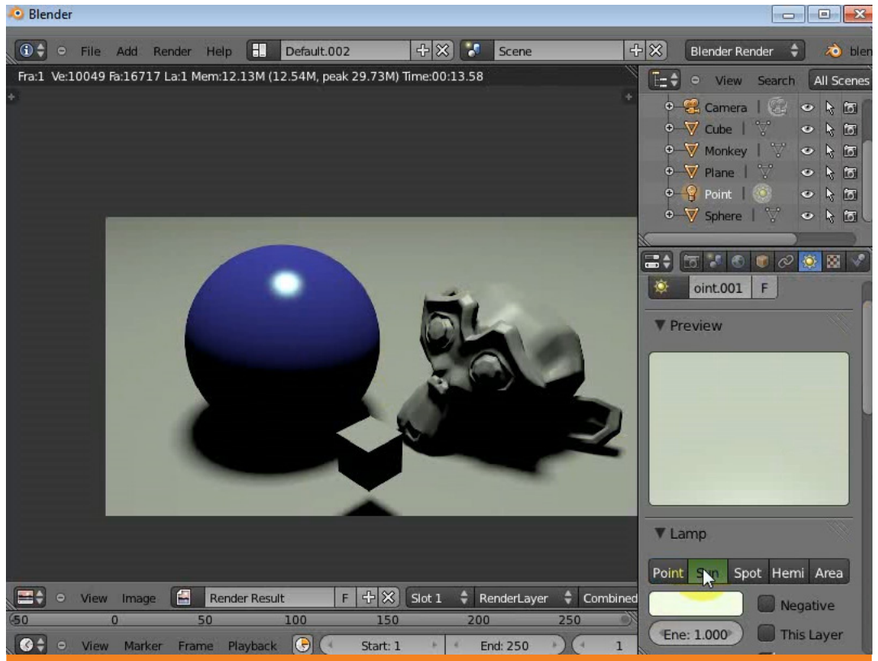


Blender - podziały geometrii

Materiały - podstawy, światło i cień

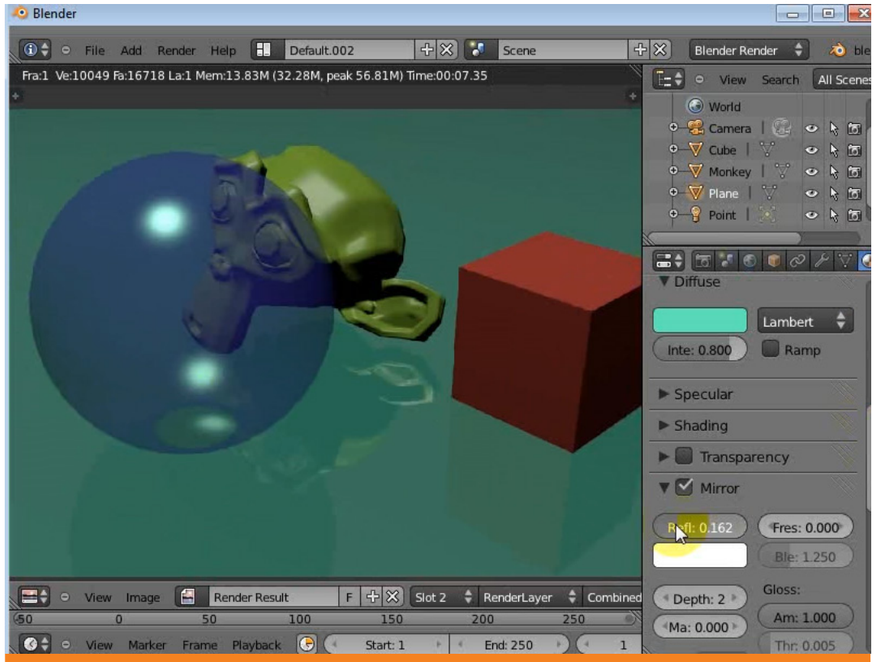


Blender - materiały - podstawy

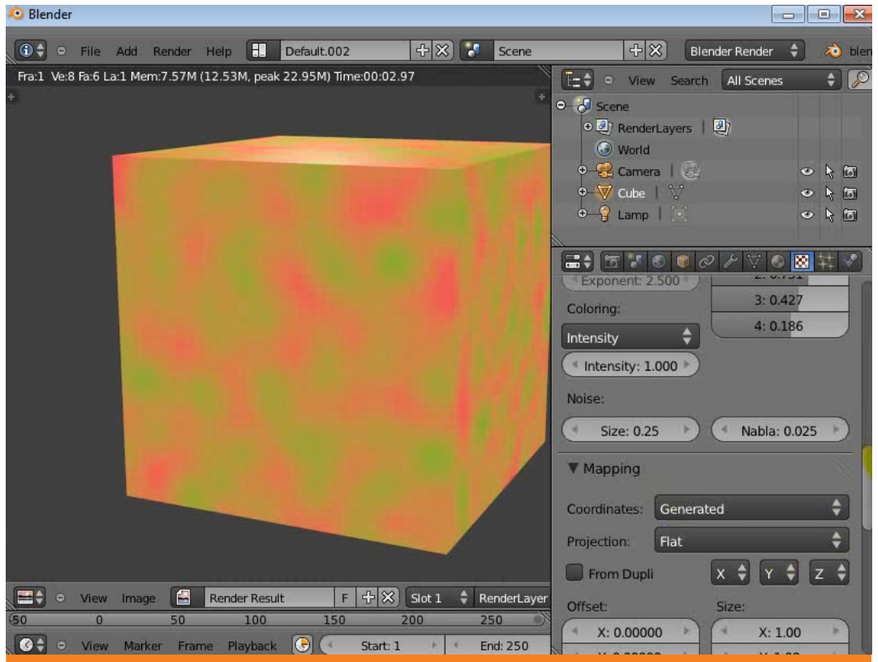


Blender - światło i cień

Materiały - rozwinięcie, tekstury



Blender - materiały - rozwinięcie

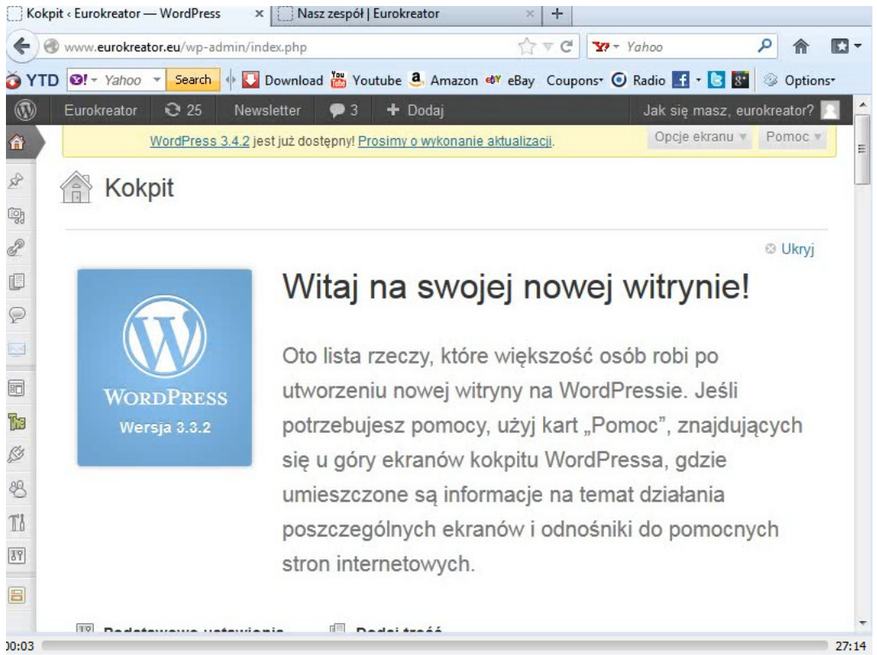


Blender - tekstury

Zasoby

Blender - zasoby

WORDPRESS



Wordpress - Panel Klienta