

POLITECHNIKA ŁÓDZKA
INSTYTUT INFORMATYKI STOSOWANEJ

STUDIA PODYPLOMOWE:
GRAFIKA KOMPUTEROWA I TECHNIKI MULTIMEDIALNE

Kamil Cecherz

**Sposoby budowania interaktywnych stron WWW w technologii
HTML5, z wykorzystaniem narzędzi firmy Adobe.**

**Praca końcowa napisana w Instytucie Informatyki Stosowanej
pod kierunkiem dr inż. Radosław Wajmana**

ŁÓDŹ 2014

Spis treści

Streszczenie	3
Wstęp:.....	5
1. Przygotowanie pola pracy do budowy strony WWW	6
1.1 Wybór technologii, dostosowanie narzędzi	6
1.2 Przygotowanie spisu zasobów dostępnych w Sieci.	9
1.3 Praca off-line z przeglądarkami oparta na drzewie katalogowym.	12
2. Tworzenie układu witryny.	14
2.1 Layout strony a struktura plików HTML.	14
2.2 Sposoby pozycjonowania elementów wpływających na stronie WWW	16
2.3 Jednostki stałe a jednostki relatywne.	17
2.4 Własności pól elementów - dopełnienia, marginesy, ramki.	19
3. Praca nad stylizacją i wzbogacaniem witryny - CSS3 i narzędzia	20
3.1 Otwarta struktura strony	20
3.2 Przygotowanie i dostosowanie zestawu grafik na stronę: wykorzystanie funkcjonalności Photoshop'a i Illustrator'a.	21
3.3 Dopasowanie kolorystyki elementów strony.	30
3.4 Tworzenie menu: formatowanie odnośników, efekty przejścia.....	31
3.5 Praca nad typografią - własność @font-face.	33
3.6 Elementy interaktywne stworzone przy pomocy CSS3: animacje.	36
Podsumowanie	39
• Uwagi	39
BIBLIOGRAFIA.....	41
SPIS ILUSTRACJI	42

Streszczenie

Niniejsze opracowanie opisuje trzy kluczowe etapy tworzenia klasycznej strony WWW: planowanie układu treści witryny, osadzanie jej zawartości w strukturze języka HTML oraz tworzenie wizualizacji przy pomocy grafiki i CSS. Praca oprócz zagadnień czysto praktycznych zawiera odwołania do literatury tematu, opisujących specyfikę wykorzystywanych technologii.

Pozwoliłem sobie na lekkie potraktowanie zagadnień związanych z projektowaniem stron. Praca nie stanowi kompendium w dziedzinie webdesign'u, porusza wiele tematów ale nie stara się ich zamknąć. Posłużyłem się degresjami i porównaniami po to, by zainteresować czytelnika i uświadomić mu jak dużej pracy twórczej wymaga budowanie witryn.

W rozdziale pierwszym udowadniam jak stereotypowe jest nazywanie projektanta stron koderem. W tym celu odwołuje się do zagadnień teoretycznych, próbując zdefiniować, nadać większy sens słowom: webmaster, strona WWW czy narzędzie pracy. Rozdział pierwszy jest próbą odpowiedzi na pytania: jak wygląda mechanizm kreowania witryn, i co rozumiemy pod pojęciem lepsza technologia?

Druga część mojej pracy koncentruje się nad przysłowiowym już *zakodowaniem* treści lub, jak kto woli, przygotowaniem *pustego layoutu*. Pokazuje w niej poprawnie jak napisać strukturę strony, by była gotowa ona do poprawnego wyświetlenia osadzanego tekstu i grafik. Zdradzam wiele technik, które pozwalają uniknąć pułapek w trudnym etapie układania kontenerów, elementów pływających na stronie.

Trzeci rozdział opracowania zawiera odpowiedź na pytanie: jak uczynić stronę bardziej atrakcyjną dla oka, czytelniejszą, efektywniejszą wizualnie i bardziej interaktywną? Zawiera on wiele przykładów wykorzystania eksperymentalnych właściwości CSS3, które w przyszłości stanowią będą, w pełni obsługiwany, standard. W ostatniej części pracy omawiam także funkcje programów Adobe, które pozwalają na efektywną pracę projektanta stron WWW.

W pracy zawarłem mnóstwo przykładów użycia zasobów Sieci: fonty, grafiki, generatory kodu, szablony kolorów, konwertery, ikony. Wszystko po to, by podkreślić jak ważne jest kierowanie się dobrymi wzorcami, tworzenie rzeczy aktualnych i dostosowanych do potrzeb użytkowników.

This paper describes three key steps to create a classic website: planning the content of your site, embedding its content in HTML structure and visualize using graphics and CSS. Work apart from purely practical literature contains references to the subject, describing the specifics of the underlying technologies.

I allowed myself to light treatment of issues related to web design. The work is not a compendium in the field webdesign moves many topics but is not trying to close them. I used digressions and comparisons in order to interest the reader and recognize him as the great creative work requires building sites.

In the first chapter I show how stereotypical to call a web designer coder. For this purpose refers to theoretical attempt to define, to give greater meaning to the words, webmaster, web page, or tool. The first chapter is an attempt to answer the questions: how the mechanism for the creation of websites, and what we mean by better technology?

The second part of my work focuses on the proverbial already coding the content or, if you prefer, prepare a blank layout. It shows how to write properly structured hand, was ready to be deposited to properly display text and graphics. Cheating on a number of techniques that allow you to avoid the pitfalls in the difficult stage of stacking containers, floating elements on the page.

The third chapter of the document contains the answer to the question: how to make the site more attractive to the eye, more readable, more effective visual and more interactive? It contains many examples of the use of experimental CSS3 properties, which in the future will be a fully supported standard. In the last part of the paper I discuss the features of Adobe, which allow you to work effectively designer web pages.

In this paper I made a lot of examples of the use of resources Networks: fonts, graphics, code generators, templates, colors, converters, icons. All this to emphasize how important it is to manage a good practice, making things updated and tailored to the needs of users.

Wstęp:

Podjąłem się stworzenia opracowania opisującego środowisko jak i metody pracy projektanta stron WWW, gdyż chciałem pokazać jak złożonym zagadnieniem może być budowanie dobrych witryn internetowych. W pracy skupiam na aspekcie właściwego wyboru technologii: budowie zaplecza webmastera. Opisuje komercyjne programy Adobe, jak i różnego rodzaju inne aplikacje dostępne w Sieci, które czynią pracę webdesigner'a bardziej elastyczną i szybszą. Celem mojego opracowania jest przedstawienie najbardziej efektywnych metod, takich, które dynamizują pracę projektanta.

Zakres mojej pracy jest dość szeroki. Opisuje cały proces powstania strony uwzględniając jej zgodność ze specyfikacją HTML5 i CSS3. Prezentuje rozwiązania pozwalające stworzyć jej czytelny i łatwy w modyfikacji układ. Opisuje wieloetapową pracę nad tworzeniem struktury i stylizacją strony na poziomie kodowania. Przedstawiam narzędzia i zasoby: począwszy od komercyjnych programów graficznych, a skończywszy na bezpłatnych generatorach kodu, konwerterach i bibliotekach grafik oraz fontów.

Chcąc lepiej wdrożyć czytelnika, nie starałem się opisywać tworzenia strony, jako żmudnego automatycznego procesu. Ludzie budujący witryny internetowe to osoby, które podczas swojej pracy muszą wykazywać się dużą kreatywnością, a jednocześnie działać wydajnie stąd w moim opracowaniu tyle namysłu nad samą istotą specyfiką Sieci.

Opracowanie stawia czytelnika przed wieloma wyborami, nie tylko pokazuje jak stworzyć stronę, ale także skłania do przemyśleń. Technologia tak szybko się zmienia, że prawdopodobnie wiele z opisanych przede mną rzeczy zdezaktualizuje się: warto, więc wybierać przyszłościowe modele pracy - nie zapominając o źródle.

Celem mojej pracy jest przedstawienie etapów tworzenia statycznej strony WWW. Projekt, który prezentuje wykorzystuje najbardziej podstawowe technologie webowe - HTML5 (połączenie funkcjonalności XHTML i możliwości JavaScriptu) oraz CSS3. Zagadnienia ściśle programistyczne ustępują w moim opracowaniu miejsca problematyce związanej z pracą webdesignera. Opowiadam o efektywnym wykorzystywaniu profesjonalnych narzędzi, bez których projektowanie stron internetowych byłoby trudne. Przytaczam też parę istotnych uwag i wskazówek pozwalających unikać pułapek początkującym grafikom myślących o wykorzystaniu swych zdolności w tworzeniu stron WWW.

1. Przygotowanie pola pracy do budowy strony WWW

1.1 Wybór technologii, dostosowanie narzędzi

Rok 2014. Web 3.0 jako etap ewolucji Internetu. Kształtujący się model Sieci, jako medium silnie immersyjnego. Internet o krok od wejścia w erę oprogramowania pozwalającego wizualizować oraz przetwarzać dane w trzech wymiarach. Okres coraz większego popytu na urządzenia mobilne. Dekada nadciągających chmur, Internetu bez kabli, Internetu o wysokiej przepustowości - Sieci, która ogarnie cały świat, jeszcze bardziej przeniknie do życia człowieka.

To nie powieść futurystyczna, ani nawet nie forma eseju. Warto jednak zdać sobie sprawę w obliczu, jakich zmian stoimy. Skupimy się na aspektach technicznych pracy webmastera, nim jednak to zrobimy musimy poprawnie zdefiniować cele:

Według Tima O'Reilly Web 2.0 „jest rewolucją biznesową w świecie komputerowym, spowodowaną ruchem w stronę Internetu jako platformy, oraz próbą zrozumienia reguł zwycięstwa na tej platformie. Główną zasadą jest: Twórz aplikacje, które lepiej okiełznają Sieć, by więcej ludzi ich używało” [1]



Ryc. nr [1] - Chmura znaczników Web 2.0 opracowana na podstawie artykułu Tima O'Reilly „What is Web 2.0”

Zbudowanie dobrej witryny wymaga postawienia się w roli menadżera, marketingowca jak i architekta. Webmaster sam dla siebie tworzy reguły logistyczne: uczy się gospodarować czasem tak, by działać szybko i wydajnie; w umiejętny sposób zarządzać zasobami. Może się to niektórym wydać nieco przewrotne, ale moim zdaniem nie umiejętność rozwiązywania problemów otwiera największe możliwości; największe możliwości daje zdolność wykorzystywania, odnajdywania gotowych rozwiązań. O co w tym wszystkim chodzi?

WWW jest niczym innym jak jedną z trzech głównych usług, które oferuje nam Internet, obok poczty elektronicznej i P2P. U podstaw filozofii Sieci stoi stwierdzenie - zdecentralizowana, wolna i dostępna dla wszystkich. O ile z drugim i trzecim określeniem możemy się zgodzić, to pierwsze z nich rodzi wątpliwości. W świetle idee Internetu szybko wdarły się prawa ekonomii, jak celnie zauważył O'Reilly, mamy do czynienia z ruchem; ruchem, który trzeba ukierunkować; Siecią, którą należy okiełznać. Nie da się zatrzymać nurtu rzeki rękoma, jak i nie da się zarządzać tworem tak nieokiełznanym i dynamicznym jak Internet bez odpowiednich narzędzi, technik, rozwiązań.

Wybór technologii jaką się posłużymy chcąc wygenerować naszą witrynę powinien opierać się na kilku podstawowych zasadach, odpowiadającym ówczesnym trendom. Internet to medium, środek komunikacji - ale! o czym warto pamiętać nie jednostronnej - strona WWW to też narzędzie. Oddajemy w ręce użytkownika zaprojektowany przez nas mechanizm, tylko od nas zależy do czego posłuży.

Praca nad stroną przebiega w trzech etapach, które się ze sobą łączą:

- **Osadzenie treści w kodzie** (tworzenie semantyki strony, zamykanie wszystkich informacji tekstowych jak i graficznych w znaczniki, logiczny i hierarchiczny układ).
- **Tworzenie i stylizowanie layoutu** (nadawanie przejrzystego wyglądu elementom osadzonym w HTML-u, uzupełnianie wizualizacji o dodatkowe interaktywne komponenty, układanie elementów i pozycjonowanie segmentów na stronie).
- **Przygotowywanie strony do współpracy z użytkownikiem** (minimalizacja błędów występujących w przeglądarkach, dostosowanie strony do wyświetlenia pod różnymi rozdzielczościami, obsługa formularzy, dostosowanie hiperłączy).

Trzy wymienione kroki należy wykonywać po kolei, choć zaawansowane metody planowania (których nie opiszę w niniejszej pracy) zakładają zespół działań, sprowadzających się do tworzenia prototypu strony - polega to na: wykonaniu układu strony w formie szkicu, po czym przeniesienie go do programu graficznego z zachowaniem odpowiedniej skali. Opisane postępowanie skraca czas pracy ale wymaga pewnego doświadczenia i umiejętności.

Przygotowanie strony WWW tak naprawdę nie wymaga żadnych drogich programów. Sam HTML5 już stanowi narzędzie, dostępne dla wszystkich i nie wymagające żadnych opłat. Szybko jednak przekonamy się, że wykonanie strony, nawet w tak prostych technologiach jak HTML5 i CSS3 wymaga dużo wysiłku z racji ograniczeń wynikających właśnie z tej prostoty. Bez znajomości języków programowania częściej zdani jesteśmy na rozwiązania ogólne - nie bez powodu szybko zrodziło się pojęcie webdeveloper (odnoszące się do osoby nie będącej ściśle grafikiem acz programistą posługującym się językami skryptowymi).

By bardziej rozjaśnić kwestię aplikacji, które będą nam pomocne w generowaniu strony WWW podzielę je na grupy:

- **Proste edytory kodu** (pozwalają na efektywne zarządzanie plikami, stosują kolorowanie składni kodu i wcięcia, dzięki jest on łatwiejszy w odczytaniu i co za tym idzie edycji, jednym z najpopularniejszych jest SublimeText 3).
- **Edytory WYSIWIG** (następcy edytorów prostych, programy te posiadają możliwości tworzenia stron z gotowych szablonów, pozwalają na korzystanie z bibliotek komponentów, posiadają liczne skróty i ułatwienia, umożliwiają „rysowanie” układu strony, w czołówce znajduje się Dreamweaver Adobe).
- **Programy graficzne** (dzięki programom graficznym możemy przycinać, wymiarować, optymalizować, stosować filtry do ilustracji i zdjęć; możemy składać lub tworzyć własne grafiki jak i projektować całe layouty, w czołówce znajdują się produkty Adobe: Photoshop, Illustrator, InDesign).
- **Przeglądarki** (pozwalają na przetestowanie stron i dokładne ich przejrzanie dzięki wbudowanym narzędziom do inspekcji kodu).

Przed przystąpieniem do pracy należy zoptymalizować wszystkie niezbędne nam programy tak, by nie wymagały późniejszej konfiguracji a ich obsługa była intuicyjna.

Podczas tworzenia projektu korzystałem z narzędzi bezpłatnych (głównie do edycji kodu źródłowego strony) jak tych komercyjnych znacznie bardziej rozbudowanych, niezbędnych przy pracy każdemu grafikowi komputerowemu i webdesigner'owi. Moja praca koncentruje się na metodach i sposobach budowania stron WWW, więc opisze w niej tylko wybrane funkcje narzędzi Adobe: Photoshopa, Illustratora i Dreamweavera, te które są najbardziej pomocne w automatyzacji pracy projektantów stron WWW.

1.2 Przygotowanie spisu zasobów dostępnych w Sieci.

Grzechem podczas tworzenia własnej witryny byłoby nie skorzystanie z zasobów Internetu. W tym miejscu odnoszę się do wcześniejszego stwierdzenia, zwracającego uwagę na umiejętność poszukiwania i wdrażania gotowych rozwiązań. Projektanci stron WWW nie starają się tworzyć niezależnych solucji, często zaś posługują się sprawdzonymi metodami. Nie oznacza to brak kreatywności, wręcz odwrotnie. Sieć działa specyficznie pozwala łączyć umysły, tworzyć społeczności; każdy jej wytwór w momencie jego ulokowania on-line staje się pomysłem wszystkich, jest ogólnodostępny.

Zagadnienie zasobów sieci ogólnodostępnych; pod pojęciem których należy rozumieć różnego rodzaju aplikacje obsługiwane z poziomu stron WWW, systemy CMS z otwartym kodem źródłowym, dane w postaci tekstowej jak i graficznej na stronach internetowych; jest na tyle szerokie by poświęcić mu osobą publikację. My jednak skupimy na małym wycinku.

Mój projekt, choć jest tworem niezależnym, stworzonym od podstaw, został oparty na tradycyjnych rozwiązaniach. Wyszukiwarka Google podczas jego tworzenia służyła mi jako druga ręka. Dzięki ogromnemu wsparciu ze strony społeczności w Sieci, mogłem spokojnie skupić się na projektowaniu odciążając przy tym własną pamięć od dokładnej nauki całej specyfikacji CSS3 i HTML5. Nie musiałem szukać pomocy dotyczącej obsługi programów graficznych, gdyż wcześniej opanowałem je biegle ale jestem pewien, że i w tej materii społeczności internetowe oferują wiele.

Przed przystąpieniem do jakiegokolwiek pracy należy w kolejności:

- Zrobić spis adresów internetowych, z których będziemy pobierać grafikę (przypominam o prawach autorskich):
 - w moim przypadku to:
 - www.flickr.com;
 - www.iconfinder.com;
 - www.google.com/imghp?hl=pl
- Sporządzić zestawienie linków zewnętrznych (mogą odnosić się do stron, animacji, grafik, artykułów):
- Wykonać listę stron, artykułów on-line, skryptów z uwzględnieniem ich autorów, pomocnych nam w wykonaniu strony:
 - w moim przypadku to:
 - <http://kodcss.pl>;
 - <http://animateyourhtml5.appspot.com/pres/#1>;
 - <http://kurshtml5.edu.pl/animacje-css/>;
 - <http://designconcept.webdev20.pl/articles/formularze-w-html5/>;
 - <http://grafmag.pl/artykuly/formularze-w-html5/>;
 - <http://html-generator.weebly.com/css-textbox-style.html>;
 - <http://www.kurshtml.edu.pl>;
 - <http://www.w3schools.com/>;
- Przygotować wykaz narzędzi dostępnych z poziomu stron WWW:
 - w moim przypadku to:
 - <http://www.fontsquirrel.com/fontface/generator>;
 - <http://fonty.pl/kolory.htm#generator>;
 - <http://www.colorzilla.com/gradient-editor/>;

Użyj kursora oraz pionowego paska do wyboru koloru i skopiuj kod z ramki (z krzyżykiem):

Lub rozpocznij od koloru: #ABCDEF OK

CSS 3D: animations, transitions and transforms can be combined

```

1 @-webkit-keyframes przyklad {
2   from {
3     left: 10px;
4   }
5   to {
6     left: 100px;
7   }
8 }
9 @keyframes przyklad {
10  from {
11    left: 10px;
12  }
13  to {
14    left: 100px;
15  }
16 }
17
18 #animation {
19   width: 150px;
20   height: 100px;
21   background-color: #f00;
22   margin: 10px 0 20px 0;
23   position: relative;
24   -webkit-animation: przyklad 2s infinite alternate;
25   animation: przyklad 2s infinite alternate;
26 }

```

Pierwszy element w 25 linii kodu służy do wprowadzenia nazwy (animation-name) reguły @keyframes, która będzie wykorzystana do animacji elementu.

Drugi element w 25 linii kodu służy do wprowadzenia czasu trwania jednego przebiegu animacji (animation-duration).

Animacje z zastosowaniem 2 różnych czasów:

Ryc. nr [2] - strony z przepisami i narzędziami dla webmasterów

Nie mam miejsca w mojej pracy, aby opisać wszystkie przepisy i rodzaje narzędzi, które odnalazłem ale mogę powiedzieć, że bardzo mi one pomogły w tworzeniu ciekawych i zaskakujących efektów na stronie bez wykorzystania stacjonarnych aplikacji.

1.3 Praca off-line z przeglądarkami oparta na drzewie katalogowym.

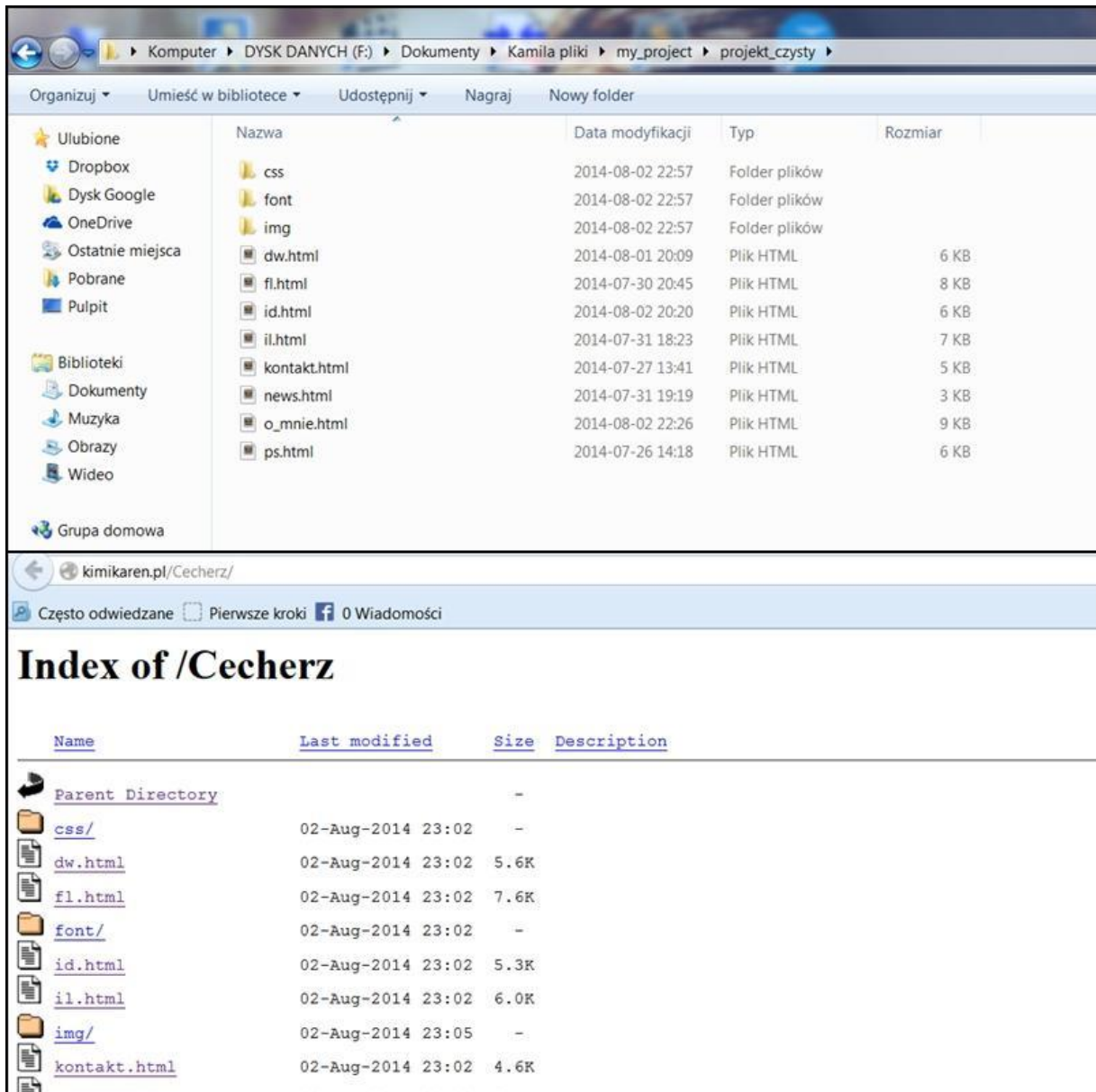
Nasza strona, by mogła stać się gotowa do życia w Sieci, musi zostać poprawnie skonstruowana lokalnie. Zadanie to, z pozoru banalne, bo polegające na utworzeniu, nazwaniu i połączeniu kilkunastu plików oraz ich segregacji w katalogach, często sprawia problemy początkującym projektantom stron WWW.

U podstawy każdej strony WWW leżą dwa pojęcia hipertekst i hiperłącze. Struktura witryny internetowej:

[...]to nielinearna i niesekwencyjna organizacja danych - tekst rozbity na fragmenty, które na wiele sposobów połączone są ze sobą odsyłaczami. To tekst, który rozgałęzia się lub działa na żądanie czytelnika. [\[2\]](#)

Użytkownicy naszej witryny nie muszą znać ani jej układu katalogowego, ani nazw poszczególnych plików, ich interesuje jedynie łatwo dostępna treść. My jednak powinniśmy pozostać schematyczni i uporządkowani, tworzyć stronę tak, by wiedzieć co gdzie się znajduje i jak zostało nazwane (w tym miejscu warto zwrócić na rozszerzenia plików o tej samej nazwie, gdyż komputer uznaje je za różne). O ile praca nad kilkoma plikami nie sprawia problemu to aktualizacja tysiąca może już być kłopotliwa bez odpowiedniej segregacji.

Przystępując do pracy na stronę warto utworzyć dwa podstawowe pliki, które będą stanowiły jej trzon: *index.html* (który przeglądarka rozpozna jako pierwszą ze stron do otwarcia, po wpisaniu adresu domeny) , *jakaś_nazwa.css* (nazwa pliku css nie jest istotna) oraz kilka podstawowych katalogów osobno na: grafikę, skrypty, fonty, dokumenty, pliki.swf, filmy, muzykę. Nie ma jednej prawidłowej zasady jak prawidłowo układać pliki, warto jednak jakąś jasną dla Nas regułę sobie przyjąć.



Ryc. nr [3] - układ plików i katalogów na stronie mojego projektu (brak pliku o nazwie index.html powoduje wyświetlenie się drzewa katalogów także on-line, dopiero podanie adresu np. kimikaren.pl/Cecherz/dw.html odsyła do widoku strony).

Błąd zaprezentowany na Ryc. nr. [3] może wprowadzić internautów oglądających naszą stronę w konsternację, dlatego należy go unikać.

2. Tworzenie układu witryny.

2.1 Layout strony a struktura plików HTML.

Nie jest łatwo budować stronę WWW *cegielka po cegielce*. Wprowadza to mnóstwo chaosu, bałaganu i trwoni wysiłek projektanta. Układ witryny (nie mylić z wyglądem) należy rozplanować przed przystąpieniem do pracy. Jak to jednak zrobić poprawnie?

Budowa witryny internetowej przypomina planowanie układu pokoi w mieszkaniu. Wybieramy sobie pewną powierzchnię do podzielenia na mniejsze pomieszczenia. Pokoje, z których każdy będzie miał swą odrębną funkcję i różnił się rozmiarem, oddzielamy od siebie ścianami nie zapominając o ich połączeniu - stwarzamy możliwość wejścia i wyjścia poprzez drzwi, zakładamy okna. Na razie nie martwimy o wygląd ścian pomalujemy je potem, nie zapomnimy też o powiadomieniu ekipy remontowej o transporcie mebli.

Strona internetowa to nic innego jak pewien zasób danych spakowanych w jednostki: pudełka, podzbiory. Stąd tak często opisując generowanie stron przy pomocy HTML mówimy o osadzaniu, zagnieżdżaniu.

Etap planowania witryny, którą prezentuje w projekcie, zajął mi najwięcej czasu. Z początku chciałem narysować układ kontenerów strony z uwzględnieniem skali w Adobe Illustratorze, co okazało się zadaniem karkołomnym, gdyż bardziej skupiałem na przejrzystości i kolorystyce projektu niż jego funkcjonalności. Po etapie frustracji postanowiłem zrezygnować ze stylistycznej warstwy szkicu strony i stworzyłem schemat w arkuszu kalkulacyjnym, choć równie dobrze mogłem użyć zwykłej kartki papieru i przyborów do pisania.

Efekt końcowy wykonania strony zazwyczaj różni od szkicu, planowanie szkieletu witryny jest jednak ważne ze względu na nasze ograniczenia: mniej wprawionym projektantom trudno będzie zobaczyć efekt działania kodu bez jego wypróbowania. Co ważne nawet, jeśli coś osadzimy w HTML-u nie musi być to widoczne w przeglądarce. Mamy wtedy do czynienia z *przeźroczystymi pokojami*: mają one ściany, spełniają swoją funkcję, utrzymują w ryzach treść; jednak nie są widoczne dla użytkownika, o ich istnieniu wiedzą jedynie projektanci.

Nim zajmiemy się dokładnym określeniem położenia elementów na stronie, musimy zaplanować strukturę witryny. Wygląd naszej strony możemy jedynie wstępnie założyć ale nigdy przewidzieć. Poszczególne jej moduły będą mieć zmienne wymiary w zależności od osadzonej w

nich treści, co wymusi prace nad odpowiednim ich wypoźycjonowaniem. Poprawnie strukturalnie i semantycznie stworzona strona może, jeśli źle ją ostylujemy, dziwacznie wyświetlać się w przeglądarce. Czysty HTML nie daje nam pełnych możliwości zmian wizualnych na stronie, to jednak znaczniki informują przeglądarkę o zawartości i ilości pojemników składających się na naszą stronę.



```

1 <!DOCTYPE html5>
2 <head>
3   <meta charset="utf-8">
4   <link href="css/main.css" rel="stylesheet" type="text/css">
5   <title>strona_1</title>
6 </head>
7 <body>
8   <header>
9   </header>
10  <nav>
11    <ul>
12      <li><a href="" title=""></a></li>
13      <li><a href="" title=""></a></li>
14      <li><a href="" title=""></a></li>
15      <li><a href="" title=""></a></li>
16      <li><a href="" title=""></a></li>
17    </ul>
18  </nav>
19  <div>
20    <nav>
21      <ul>
22        <li><a href="" title=""></a></li>
23        <li><a href="" title=""></a></li>
24        <li><a href="" title=""></a></li>
25        <li><a href="" title=""></a></li>
26        <li><a href="" title=""></a></li>
27      </ul>
28    </nav>
29    <article>
30    </article>
31    <section>
32    </section>
33    <aside>
34    </aside>
35  </div>
36  <footer>
37  </footer>
38 </body>
39 </html>

```

Ryc. nr [4] - Wstępny układ strukturalny strony w kodzie HTML

Z zaprezentowanego powyżej kodu trudno domyśleć się wyglądu strony. Bez zastosowania stylu przeglądarka wyświetla każdy kolejny element blokowo jeden pod drugim - stąd potrzeba szkicowania schematów.

2.2 Sposoby pozycjonowania elementów pływających na stronie WWW

Obecnie standard w tworzeniu modelu witryn internetowych stanowi technika elementów pływających:

Za pomocą właściwości float można sprawić, żeby wybrany element był otoczony tekstem albo żeby elementy blokowe układały się obok siebie, co pozwala co pozwala na przykład na utworzenie układu kolumnowego. [3]

Istnieją jeszcze witryny oparte na układzie tabel i ramek, jednak nie są one uznawane za zgodne ze standardem. Tabele w założeniu nigdy nie miały służyć jako elementy strukturalne, a ramki przestały być stosowane ze względu na ich wpływ na logiczny układ treści i szybkość wczytywania strony.

Na bardziej precyzyjne rozmieszczanie modułów strony pozwala właściwość *position*, która przyjmuje cztery różne wartości ze względu na sposób pozycjonowania:

- **pozycjonowanie względne** [...] *polega na wyznaczaniu położenia elementów względem miejsca, w którym by się znalazły, gdyby nie zmieniono ich pozycjonowania.*
- **pozycjonowanie bezwzględne** [...] *położenie wyznaczają współrzędne osi poziomej i pionowej względem krawędzi najbliższego pozycjonowanego elementu nadrzędnego.*
- **pozycjonowanie stałe** [...] *podobne do bezwzględnego [...] położenie jest zawsze blokowe i wyznaczane względem krawędzi obszaru widoku [...] element [...] nie przesuwa się z resztą strony. [4]*

Powyższe informacje są istotne szczególnie dla osób, które postanowią zbudować stronę metodą *try and correct*, czyli bez przygotowania layoutu strony w skali przy pomocy programu

graficznego. Metoda ta wymaga dużo cierpliwości przy wprowadzaniu poprawek, ale skraca etap planowania.

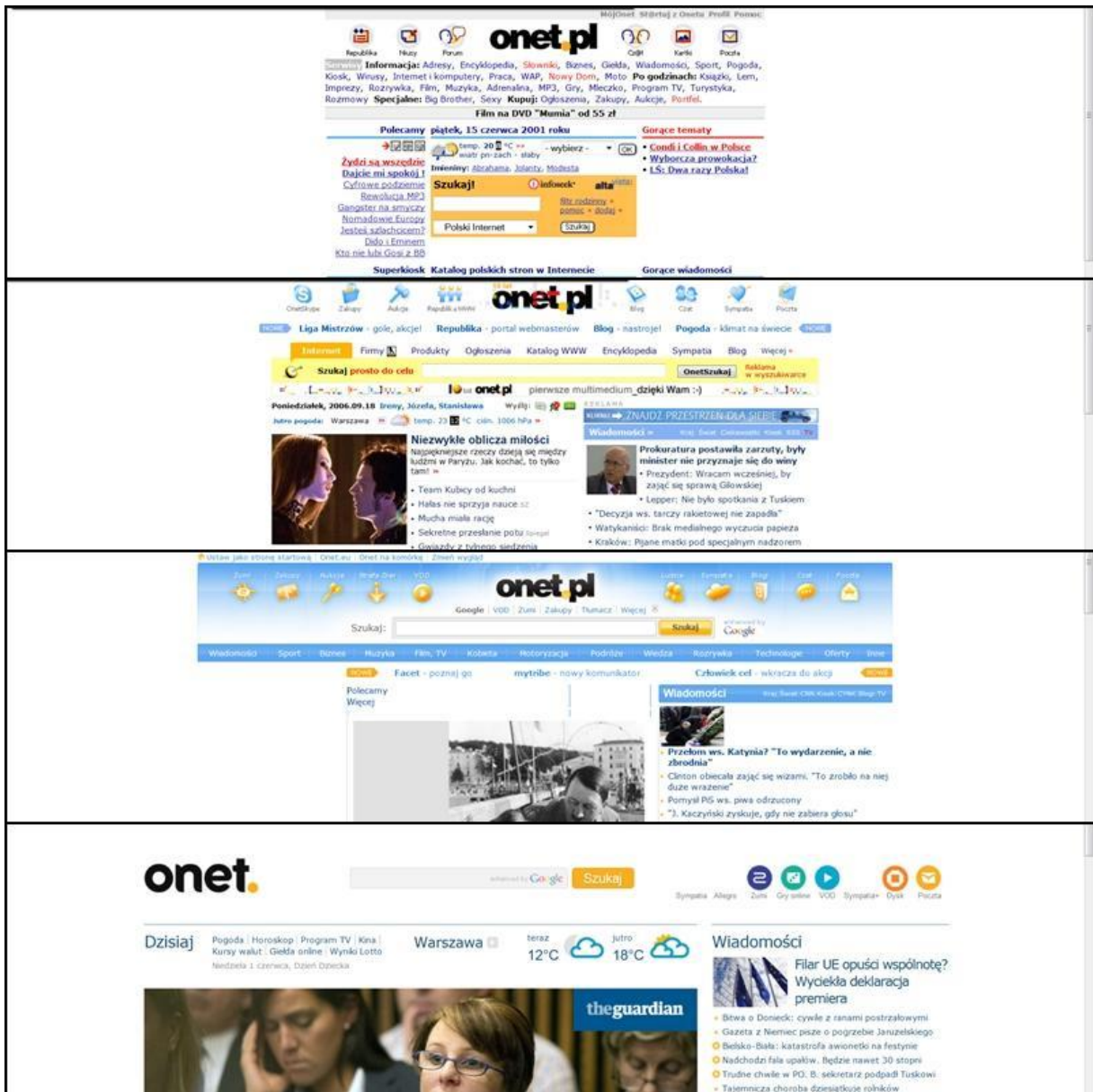
2.3 Jednostki stałe a jednostki relatywne.

Podczas projektowania układu mojej strony zdecydowałem się skorzystać z możliwości, jakie dają jednostki względne. Dzięki czemu udało mi się osiągnąć ciekawe rezultaty w postaci witryny skalowalnej w przeglądarce. Obecnie trend stanowią strony responsywne (uwzględniające rodzaj urządzenia, na którym są wyświetlane), ja osiągnąłem zbliżone rezultaty. Nie korzystałem jednak z nowych funkcjonalności jakie daje HTML5 i CSS3, czyli zapytań o media, stąd mój projekt nie może być uznany za wzorcowy w tym aspekcie.

Dużo trudniej wykonać układ strony w jednostkach relatywnych, jednak ich zastosowanie pozwala witrynie lepiej dostosowywać się do użytkowników. Strona zwymiarowana w pikselach zawsze będzie miała identyczną rozpiętość w szerokości. Jeśli zechcemy możemy też określić stałą wysokość kontenerów, nie jest to jednak polecane w przypadku bloków z osadzoną wewnątrz treścią tekstową, gdyż za każdym razem, gdy zechcemy coś dopisać, kontener nie zwiększy się acz utnie nam treść lub pokaże niezbyt estetyczne paski przewijania wewnątrz strony. Użytkownicy stron nie lubią dużej ilości scroll'ów, gdyż przeszkadzają w szybkim przeglądaniu treści.

Musimy pamiętać, że nie wszyscy używają w swoich urządzeniach takiej samej rozdzielczości jak my. Strona wykonana w jednostkach stałych, która u Nas prezentuje się dobrze może wyglądać gorzej u innych oglądających. Jeśli jesteśmy ciekawi jak prezentowały się strony dawniej, zwracam uwagę na szerokość layoutu, możemy zajrzeć na: <http://archive.org/web/web.php> Strona pokazuje starsze wersje popularnych stron WWW.

Wyobraźmy sobie teraz, że ktoś zbudował stronę pracując na urządzeniu małym ekranie, niskiej rozdzielczości w np. Notebook'u i stosując jednostki stałe. Jeśli wspomniany wyżej projektant założy, że jego strona powinna rozciągać od krawędzi do krawędzi ekranu 12 cali zapewne ucieszy tym użytkowników małych urządzeń ale może wywołać niechęć u osób preferujących większy kaliber sprzętu.



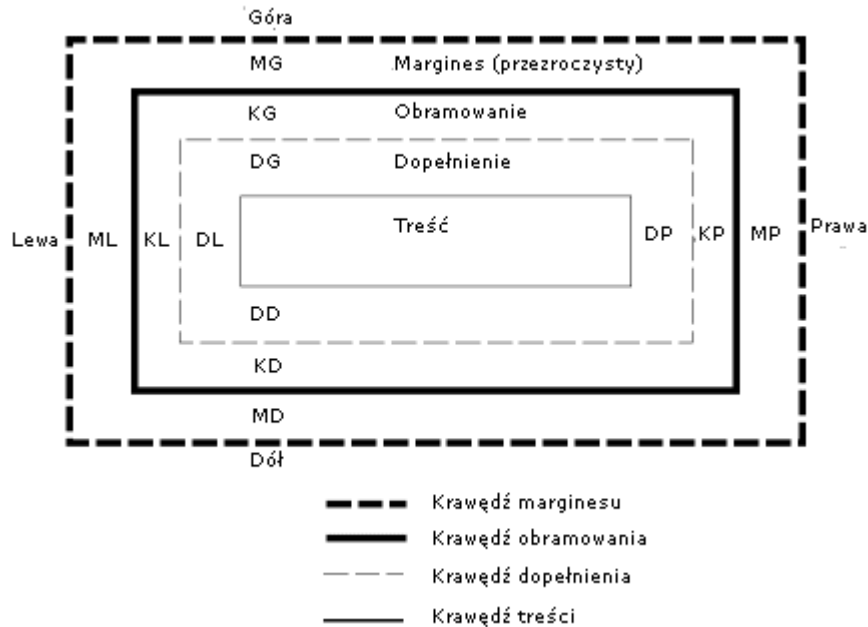
Ryc. nr [5] - strona Onetu na przestrzeni lat - 2001, 2006, 2010, 2014 wyświetlana na monitorze o rozdzielczości 1920x1080 w przeglądarce Firefox 30

Jeśli natomiast zwymiarujemy szerokość głównego kontenera strony body np. w procentach: *width: 80%*, *margin: auto* na każdym urządzeniu nasza strona będzie wyglądać podobnie, gdyż określiliśmy jej rozpiętość jako $\frac{4}{5}$ całości, gdzie $\frac{1}{5}$ stanowi puste miejsce na marginesy lewy i prawy. Oczywiście musimy pamiętać, osadzając kolejne elementy wewnątrz głównego, że będą one liczone względem tego nadrzędnego. Tak więc np. header elementu body będzie stanowił 100% jego długości wtedy, gdy chcemy by rozciągał się od marginesu do

marginesu. Z początku rzecz wydaje się banalnie prosta, sprawę jednak mogą skomplikować dopełnienia, ramki i marginesy kolejnych elementów potomnych. Zmiana struktury strony w jednostkach relatywnych może być trudniejsza.

2.4 Własności pól elementów - dopełnienia, marginesy, ramki.

Każdy element HTML prócz wysokości i szerokości ma dodatkowe właściwości domyślne do przemodelowania za pomocą kaskadowych arkuszy stylu. Ustawienia dotyczące marginesów, dopełnień jak i ramek mają duży wpływ na estetyczne rozmieszczenie treści na stronie: są potrzebne do pozycjonowania modułów składowych strony.



Ryc. nr [\[6\]](#) Graficzny opis modelu polowego.

Poszczególne elementy potomne strony WWW dziedziczą właściwości elementów nadrzędnych (ich rodziców). Jest to fundamentalna cecha języka HTML (w css wartość *inherit* można zmienić na *none*, wtedy element przestaje przejmować cechy rodzica).

Zagnieżdżając kolejne elementy musimy pamiętać o hierarchii poszczególnych potomków. Element A, który jest dzieckiem elementu B i wnukiem elementu C, będzie liczył, wartości szerokości wyrażone w procentach, względem elementu B ale rodzaj czcionki

odziedziczy już po elemencie C (jeśli oczywiście nie wymusimy zmiany bezpośrednio w A). Zwracanie uwagę na dziedziczenie jest szczególnie ważne przy tworzeniu strony opartej na jednostkach relatywnych. Sam miałem spore problemy z dostosowaniem pól, gdyż z początku nie założyłem, że będę potrzebował miejsce na ramki, w efekcie musiałem przeprojektowywać cały szablon. Każdy webmaster powinien pamiętać, że strona internetowa, to także to czego nie widać.

Kiedy nasz przemyślany układ pokoi, kontenerów zostanie przełożony na kod z uwzględnieniem grubości ścian może przystąpić do trzeciego etapu, czyli osadzania treści i jej stylizacji: do tego potrzebna będzie nam biblioteka grafik, mapa odnośników i znajomość podstaw CSS.

3. Praca nad stylizacją i wzbogacaniem witryny - CSS3 i narzędzia

3.1 Otwarta struktura strony

Etap opisany w rozdziale drugim nie kończy się na wykonaniu jednego szablonu. Często nie jesteśmy w stanie przewidzieć ile podstron będzie liczył nasz serwis i jak będzie wyglądała ich funkcjonalność. Stąd nie należy przechodzić do etapu stylizacji strony bez dokładnego rozplanowania układu treści.

Może dojść do sytuacji, że przygotujemy sobie dobrze wyglądającą stronę główną z zaskakującymi efektami wizualnymi i zechcemy ją powielić: poprawniej jest wykonać jeden szablon, który zostanie zastosowany do wszystkich podstron. Gdy już to zrobimy, wykonamy szereg udoskonaleń wizualnych, dodamy hiperłącza, trudno nam będzie wprowadzać niektóre zmiany: np. gdy zechcemy zrezygnować ze stopki, będziemy musieli usunąć ją ze wszystkich plików .html

Sam wpadłem w podobną pułapkę. Wykonując podstronę z formularzem i zakładkę news zostałem zmuszony do modyfikacji szablonu, który z początku miał pełnić inną funkcję. Zapomniałem o wykonaniu osobnych plików .html i .css dla tych zakładek na wcześniejszym etapie pracy, co przysporzyło mi tylko trudności.

Warto pamiętać, że chociaż kolejne stadia pracy nad witryną zazębiają się ze sobą, to nigdy nie zostają ostatecznie zamknięte. Strona WWW to dokument bardzo dynamiczny, zmiany

są wpisane w jego *sieciową egzystencję*. Struktura witryny tak naprawdę nigdy nie zostaje zamknięta i każdy projektant powinien ją budować, tak dało ją się szybko i łatwo aktualizować.

W rozdziale trzecim opisze wiele technik użytecznych przy generowaniu strony WWW. Tworzenie instrukcji obsługi dla Adobe Dreamweaver'a wydało mi się krokiem zbędnym, gdyż najlepiej jego funkcjonalność widać podczas pracy z CSS. Wykorzystanie arkuszy stylu będzie głównym zagadnieniem tej części pracy.

3.2 Przygotowanie i dostosowanie zestawu grafik na stronę: wykorzystanie funkcjonalności Photoshop'a i Illustrator'a.

Po pobraniu z Internetu, zgraniu na dysk twardy komputera, odpowiednim posegregowaniu i ponazywaniu grafik, możemy przystąpić do ich obróbki celem zamieszczenia na stronie.

Warto nadmienić na samym początku, że mamy kilka podstawowych formatów grafik stosowanych na stronach, najpopularniejsze z nich to:

- **JPEG** [...] *niezwykle popularny format, obsługiwany przez wszystkie przeglądarki i programy graficzne.*

Dane w JPEG są kompresowane stratnie przy użyciu dyskretnej transformacji cosinusowej oraz kodowania Huffmana. W rezultacie pliki mają stosunkowo niewielkie rozmiary, ale odbywa się to kosztem jakości. Ponadto każdorazowe zapisanie pliku w dowolnym programie graficznym (opcja „save”) powoduje działanie kompresji i bezpowrotną utratę danych.

Maksymalna głębokość koloru, z jaką można zapisać obrazy w tym formacie to 24 bity na piksel. Co więcej, format oferuje duży wybór w zakresie modelu przestrzeni barwnej (m.in. RGB, sRGB, CMYK). Niestety, nie umożliwia zapisania metadanych w schemacie XMP. [5]

- **PNG** [...] należy do rastrowego formatu plików, który prezentowany jest przy pomocy poziomo-pionowej siatki złożonej z pikseli, każdy posiadający odpowiednią barwę. Format ten stosowany jest do bezstratnej kompresji plików graficznych, co oznacza, że istnieje możliwość odtworzenia pliku graficznego poddanego kompresji (czyli zmniejszenie liczby bitów) w takiej samej postaci, jak przed kompresją.

PNG ma rozmaite zastosowania. Do niektórych z nich należy zaliczyć fotografię, gdzie możliwe jest zastosowanie palety od 2 do 256 kolorów wraz z trybem szarości. Dzięki temu pozbywamy się nadmiaru informacji skupiając się na treści obrazu. Umożliwia także zapis zdjęcia w 48-bitowej głębi kolorów RGB, co daje nam miliony kolorów.

PNG stosuje również filtry graficzne, których zadaniem jest uproszczenie danych tuż przed kompresją, co powoduje poprawienie jej wydajności.

Edycja kolorów możliwa jest dzięki wielu programom graficznym. Do tych darmowych i prostszych w obsłudze zaliczyć można np. IrfanView, do tych bardziej zaawansowanych – GIMP-a, dzięki któremu możemy edytować poszczególne warstwy i automatycznie redukować liczbę kolorów.[\[6\]](#)

- **SVG HTML5** wprowadza nowy wymiar dostarczania wysokiej jakości grafiki do stron internetowych. Nie trzeba już używać żadnych wtyczek, dodatków, pluginów. Wystarczy poznać format SVG lub wirtualne płótno Canvas. Oba mają różne zastosowanie, ale jedno jest wspólne - za wyświetlenie grafiki będzie odpowiadała przeglądarka użytkownika, która może wykorzystywać moc karty graficznej komputera.

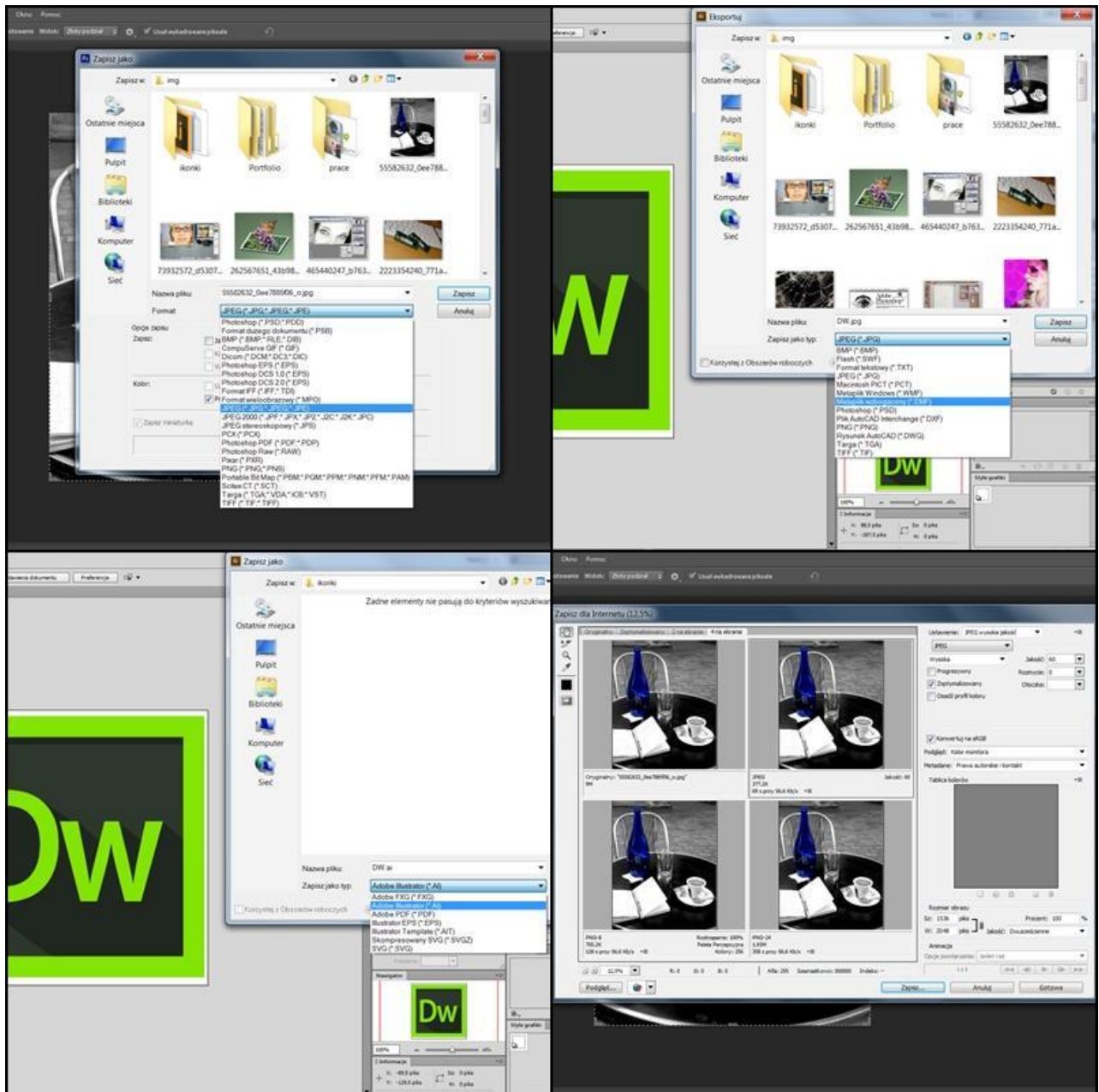
SVG (ang. Scalable Vector Graphics) to język opisu grafiki w postaci XML-owej. Wystarczy zwykły notatnik do przygotowania jej, choć można też posłużyć się specjalnym programem. Opisana w ten sposób grafika jest wektorowa, co w praktyce oznacza, że nie występuje zjawisko pikselizacji przy zbliżeniu lub

oddaleniu. Dzieje się tak dlatego, że w przeciwieństwie do JPEG-ów, GIF-ów itd., SVG opisuje kształt matematycznie, a nie jest zbiorem ściśle zdefiniowanych pikseli[...] [\[7\]](#)

- **GIF** [...] jest formatem pliku graficznego z kompresją bezstratną, czyli taką, która polega na zmniejszeniu liczby bitów poprzez kompresję, a następnie odtworzenie takiego pliku w postaci pierwotnej, czyli bezstratnej[...] Tworzone w formacie GIF animacje zyskały na przestrzeni lat ogromną popularność, pojawiając się w takich serwisach internetowych jak Tumblr, Myspace, czy 9Gag, przedstawiając zabawne animacje, bannery reklamowe, avatary na forach reklamowych czy podpisy w e-mailach. Warto wspomnieć, że pierwsze GIF-y były czarno-białe i wykorzystywały paletę 256 kolorów. Spowodowane to było niską przepustowością łącz modemowych – a im mniej „ważący” gif, tym szybciej się ładował. [\[8\]](#)

Wybór typu grafiki może mieć duże znaczenie: liczy się wielkość pliku, obsługa przeźroczystości i możliwość skalowania. W wyliczeniu celowo pomijam możliwość generowania animacji, gdyż moja strona stwarza wrażenie dynamicznej dzięki HTML5 i CSS3. Nie osadzałem animacji w plikach graficznych ani nie korzystałem z Adobe Flash.

W pierwszej kolejności skupimy się na kompresji i eksporcie grafiki do wybranego formatu. Zarówno Illustrator jak i Photoshop, pod tym względem, oferują nam mnóstwo narzędzi.



Ryc. nr [6] - Praca w Dreamweaver i Illustrator, okienka zapisu i eksportu plików do różnych formatów, czwarty obrazek prezentuje narzędzie do porównywania i optymalizacji grafiki w Photoshop'ie w różnych formatach i jakości (podobne narzędzie spotkamy także w Illustratorze).

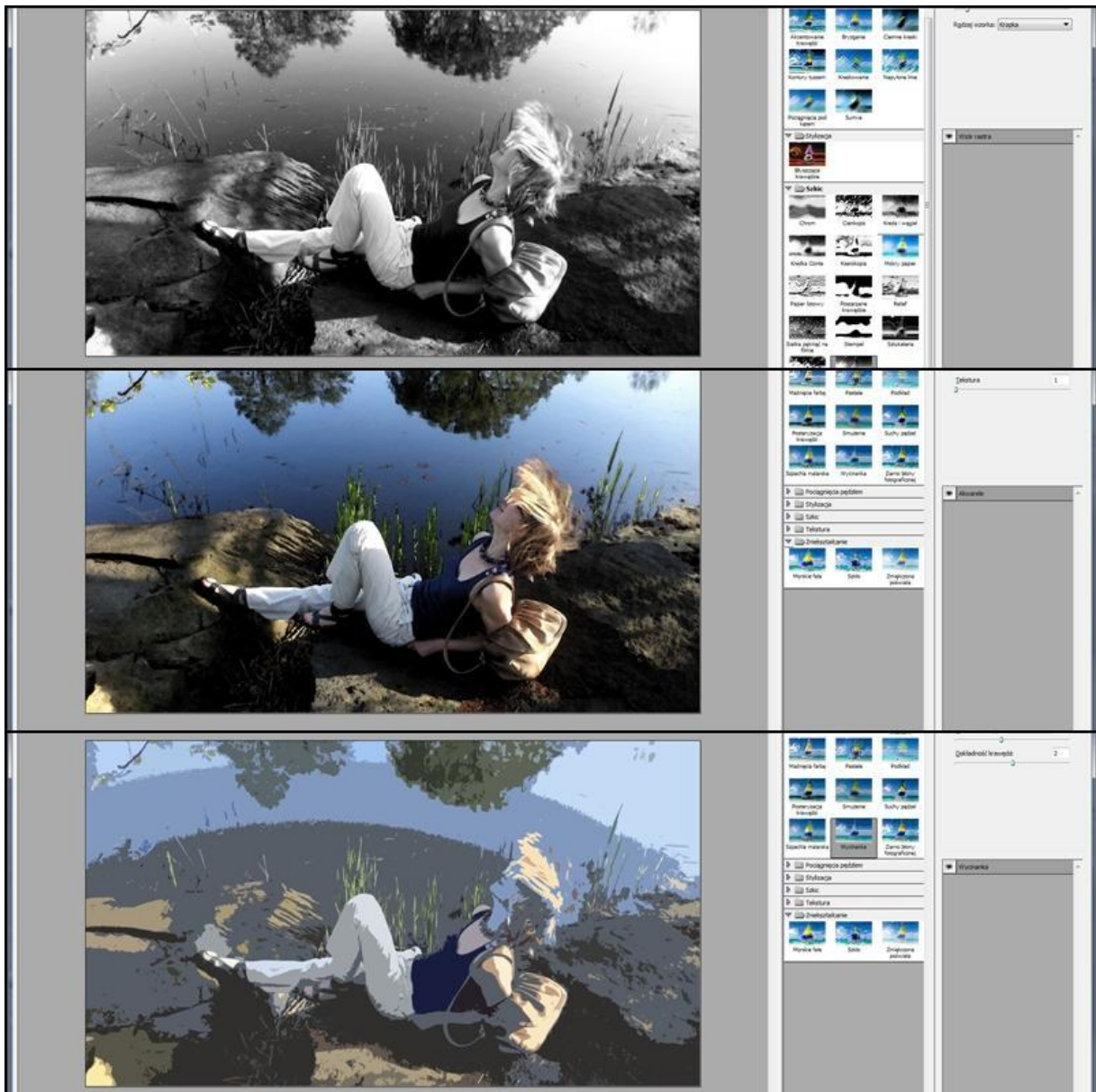
Narzędzie do optymalizacji grafiki uwalnia nas od *rozu* komputera, sami, przy pomocy własnych oczu może ocenić czy konwersja stratna wywołuje jakieś widoczny defekt, czy też jest niezauważalna dla innych. Możemy dzięki tej funkcji porównać i selekcjonować wiele grafik.

Przy edycji grafiki na stronę nie zaszkodzi odrobina wiedzy teoretycznej, warto wiedzieć np. że .jpg nie obsługuje przezroczystości, a tylko formaty wektorowe jak .svg czy .gif ma możliwość skalowania przy pomocy css, bez utraty jakości. Przestrzegam również przed wprowadzaniem nowinek, a zachęcam by powoływać się na stare sprawdzone sposoby. Sam spędziłem wiele czasu w poszukiwaniu odpowiedzi, na pytanie, jak rozwiązać problem obsługi filtrów graficznych z poziomu CSS. Okazało się, że Firefox (moja ulubiona przeglądarka nie obsługuje ich), potem przeczytałem na jednym z portali:

[...]filtry są w chwili obecnej funkcją eksperymentalną wspieraną przez silnik webkit[...][\[9\]](#)

Zanim więc postanowimy na dobre uwolnić się od programów graficznych dobrze to przemyślimy.

Nie sposób jest umówić wszystkich możliwości Photoshopa czy Illustratora, programy te pozwalają robić z grafiką praktycznie wszystko. Ja jednak skupię na automatycznych funkcjach tych aplikacji czyli filtrach. Filtry to wbudowane w program zestawy operacji, narzędzia; które w szybki sposób pozwalają uzyskać zmiany takie jak: sepia, rozmycie, saturacja, odwrócenie barw, negatyw, skala szarości i wiele innych, takich jak: deformacje czy zmiana tekstury. Co ważne sam efekt, gotowy do zastosowania, możemy regulować z poglądem tego, co się stanie po jego użyciu. To bardzo wygodne i przyjemne.



Ryc. nr [8] - Prezentacja działania niektórych filtrów z Adobe Photoshop - wzór rasta, akwarele, wycinanka

Photoshop i Illustrator posiadają niemal identyczne zestawy filtrów. Różnią się jednak dodatkowymi narzędziami, określanymi jako efekty: Photoshop jest zorientowany głównie na grafikę rastrową, Illustrator zaś wektorową. Grafika wektorowa wykazuje większą podatność na formowanie, ze względu na swą specyfikę:

[...]sposób zapisu obrazów za pomocą figur geometrycznych, linii i krzywych, które są przechowywane w postaci równań algebraicznych[...][\[10\]](#)

Rastrowa grafika, gdzie:

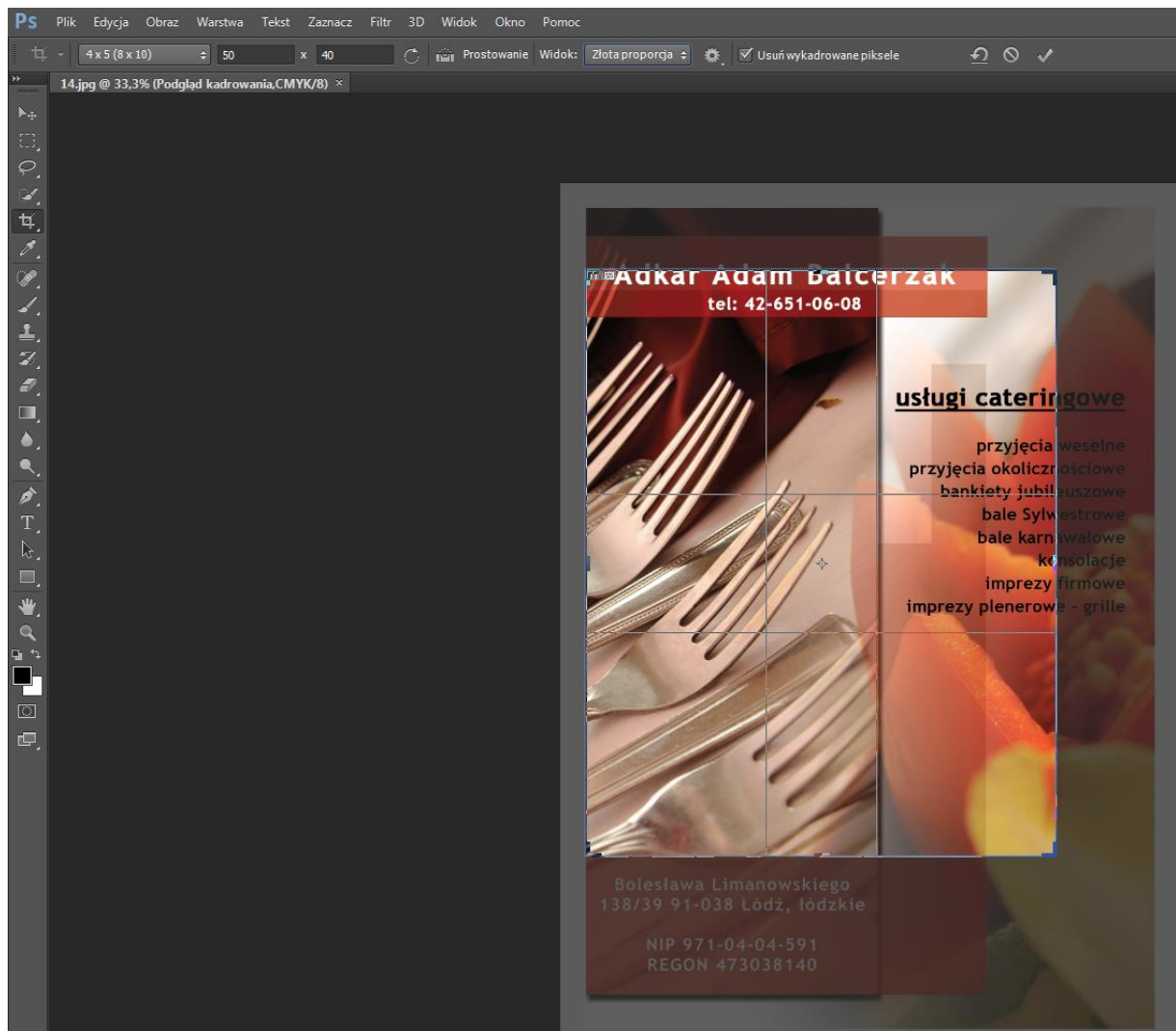
[...]bez zastosowania kompresji kolor każdego piksela jest definiowany pojedynczo[...][\[11\]](#)

ma możliwość przechowywania dużej ilości szczegółów. Ten sposób prezentacji grafiki służy do zdjęć, gdzie wierne odzwierciedlenie rzeczywistości jest istotne.

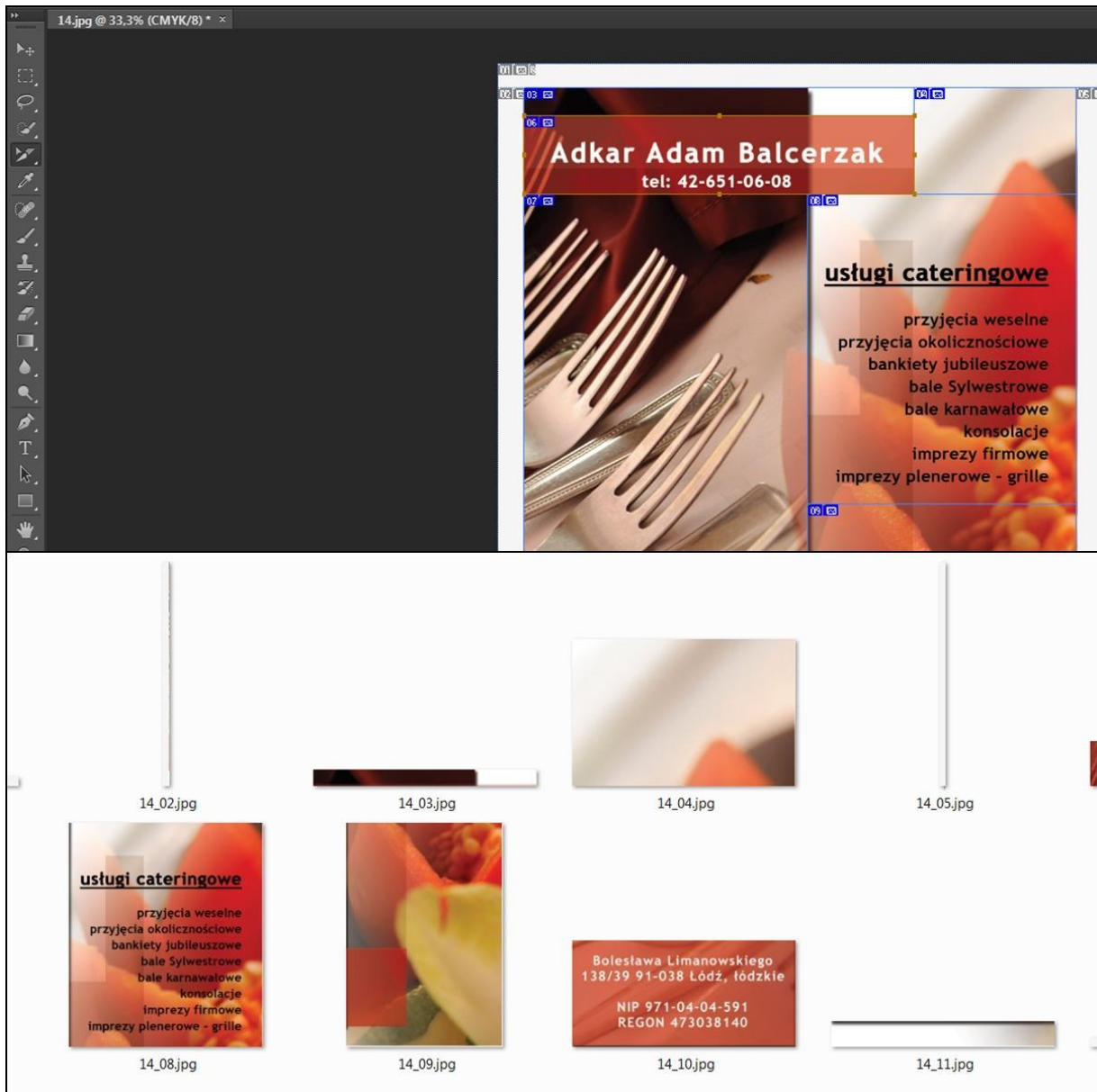
Oczywiście nie ma przeszkód, by format rastrowy zmienić w wektorowy: Illustrator ma taką możliwość. Funkcja ta nazywa obrysowywanie ścieżek i czasem bywa przydatna. O ile wektoryzacja jest dość skomplikowanym procesem, gdyż nie każdą bitmapę da się przełożyć na proste figury geometryczne, to resteryzacja przyjdzie nam łatwo, gdyż jest procedurą mniej złożoną. Dwa wspomniane przeze mnie produkty Adobe, obsługują wiele formatów plików i pozwalają na ich eksport (o czym warto pamiętać zresteryzowany plik traci swoje właściwości, czyli możliwość bezstratnego skalowania).

Kolejne wersje Photoshopa oferują nam coraz wygodniejsze narzędzia do kadrowania. Wydawałoby się, że kwestii przycinania zdjęć nie da wymyśleć niczego nowego, a jednak Adobe ciągle zaskakuje użytkowników. Photoshop oferuje trzy narzędzia z grupy Crop Tools, ja jednak opisze tylko dwa, niezwykle przydatne w pracy projektanta stron www:

- **Kadrowanie z uwzględnieniem proporcji okna:** pozwala w bardzo przejrzysty i intuicyjny sposób wycinać prostokątne obszary, w których możemy ustalać stosunek długości boków. Narzędzie udostępnia różne opcje widoku siatki takie jak: złoty podział, złota spirala czy przekątna. Udogodnienie to przydaje się zawodowym fotografom jak i grafikom.
- **Cięcie na plasterki:** narzędzie pozwalające dzielić pojedyncze zdjęcie, w bardzo dynamiczny i szybki sposób, na wiele prostokątnych grafik. Pociętą fotografię można zapisać w małych kawałkach z kodem HTML lub bez niego. Daje to projektantowi dużą wygodę i skraca czas jego pracy. Warto dodać, że narzędzie do cięcia na plasterki świetnie współpracuje z narzędziem do optymalizacji grafiki.



Ryc. nr [9] - Prezentacja działania narzędzia kadrowanie, Adobe Photoshop CS6



Ryc. nr [\[10\]](#) - Prezentacja działania narzędzia cięcie na pasterki, Adobe Photoshop CS6. Ilustracja na dole pokazuje osobno automatycznie zapisane fragmenty wizytówki.

Przy umieszczaniu grafiki warto korzystać z pomocy Dreamweaver'a, program posiada opcję automatycznego tworzenia hiperłącza do pliku po wskazaniu na miejsce jego położenia. Ta prosta funkcja często przydaje się początkującym, a staje się niezbędna dla webmasterów zarządzających wielkimi serwisami, pojedynczy plik, bowiem łatwo zgubić podczas nawalu pracy.

Narzędzia Adobe świetnie ze sobą współpracują, co daje im jeszcze większe pole działania. Sam ograniczyłem się do kilku funkcji wymienionych aplikacji, wskazanych powyżej, ale z pewnością znajdą się projektanci, którzy jeszcze bardziej wykorzystują potencjał programów Adobe. Ja skupiłem się na pracy z kodem, ale z równie owocnym rezultatem, można stworzyć witrynę wizytówkę bazując na plasterkach i warstwach w Photoshop.

3.3 Dopasowanie kolorystyki elementów strony.

Właściwy dobór kolorów to istotny etap tworzenia strony. Jeśli źle dopasujemy barwy na stronie będą one zniechęcać odwiedzających. Każdy kolor jest informacją, zawiera pewne przesłanie, odpowiada określonym trendom. Nie musimy oczywiście studiować symboliki kolorów ale warto byśmy, budując stronę, myśleli przede wszystkim o jej przejrzystości i czytelności. Zwykle najlepsze są strony, które ograniczają formę a skupiają się na treści: tak by w oczy rzucały się nagłówki, czcionka i nazwy linków.

Warto przed przystąpieniem do pracy stworzyć sobie profil kolorystyczny naszej strony z zapisanymi kolorami w notacji HEX (gdyż są najkrótsze) i RGBA (gdyż możemy potrzebować przezroczystości). Ja podczas mojej pracy korzystałem ze strony **<http://fonty.pl/kolory.htm#generator>** i palety kolorów na niej zawartej, ale podobnych generatorów spotkamy w sieci wiele. Muszę przyznać, że strona WWW z paletą uwolniła mnie od pracy z Photoshop'em, gdyż okazała się szybszym rozwiązaniem.

Statyczny nagłówek mojej strony urozmaiciłem o efekt gradientu, trudno go było stworzyć posługując się wyłącznie wyobraźnią i kodem, więc skorzystałem z generatora gradientów **<http://www.colorzilla.com/gradient-editor/>**, uzyskałem następujące właściwości css, które potem wstawiłem do głównego pliku ze stylami:

1. **background:** `-moz-linear-gradient(left, #236b8e 0%, #66cc00 0%, #236b8e 46%);`
2. **background:** `-webkit-gradient(linear, left top, right top, color-stop(0%,#23688e), color-stop(0%,#66cc00), color-stop(46%,#236b8e));`
3. **background:** `-webkit-linear-gradient(left, #236b8e 0%,#66cc00 0%,#236b8e 46%);`
4. **background:** `-o-linear-gradient(left, #236b8e 0%,#66cc00 0%,#236b8e 46%);`
5. **background:** `-ms-linear-gradient(left, #236b8e 0%,#66cc00 0%,#236b8e 46%);`
6. **background:** `linear-gradient(to right, #236b8e 0%,#66cc00 0%,#236b8e 46%);`
7. **filter:** `progid:DXImageTransform.Microsoft.gradient(startColorstr='#236b8e', endColorstr='#236b8e',GradientType=1);`

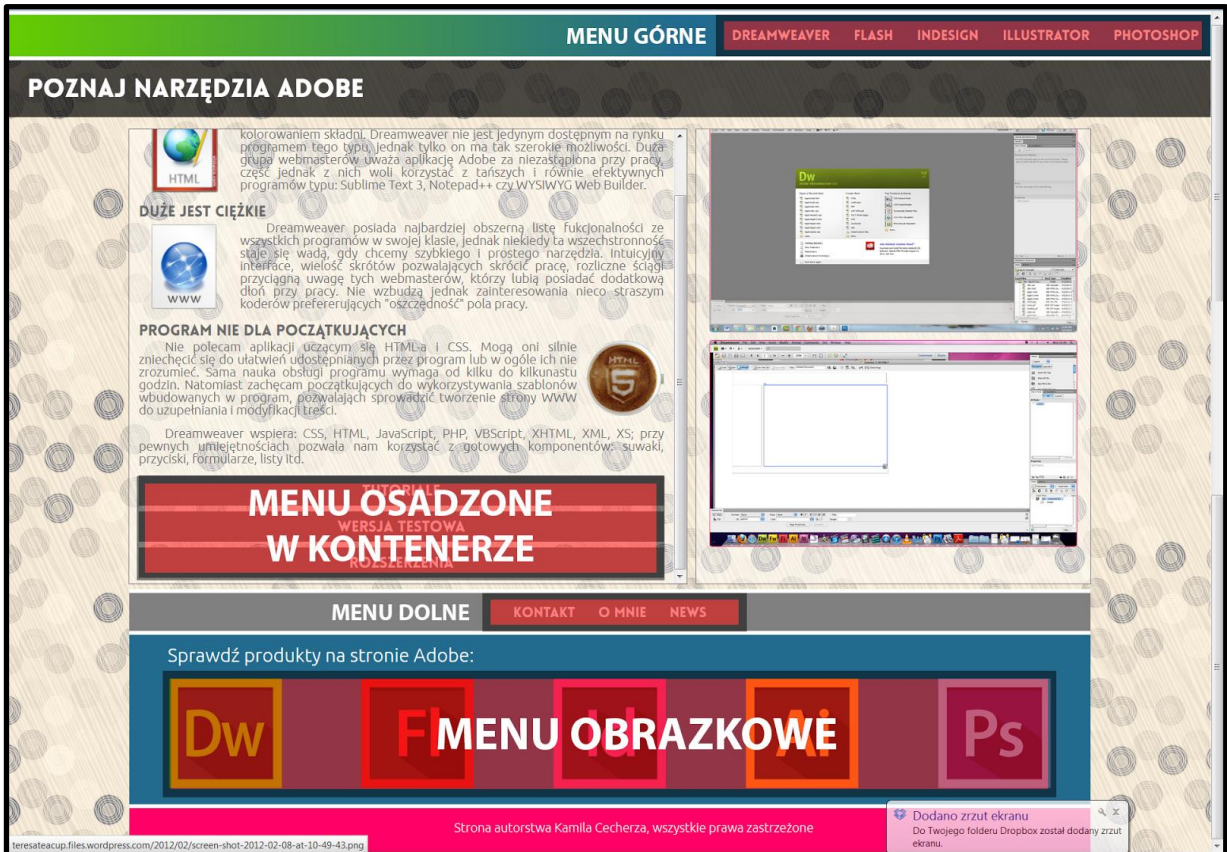
Tło nagłówka z animowanym napisem stworzyłem używając przeźroczystości, notacja RBA. Jako tapetę dla całej strony wykorzystałem małą grafikę, którą powieliłem stwarzając efekt spójności, własności *background-image* i *background-repeat*. Dalsze elementy zabarwiłem bazując na wyżej wspomnianej palecie, z kolorami w notacji HEX.

3.4 Tworzenie menu: formatowanie odnośników, efekty przejścia.

CSS poziomu 3 trzeciego wprowadza całkiem nowe właściwości, które w prosty sposób pozwalają nadawać elementom na stronie płynne efekty ruchu. Obecnie wszystkie aktualne wersje znanych przeglądarek obsługują *transition*.

Podczas tworzenia nawigacji na mojej stronie: menu górnego, menu dolnego, menu obrazkowego, menu osadzonego w kontenerze artykułu cały czas wykorzystywałem efekty przejścia. Dzięki temu uzyskałem wrażenie większego dynamizmu tych elementów. Z pozoru niewielka modyfikacja, bo pseudo-klassy są znane już od wcześniejszych wersji specyfikacji CSS, polegająca na kilkusekundowym opóźnieniu zmiany stanu elementu: jego koloru, kształtu czy położenia, pozwala wejść odbiorcy strony na inny poziom odbioru.

Płynne przejścia są dużo bardziej atrakcyjne niż przejścia dwustanowe. CSS3 oferuje nam ciekawe możliwości bez konieczności znajomości programowania, a niekiedy uwalnia od Flash'a. Nie musimy opisywać *kolejnych klatek*, wystarczy wygenerowanie dwóch linijek kodu, w których uwzględnimy: czas i rodzaj przejścia. To niezwykle intuicyjne.



Ryc. nr [11] - Rodzaje menu na stronie mojego projektu.

Etapy tworzenia pojedynczego efektu *transitions* są stosunkowo proste, pokażę to na przykładzie mojej strony:

```

310 section#hyperlink_image > div:nth-child(5) {
311     background-image: url(..img/ikonki/AI.png);
312 }
313 section#hyperlink_image > div:nth-child(6) {
314     background-image: url(..img/ikonki/PS.png);
315 }
316 section#hyperlink_image > div > a {
317 }
318 section#hyperlink_image > div > a > img {
319     width: 100%;
320     height: auto;
321     transition: 0.5s ease-in opacity;
322 }
323 section#hyperlink_image > div > a > img:hover {
324     opacity: 0;
325 }

```

Ryc. nr [12] - Fragment głównego arkusza stylu, dotyczący obrazkowego menu.

1. W wierszach od 310 do 314 widzimy sekcje, mające nadane obrazkowe tło, kwadratowe logo. Na razie jest ono niewidoczne, zasłania je większy element, który jest tą samą grafiką.
2. Wiersz 321 opisuje zachowanie hiperłącza, w którym zawarty jest element `img`, który przysłania tło (mniejsze logo odpowiednio wyśrodkowane). Mamy tu czas, rodzaj przebiegu przejścia i właściwość, do jakiej zostanie ono zastosowane.
3. Wiersz 323 opisuje zachowanie elementu po najechaniu na niego kursorem myszy (pseudo-klasa `:hover`). Wartość opacity ustawiona jest na zero, co znaczy że element w czasie 0.5s będzie stopniowo zanikał po detekcji zdarzenia, do momentu odkrycia tła (mniejszego logo).

Zastosowany patent jest koncepcyjnym zastosowaniem przejść, ale możemy tworzyć dużo prostsze efekty polegające tylko na zmianie koloru z poziomu kodu. Więcej informacji jest na stronie:

<http://www.w3.org/TR/css3-transitions/>

3.5 Praca nad typografią - własność `@font-face`.

Tworząc stronę warto poświęcić czas typografii, treść ubrana w lepszy garnitur czcionek jest atrakcyjniejsza i lepiej skupia uwagę odbiorcy. O ile w przypadku efektów takich jak: animacje, przejścia czy filtry zalecam pewną ostrożność, to w przypadku czcionek sugeruje większą odwagę. Jeśli poprawnie wykonamy wszystkie kroki jest niewielkie prawdopodobieństwo, że użytkownik nie zobaczy właściwego kroju tekstu. Konieczność stosowania bezpiecznych czcionek powoli przestaje obowiązywać, należy jednak pamiętać, że kroje takie: Arial, Times New Roman, Georgia czy Verdana stanowią dobry standard: na wszystkich systemach operacyjnych, także Linux, MacOS wyglądają podobnie; obsługują polskie znaki; są utrwalone w świadomości użytkowników Sieci. Bezpiecznie fonty nie pozwalają nam się wyróżnić ale nadal są stosowane w wielu przypadkach.

Dawniej przed opracowaniem możliwości importu czcionek na stronę w postaci reguły @font-face, projektanci zapisywali je w postaci zwykłej grafiki: było to dość pracochłonne i uniemożliwiało użytkownikowi to np. zaznaczanie, kopiowanie i drukowanie tekstu.

Dziś mamy możliwość korzystania z setek formatów czcionek dostępnych jak wolne fonty na wielu serwisach w Sieci. Daje to dużo większe możliwości stylizacyjne: każdy grafik powinien jednak wiedzieć, że i w przypadku ilości krojów na raz, na stronie nie jest wskazana przesada. Jak na mojej stronie użyłem w zasadzie dwóch formatów:

- **loveloblack** <http://fontfabric.com/lovelo-font/>
- **ubuntulight** <http://www.fontsquirrel.com/fonts/ubuntu>

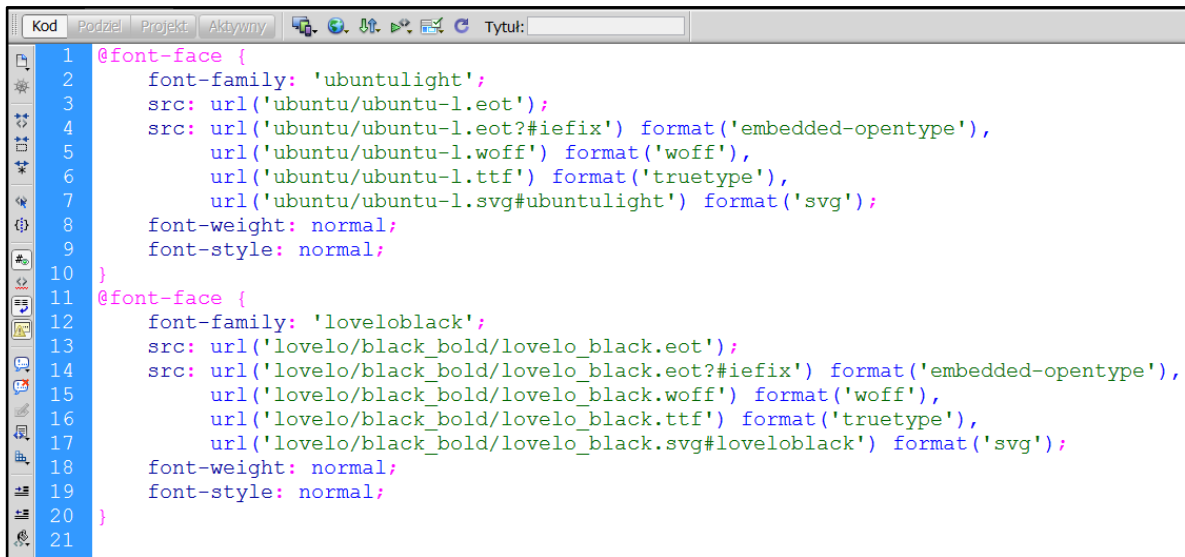
Udostępnionych na zasadach licencji Creative Commons.

Sporo czasu zaoszczędziłem dzięki narzędziu: webfont generator, dostępnym na stronie:

<http://www.fontsquirrel.com/tools/webfont-generator>

które umożliwiło mi konwersję czcionek do wielu formatów i automatyczne wygenerowanie plików css. Jest to bezpieczne rozwiązanie, gdyż obecnie każdy typ przeglądarki obsługuje własne formaty czcionki: .eot, .ttf, .otf, .wtf lub po prostu .svg

Chcąc zaoszczędzić pracę przy tworzeniu ścieżek, wszystkie reguły stylu odpowiedzialne za fonty umieściłem z osobnym pliku, gdzie pogrupowałem też czcionki:

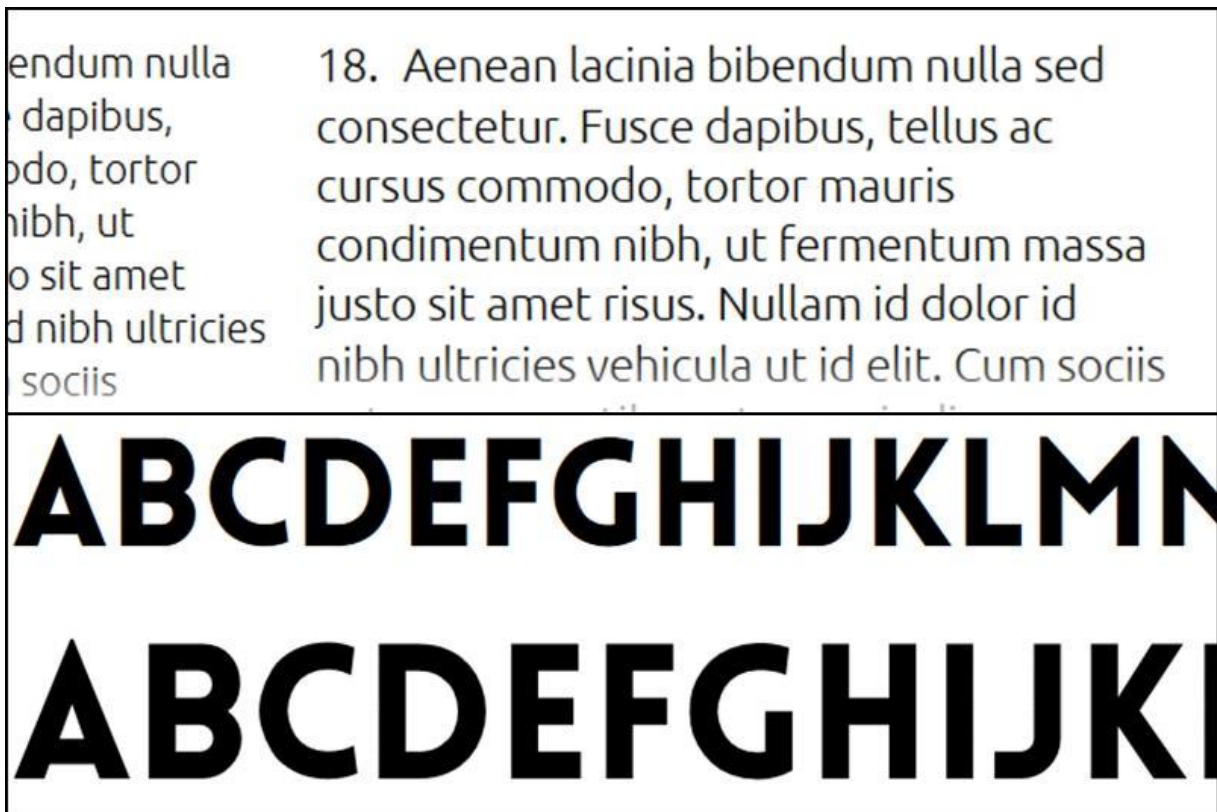


```

1  @font-face {
2      font-family: 'ubuntulight';
3      src: url('ubuntu/ubuntu-l.eot');
4      src: url('ubuntu/ubuntu-l.eot?#iefix') format('embedded-opentype'),
5           url('ubuntu/ubuntu-l.woff') format('woff'),
6           url('ubuntu/ubuntu-l.ttf') format('truetype'),
7           url('ubuntu/ubuntu-l.svg#ubuntulight') format('svg');
8      font-weight: normal;
9      font-style: normal;
10 }
11 @font-face {
12     font-family: 'loveloblack';
13     src: url('lovelo/black_bold/lovelo_black.eot');
14     src: url('lovelo/black_bold/lovelo_black.eot?#iefix') format('embedded-opentype'),
15          url('lovelo/black_bold/lovelo_black.woff') format('woff'),
16          url('lovelo/black_bold/lovelo_black.ttf') format('truetype'),
17          url('lovelo/black_bold/lovelo_black.svg#loveloblack') format('svg');
18     font-weight: normal;
19     font-style: normal;
20 }
21

```

Ryc. nr [13] - Arkusz styli odpowiedzialny za import czcionek.



Ryc. nr [14] - Prezentacja wyglądu czcionek *ubuntu* i *lovelo*, czcionka *lovelo* posłużyła mi do tworzenia nawigacji, *ubuntu* zaś wykorzystałem do tekstów artykułów.

Przyznam, że byłem zaskoczony jakie efekty podczas mojej pracy z czcionkami jak daje manipulacja: kolorami, rozmiarem, przeźroczystością. Z początku miałem wykorzystać kilka typów fontów, ale w trakcie stylizacji strony, uznałem ten krok za zbędny.

3.6 Elementy interaktywne stworzone przy pomocy CSS3: animacje.

Wisienką na torcie, na mojej stronie są widżety. Zdaje sobie sprawę, że stworzone animacje wprowadzają pewien estetyczny nieład na stronie, a także z faktu, iż do ich poprawniejszego wykonania przydałaby się znajomość JavaScript'u, ale nie mogłem się oprzeć wypróbowaniu nowych technik CSS3. Nie wstydzę się przyznać, że z początku nie miałem pojęcia jak posłużyć się pseudo-programistycznym językiem kreowania animacji na stronie i w efekcie musiałem zacząć modyfikować fragmenty kodu z różnych miejsc, takich jak:

<http://kurshtml5.edu.pl/animacje-css/>

Animacje wykorzystałem w dwóch miejscach na mojej stronie:

- **Pasek z nagłówkiem:**

Podczas jego tworzenia napotkałem na problem z właściwością *overflow*. Musiałem przekonać przeglądarkę, że gdy napis wychodzi poza pole strony to *znika* na pewien czas. Domyślnie Firefox poszerzał rozmiar okna, co nie wyglądało zbyt profesjonalnie. Poradziłem sobie z problemem ustawiając właściwość *y-overflow* z kontenerem całej strony `<html>`, na *hidden*. Zabieg ten uniemożliwił przesuwanie zawartości strony w poziomie, w efekcie czego strona przestała pokazywać pasków przewijania podczas wychodzenia napisu poza obszar layout'u.

```

61 /* ZAWARTOŚĆ HEADER */
62 header {
63     margin: 0;
64     margin-bottom: 1%;
65     background-color: rgba(0,0,0,0.7);
66     color: white;
67     font-family: 'loveloblack';
68 }
69 header > h1 {
70     margin: 0;
71     padding: 1%;
72     text-align: right;
73     animation: przesun 15s linear infinite;
74     font-size: 2em;
75 }
76 /* animacja CSS - do header > h1 */
77 @keyframes przesun
78 {
79     0% { transform:translateX(30%); }
80     100% { transform:translateX(-100%); }
81 }
82 /*-----*/

```

Ryc. nr [15](#) - Fragment kodu odpowiedzialny za animację nagłówka na stronie.

Tworzenie linearnych animacji jest proste, jak nie trudno się domyśleć czytając linijkę 73: *przesun* to nazwa animacji, 15s to czas jej trwania, linear to typ, zaś infinite informuje nas o nieskończonej ilości powtórzeń. Osobno określamy przekształcenia *keyframes* czyli klatek kluczowych, w tym wypadku określamy położenie obiektu na osi X w kolejnych stadiach animacji. Własności procentowe mogą wydawać się, w powyższym przykładzie, niejasne, ale ustaliłem, metodą prób i błędów, że takie właśnie ustawienia dają najlepszy efekt. Na wczesnym etapie pracy z animacjami możemy zgodzić się na pewne ustępstwa.

- **Przewijana rolka z hiperłączami obrazkowymi:**

Efekt rolki został również osiągnięty poprzez zastosowanie prostej linearnej animacji. W tym wypadku całość jednak wzbogaciłem o przejścia oraz zdarzenie polegające na zatrzymaniu animacji po najechaniu kursorem na grafikę w prawym kontenerze.

Użytkownik może spokojnie przyjrzeć się graficznym hiperłączom, które po najechaniu myszką przestają przesuwać się do góry i, w zależności od wyboru graficznego elementu, tracą własność *opacity*.

```

188 div#main_content > section > div#rool > div > a:hover {
189     border-color: #ff0066;
190     opacity: 1
191 }
192 div#main_content > section > div#rool > div > a:active {
193
194 }
195 div#main_content > section > div#rool > div > a > img {
196     width: 100%;
197     height: auto;
198 }
199 /* animacja CSS - div#main_content > section > div#rool */
200 @keyframes rool
201 {
202     0% { transform:translateY(60%); }
203     100% { transform:translateY(-100%); }
204 }
205 /*-----*/
206 div#main_content > section > div#rool {
207     animation: rool 30s linear infinite;
208
209 }
210 div#main_content > section > div#rool:hover {
211     animation-play-state: paused;
212 }

```

Ryc. nr [16] - Fragment kodu odpowiedzialny za działanie rolki z graficznymi hiperłączami.

Najbardziej zapierające dech w piersiach efekty, bo pamiętajmy, że pracujemy z technologią nie wymagającą od projektanta znajomości programowania, osiągnąć można stosując przejścia, animacje i animacje 3D. Najlepiej pokazuje to prezentacja zamieszczona na stronie:

<http://animateyourhtml5.appspot.com/pres/#1>

Łącząc rotacje, rozciąganie, skalowanie, przesuwanie; a także możliwość generowania krzywych możemy budować strony z wirtualną głębią. Dokument HTML5 przestaje być suchy, staje się prezentacją z mocno wyeksponowaną treścią w postaci animowanych grafik.

Zanim jednak płaskie strony WWW zmieniają się w wirtualne pokoje potrzeba trochę czasu: technologie są nadal w fazie eksperymentalnym, Adobe Flash jest nadal rozwijany i szybko nie ustąpi miejsca, ludzie są przyzwyczajeni do starych form eksponowania treści na stronie WWW.

Podsumowanie

Oceniam wykonany przede mną projekt jako otwarty. W moim zamyśle nie była próba stworzenia strony idealnej: odpowiadającej każdemu, odpornej na zmiany technologiczne, wielofunkcyjnej, zaawansowanej. Posłużyłem się najprostszą, a zarazem najbardziej fundamentalną metodą: stworzyłem witrynę *odręcznie*. Zapoznałem się z każdą linijką kodu, który osadzałem.

Mimo dostępnych dziś ułatwień w postaci: gotowców, szablonów, narzędzi, programów jestem zwolennikiem kreowania struktury witryny od zera, przy pomocy prostej aplikacji do kolorowania składni (w początkowym etapie przygody z webdesignem). Dogłębne poznanie podstawowych zasad osadzenia treści przy pomocy znaczników pozwala na lepszą orientację. Stronę, którą napiszemy sami zmienimy z łatwością, gdyż lepiej ją znamy i rozumiemy. Myślę, że dla każdego grafika HTML powinien stać się częścią jego elementarnej wiedzy obok znajomości obsługi oprogramowania.

Zachęcam do zgłębiania zasobów otwartego skarbcza jakim jest Internet, wyszukiwaniu i tworzeniu tutoriali oraz narzędzi. Sieć jest obecnie, tym czym jest, dzięki wkładowi każdego z jej użytkowników: dziś jesteś konsumentem jutro możesz stać się twórcą.

- **Uwagi**

Strona, którą stworzyłem nie była dokładnie testowana pod różnymi przeglądarkami. Pracowałem jedynie na Firefox 30 i 31. W zamyśle mojej pracy było pokazanie zmagania początkującego projektanta stron WWW. Założenie korzystania tylko z jednej przeglądarki nie wynika z braku chęci do pracy. Pewne drobne różnice na stronie będą widoczne w różnych przeglądarkach.

Nie opisałem etapu eksportu stron WWW na serwer, gdyż skoncentrowałem się głównie na etapie projektowania, obejmującego temat mojej pracy.

Zrezygnowałem z wykorzystania i przedstawienia, potężnego narzędzia do tworzenia interaktywnych stron WWW, Adobe Flash. Zdecydowały o tym dwa powody: Flash jest obecnie wypierany przez HTML5; strona, na której tworzyłbym animacje w postaci klipów z

wykorzystaniem ActionScript 3.0 nie mógłby zostać przeze mnie dokładnie opisana (programowanie to temat na osobną publikację).

Wszelkie grafiki dostępne na mojej stronie, a nie będące moją własnością, zostały podlinkowane. Nie miałem zamiaru używać ich komercyjnie. Grafiki nie mające linków są na licencji Creative Commons i są bezpłatne.

Błąd opisany na ryc. nr [3] nie będzie widoczny, gdyż w końcowym etapie mojego projektu wygenerowałem stronę index.

Może się zdarzyć, że animacje, przejścia i transformacje nie będą obsługiwane przez starsze przeglądarki, HTML5 nowym standardem wciąż rozwijanym.

BIBLIOGRAFIA

- [1] Web 2.0 - http://pl.wikipedia.org/wiki/Web_2.0 gdzie wykorzystano jako cytaty: Tim O'Reilly - "What is Web 2.0" <http://oreilly.com/web2/archive/what-is-web-20.html>, opublikowany dn. 09/30/2005.
- [2] Pisarski, *Hipertekst - definicje* [na stronie o nazwie:] *Techsty - literatura i nowe media* - <http://www.techsty.art.pl/hipertekst/definicje.htm>
- [3] Jason Cranford Teague (tłum. Łukasz Piwko, Maria Chaniewska), *CSS3: Szybki start*, wydawnictwo Helion, 2012, wydanie 5, str. 273.
- [4] Ibidem, ss. 302-303.
- [5] Małgorzata Kołtun, *JPEG*, opublikowano: 20.05. 2014 09:40 – Narodowy Instytut Audiowizualny: <http://www.nina.gov.pl/digitalizacja/s%C5%82ownik-digitalizacji/artyku%C5%82//2014/05/20/jpeg>
- [6] *Co to jest PNG? – definicja*, opublikowano 11.07. 2014 – Akademia Produkcji: <http://www.akademiaprodukcji.pl/png/>
- [7] Piotr Stępień - *SVG podstawy* opublikowano 25.07. 2011 – Centrum XP: <http://www.centrumxp.pl/InternetExplorer/2179,04-SVG-podstawy.aspx>
- [8] *Co to jest PNG? – definicja*, opublikowano 24.06. 2014 – Akademia Produkcji: <http://www.akademiaprodukcji.pl/gif/>
- [9] Michał Kortas / 5 sierpnia 2013 - *Filtry graficzne w CSS3 – przyszłościowa edycja zdjęć* opublikowano na stronie: <http://webroad.pl/html5-css3/670-filtry-graficzne-w-css3-przyszlosciowa-edycja-zdjec>
- [10] Spisano ze strony: <http://www.e-tapety.pl/grafika/definicje/grafika-wektorowa>
- [11] Spisano ze strony: http://pl.wikipedia.org/wiki/Grafika_rastrowa

SPIS ILUSTRACJI

[1] Web 2.0 - http://pl.wikipedia.org/wiki/Web_2.0 - Chmura znaczników Web 2.0 opracowana na podstawie artykułu Tima O'Reilly „What is Web 2.0” (źródło to samo jak w przypisie [1]).

[2] Wykonano na stronach:

<http://www.colorzilla.com/gradient-editor/>;

<http://fonty.pl/kolory.htm#generator>;

<http://www.fontsquirrel.com/fontface/generator>;

<http://animateyourhtml5.appspot.com/pres/#1>;

<http://kurshtml5.edu.pl/animacje-css/>;

[3] Zrzuty ekranu prezentują widok z Eksploratora Windows 7 oraz przeglądarki Mozilla Firefox.

[4] Zrzut ekranu prezentuje pracę z Dreamweaverze CS6, widok kodera.

[5] Zrzuty ekranu prezentują pracę przeglądarki Firefox, wgląd do starszych wersji www.onet.pl uzyskano dzięki Internet Archive: Digital Library - <https://archive.org/>

[6] Wykorzystano materiał graficzny dostępny na stronie: <http://www.bt4.pl/dokumentacje/css-d/css21/box#mpb-examples>, który jest tłumaczeniem artykułu dostępnego na <http://www.w3.org/TR/2009/CR-CSS2-20090908/> pt. *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification W3C Candidate Recommendation 08 September 2009* rdz. 8.1 Box dimensions

[7] Zrzuty ekranu prezentują pracę programów Adobe Illustrator CS6 oraz Adobe Photoshop CS6 (menu górne: plik - zapisz, plik - zapisz dla internetu, plik - eksportuj).

[8] Zrzuty ekranu prezentują pracę programu Adobe Photoshop CS6 (menu górne - filtr - galeria filtrów)

[9] Zrzut ekran prezentuje pracę programu Adobe Photoshop CS6 (menu boczne - kadrowanie)

[10] Zrzut ekran prezentuje pracę programu Adobe Photoshop CS6 (menu boczne - crop tools - cięcie na plasterki) oraz widok z eksploratora Windows 7

[11] Zrzut ekranu prezentuje pracę przeglądarki Mozilla Firefox 30

[12] Zrzut ekranu prezentuje pracę programu Dreamweaver CS6 (widok kodera)

[13] Zrzut ekranu prezentuje pracę programu Dreamweaver CS6 (widok kodera), wgląd do arkusza stylu, który znajduje się w folderze font. Zastosowano czcionki ubuntu i lovelo dostępnymi na zasadzie licencji Creative Commons: <http://fontfabric.com/lovelo-font/> <http://www.fontsquirrel.com/fonts/ubuntu>

[14] Zrzuty ekranu prezentuje pracę przeglądarki Mozilla Firefox 30, wykorzystano pliki HTML z osadzoną treścią wygenerowane przez *FontSquirrel*.

[15], [16] - Zrzut ekranu prezentuje pracę programu Dreamweaver CS6 (widok kodera)